Alma Mater Studiorum - Università di Bologna

DOTTORATO DI RICERCA IN

INGEGNERIA ELETTRONICA, TELECOMUNICAZIONI E TECNOLOGIE DELL'INFORMAZIONE

Ciclo 35

Settore Concorsuale: 09/F2 - TELECOMUNICAZIONI

Settore Scientifico Disciplinare: ING-INF/03 - TELECOMUNICAZIONI

TECHNOLOGIES FOR URBAN AND RURAL INTERNET OF THINGS

Presentata da: Riccardo Marini

Coordinatore Dottorato

Aldo Romani

Supervisore

Chiara Buratti

Co-supervisore

Roberto Verdone

Esame finale anno 2023

SCHOOL OF ENGINEERING AND ARCHITECTURE

PHD PROGRAM IN
ELECTRONICS, TELECOMMUNICATIONS,
AND INFORMATION TECHNOLOGY ENGINEERING
(ET-IT)

SETTORE CONCORSUALE 09/F2
SETTORE SCIENTIFICO DISCIPLINARE ING-INF/03

PhD Thesis on

TECHNOLOGIES FOR URBAN AND RURAL INTERNET OF THINGS

Supervisor
Prof. CHIARA BURATTI

Candidate
RICCARDO MARINI

Co-supervisor
Prof. ROBERTO VERDONE

PhD Program Coordinator
Prof. ALDO ROMANI

CYCLE XXXV
DEFENCE SESSION OF 2023

Abstract

Nowadays, application domains such as smart cities, agriculture, or intelligent transportation, require communication technologies that combine long transmission ranges and energy efficiency to fulfill a set of capabilities and constraints to rely on. In addition, in recent years, the interest in Unmanned Aerial Vehicles (UAVs) providing wireless connectivity in such scenarios is substantially increased thanks to their flexible deployment. The first chapters of this thesis deal with LoRaWAN and Narrowband-IoT (NB-IoT), which recent trends identify as the most promising Low Power Wide Area Networks technologies. While LoRaWAN is an open protocol that has gained a lot of interest thanks to its simplicity and energy efficiency, NB-IoT has been introduced from 3GPP as a radio access technology for massive machine-type communications inheriting legacy LTE characteristics. This thesis offers an overview of the two, comparing them in terms of selected performance indicators. In particular, LoRaWAN technology is assessed both via simulations and experiments, considering different network architectures and solutions to improve its performance (e.g., a new Adaptive Data Rate algorithm). NB-IoT is then introduced to identify which technology is more suitable depending on the application considered. The second part of the thesis introduces the use of UAVs as flying Base Stations, denoted as Unmanned Aerial Base Stations, (UABSs), which are considered as one of the key pillars of 6G to offer service for a number of applications. To this end, the performance of an NB-IoT network are assessed considering a UABS following predefined trajectories. Then, machine learning algorithms based on reinforcement learning and meta-learning are considered to optimize the trajectory as well as the radio resource management techniques the UABS may rely on in order to provide service considering both static (IoT sensors) and dynamic (vehicles) users. Finally, some experimental projects based on the technologies mentioned so far are presented.

Contents

List of Acronyms

2D 2nd dimension

3D 3rd dimension

3DN 3D Network

3DQN Double Dueling Deep Q Network

3GPP 3rd Generation Partnership Project

4G fourth-generation

5G fifth-generation

6G sixth-generation

ABP Activation by Personalization

ACK acknowledgement

ADR Adaptive Data Rate

ATG Air-to-Ground

AWGN Additive White Gaussian Noise

BLER Block Error Rate

BPSK Binary Phase Shift Keying

BS Base Station

BSR Buffer Size Report

BW frequency-sweep interval

CA Cooperative Awareness

CAM Cooperative Awareness Message

CE	Coverage Enhancement
CoMPS	Continual Meta Policy Search
CP	Collective Perception
CPM	Collective Perception Message
CPS	Collective Perception Service
CR	Coding Rate
CSI	Channel State Information
CSS	Chirp Spread Spectrum
DC	Duty Cycle
DCI	Downlink Control Indicator
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q-Network
DNN	Deep Neural Network
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
DRX	Discontinuous Reception
ED	End Device
eDRX	extended Discontinuous Reception
EDT	Early Data Transmission
eNB	evolved Node-B
ESN	Echo State Network
FDD	Frequency Division Duplexing
FEC	Forward Error Correction
FIFO	First In First Out
GD	gradient descent

GoB	Grid-of-Beams
GUE	Ground User Equipment
GW	Gateway
HARQ	Hybrid Automatic Repeat Request
ILP	Integer Linear Program
IoT	Internet of Things
IP	Internet Protocol
ISD	inter-site distance
ISM	industrial, scientific and medical
ITS-S	Intelligent Transport System Station
KPI	Key Performance Indicator
LAN	Local Area Network
LL	Link Layer
LLC	Logical Link Control
LoRaWAN	Long Range Wide Area Network
LoS	Line of Sight
LPWAN	Low Power Wide Area Network
LSP	large-scale parameter
LTE	Long Term Evolution
LTE-M	Long Term Evolution for Machines
MAC	Medium Access Control
MAML	Model Agnostic Meta Learning
MBS	macro Base Station
mBS	micro Base Station
MCS	Modulation and Coding Scheme

MDP Markov decision process

MIB Master Information Block

ML Machine Learning

mMTC massive Machine-Type Communication

MNO Mobile Network Operator

MTC Machine Type Communications

NB-IoT Narrowband-IoT

NFZ no-fly zone

NLoS Non-Line of Sight

NN Neural Network

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

NS Network Server

NSSS Narrowband Secondary Synchronization Signal

O2I Outdoor-to-Indoor

O2O Outdoor-to-Outdoor

OFDM Orthogonal Frequency Division Multiplexing

OFDMA Orthogonal Frequency Division Multiple Access

OHE One Hot Encoding

OTAA Over-The-Air Activation

PHY Physical

POMDP	Partially Observable Markov Decision Process
PPP	Poisson Point Process
PRB	Physical Resource Block
PSM	Power Saving Mode
QoE	Quality of Experience
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
RA	Random Access
RACH	Random Access Channel
RAR	Random Access Response
RB	Resource Block
RE	Resource Element
RL	Reinforcement Learning
RMSE	Root Mean Squared Error
RRC	Radio Resource Control
RRM	Radio Resource Management
RSRP	Radio Signal Received Power
RSSI	Received Signal Strength Indicator
RSU	Road Side Unit
RU	Resource Unit
RX1	Receive Window 1
RX2	Receive Window 2
SAR	Search and Rescue
SC-FDMA	Single Carrier Frequency Division Multiple Access
SF	Spreading Factor

SIB	System Information Block
SIC	Successive Interference Cancellation
SINR	Signal-to-Interference-plus-Noise Ratio
SIR	Signal-to-Interference Ratio
SNR	Signal-to-Noise Ratio
SSB	Signal Synchronization Block
SSP	small-scale parameter
SUMO	Simulation of Urban Mobility
TBS	Transport Block Size
TCP	Thomas cluster process
TDD	Time Division Duplexing
ToA	Time on Air
TR	Technical Report
TSP	Traveling Salesman Problem
UABS	Unmanned Aerial Base Station
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
UE	User Equipment
UMa	Urban Macro
UMi	Urban Micro
UTM	Unmanned Aerial System Traffic Management
UUE	Unmanned Aerial User Equipment
V2C	Vehicle-to-Cellular
V2I	Vehicle-to-Infrastructure
V2R	Vehicle-to-Roadside

V2V Vehicle-to-Vehicle

V2X Vehicle-to-Everything

List of Figures

List of Tables

List of Algorithms

Introduction

The Internet of Things (IoT) is a system of interrelated computing devices and everyday objects which are able to transfer data over a network without requiring human-to-human or human-to-machine interaction, by communicating over the Internet [1]. The idea behind IoT is to revolutionize the way we live and work: smart city or smart home applications [2], as well as smart agriculture [3] or Industry 4.0 [4], are a few among all the possible applications. Some IoT connectivity technologies are prevalent in a specific application domain, such as Bluetooth Low Energy in Personal Area Networks [5] and Zigbee in Home Automation systems [6]. Others, like Wi-Fi Low Power, Low Power Wide Area Networks (LPWAN) [7] and cellular systems, such as the Long Term Evolution for Machines (LTE-M) and Narrowband-IoT (NB-IoT), have a much broader scope. Notably, this landscape is constantly and rapidly evolving, with new technologies being regularly proposed, and existing ones being updated and modified. This facilitates their further proliferation into new application domains.

This thesis, at first, is dedicated to the two most promising Low Power Wide Area Network (LPWAN) technologies [8], namely Long Range Wide Area Network (LoRaWAN) and NB-IoT. LoRaWAN has been released by the LoRa Alliance as an open protocol in 2015 [9]. At the Physical (PHY) layer, it relies on LoRa, a proprietary solution developed by Cycleo and later acquired by Semtech in 2012. The main benefits of LoRaWAN consist of very low energy consumption and a large communication range, at the cost of having a low data rate. Such characteristics make it very suitable for a lot of common use cases in smart cities and smart agriculture. NB-IoT was introduced by 3rd Generation Partnership Project (3GPP) in Release 13 as an alternative solution to the LPWAN technologies already available on the market (e.g., LoRaWAN). It is based on Long Term Evolution (LTE), even though it can be considered an adaptation for low-cost Machine Type Communications (MTC).

In the following, their main characteristics will be presented and their performance in terms of a variety of Key Performance Indicators (KPIs), which concern both the PHY and the Medium Access Control (MAC) layers, will be investigated. In addition, specific solutions for LoRaWAN networks, such as the introduction of a new Adaptive Data Rate (ADR) algorithm as well an analysis of fog-based network deployments, will be discussed. In addition, some experimental activities have been carried out while working on the topics mentioned above. Accordingly, this thesis will present an innovative system exploiting LoRaWAN and based on machine learning techniques able to estimate soil moisture without the need for ad-hoc sensors. It will be shown that by exploiting the underground radio wave propagation, it is possible to extract useful information which are sufficient to measure with a good grade of approximation the humidity of a terrain.

After considering static scenarios with fixed terrestrial Base Stations (BSs) and ground users, the role of Unmanned Aerial Vehicle (UAV) (a.k.a. drones) acting as flying BS (UABS) will be investigated. Nowadays, the rising and relevance of UAVs in civil applications such as delivery, photogrammetry, monitoring, and others, call for technology, safety, and security improvements in the UAVs' construction technology. For these reasons, it can be predicted an increase in the use of UAVs even in urban environments for a variety of new applications aiming at improving citizens' daily lives. With this perspective, multiple UAVs providing the cellular network service for urban and rural applications could be easily envisaged. As a matter of fact, the use of UABSs may deeply improve network performance by matching stringent requirements on communication performance imposed by the several applications cited before, for example, by providing dedicated ground-to-UAVs links.

After dealing with a usual NB-IoT network, in this thesis, a UAV-aided NB-IoT network will be analyzed, by considering urban scenarios with an UABS following predefined trajectories according to the distribution of static IoT nodes. Moreover, even though until now the usage of UABSs has been considered especially suited for MTC and IoT links, since ground nodes are usually static (i.e., they do not change their position over time), an important use case is provided by considering mobile users (e.g., vehicles), which makes the exploitation of UABSs even more promising. It is indeed foreseen that beyond fifth-generation (5G) networks will require degrees of flexibility that current technologies do not provide, thus the role of UABSs in such networks is gaining much attention.

Therefore, the second part of the thesis is focused on vehicular users and use cases, which deeply change the assumptions considered when dealing with static users. UABSs can support high demanding Vehicle-to-Everything (V2X) applications, such as advanced driving [10, 11] and extended sensing [12, 13], as specified by 3GPP [14]. To this end, the trajectory design assumes a fundamental role in order to match network requirements, therefore machine learning-based solutions (Reinforcement Learning and Meta-Learning) will be presented in order to address such problems. The use of machine learning algorithms outperforms standard optimization tools when dealing with complex problems which would require deep simplifications, such as the one proposed in the following.

Besides the trajectory design, another important problem that will be considered is the design of efficient Radio Resource Management (RRM) schemes when dealing with UAV-aided network. As a matter of fact, proper coordination between the terrestrial BSs and flying UABSs must be carefully conceived in order to guarantee optimal performance.

The remainder of the thesis is organized as follows. Chapter 1 focuses on LoRaWAN by providing a description of the technology as well as a literature overview; after this, a full analysis of the performance LoRaWAN network is provided, in usual cloud-based scenarios as well as fog-based ones. At the same time, a new ADR algorithm is presented and compared with the standard solution. Chapter 2 provides an overview of the NB-IoT technology as well as a detailed comparison with LoRaWAN. In Chapter 3, the focus moves towards the use of UABS in a NB-IoT network, by offering insights on the use of a moving BS and the improvement offered in terms of network performance by addressing different UABS predefined trajectories and varying different network parameters. Chapter 4 is focused on the design of trajectories for UABSs by exploiting Reinforcement Learning (RL)-based and meta-learning-based techniques and addressing the possibility of having moving vehicles in an urban scenario. Chapter 5 is then dedicated to the study of RRM techniques for UABS-aided networks. Finally, Chapter 18 concludes the thesis. In the Appendix, the experimental activity related to the deployment of the Sensing-without-Sensors system for soil moisture estimation is described.

Chapter 1

LoRaWAN

1.1 Introduction

In large-scale IoT networks, long-range connectivity and energy efficiency are challenging requirements, which have been met by LPWAN technologies, such as LoRaWAN, NB-IoT [15], SigFox [16]. Such networks combine a long battery life and wide coverage, at the cost of a low bit rate. Among those, LoRaWAN is deemed as one of the most promising, being relatively flexible and straightforward, both technology- and business-wise [17, 18]. In a LoRaWAN network, End Devices (EDs), representing IoT nodes (e.g., sensors or actuators), send packets to Gateways (GWs), which forward them to a Network Server (NS) they are connected to via Internet (e.g., using WiFi, 4G/5G, Ethernet, etc.). The NS serves as a centralized entity, responsible for network upkeeping.

At the physical layer, LoRaWAN relies on LoRa, which is a proprietary physical layer solution patented by Semtech Corporation[1], while the MAC and network layers have been defined by the LoRa Alliance[2] in the respective specifications [9]. The LoRa modulation is based on chirp spread spectrum, which exploits chirps whose frequency increases or decreases linearly over a certain amount of time.

LoRaWAN EDs mainly operate in license-free industrial, scientific and medical (ISM) bands, whose availability and usage conditions somewhat differ for various regions of the globe [19]. Specifically, in the EU, the most widespread implementation is based on the 863-870 MHz ISM band, even though a version of LoRa working at 2.4 GHz is gaining much attention [20].

First, this chapter proposes the architecture of a new LoRaWAN simulator, LoRaWANSim, which has been released as open access, that integrates both MAC layer and PHY layer functionalities. Second, original results obtained by means of the simulator are provided, which give an insight into the performance of LoRaWAN and the impact of different network setups.

As for the simulator itself, the novelty of the approach lies in the integration of two separate simulators, namely, the MAC layer simulator and the PHY layer simulator, which interact with each other at run-time. In particular, the PHY layer simulator is in charge of providing the outcome (success/failure) of every sin-

[1]https://www.semtech.com/lora

[2]https://lora-alliance.org/

gle transmission to the MAC layer simulator, which knows who is transmitting to whom and when. This depends on a number of aspects and parameters (e.g., experienced signal-to-noise ratio, adopted Spreading Factor (SF) and Coding Rate (CR), interleaving, Gray coding, modulation/demodulation) that are accurately considered/reproduced at the physical layer. This level of integration is not usual in currently available LoRaWAN simulators, as well as in network simulators, in general. Among the other unique features, which are not present in the state-of-the-art tools are (i) support of multiple gateways, (ii) possibility of prioritizing and enabling/disabling the receive windows, (iii) accounting for uplink-downlink interference in RX1, and modeling half or full-duplex LoRaWAN gateways. The simulator is also available for free on https://github.com/kvmikhayl/LoRaWAN_simulator.

As for the numerical results, several original results are provided. Specifically, the impact of different CRs in interference-limited and noise-limited scenarios is investigated, also showing that in heavily interference-limited conditions, the adoption of powerful coding rates is counterproductive for both the delivery rate and the energy consumption. Moreover, the impact of downlink transmissions (e.g., acknowledgments) on the average energy consumption of EDs is assessed, showing that increasing the number of gateways affects not only the packet delivery rates in uplink and downlink but also the average consumption of devices, hence, ultimately, the battery life.

After the description of the simulator, this chapter proposes a novel ADR algorithm to be implemented at the network server, denoted as Collision-Aware ADR (CA-ADR), aiming at finding a set of data rates to be assigned to EDs such that packet success rate and network throughput improve w.r.t. the standard ADR solution. The algorithm exploits the orthogonality of signals emitted with different data rates [21] and tries to minimize interference while maintaining connectivity toward the GW. The algorithm has been tested and compared with the standard solution as well as another approach presented in the literature [22, 23], adopting an integrated methodology involving both simulations and experiments. In particular, a small testbed composed of one ED sending data to a GW, connected to a NS, has been used to characterize delays, both in terms of processing time and transmission time in the different links (wireless and wired). A traffic emulator has been developed to stress the NS and characterize its performance in the presence of high network traffic. The outcomes of these experiments have been provided as input to a simulator, evaluating the network-level performance when considering a large number of EDs deployed in a given area. The performance of CA-ADR has been evaluated assuming the LoRaWAN network deployed in both a cloud and a fog computing scenario. In particular, two different setups for the NS, in terms of computational capabilities and location, have been considered. The impact of the NS deployment on the network performance has been assessed.

The remainder of the chapter is organized as follows: Section 1.2 provides details about the LoRaWAN technology. Section 1.3 describes the state of the art related to LoRaWAN, Section 1.4 introduces LoRaWAN-Sim, going through each block and performance metric obtainable, as well as its validation, and provide some interesting results. After, Section 1.5 illustrates the new ADR algorithm and Section 1.6 provides some details about the cloud and fog architecture in LoRaWAN. Finally, Section 1.7 concludes the chapter

1.2 Technology background

Fundamentals of LoRa and LoRaWAN relevant to understanding the key aspects of the protocol and architecture are presented and discussed in the following.

1.2.1 LoRa PHY

At the PHY layer, LoRa adopts the M-ary Chirp Spread Spectrum (CSS) modulation, which is based on *chirps*, that is, sine-wave signals whose frequency sweeps linearly with time. Denoting with f_0 the central frequency of the sweep interval $[f_0 - \frac{\mathrm{BW}}{2}, f_0 + \frac{\mathrm{BW}}{2}]$, and assuming $t = 0$ as the signal starting instant, a single LoRa chirp can be mathematically expressed as

$$c(t) = V_0 \cos\left(2\pi f_0 t + 2\pi \int_0^t \Delta f(s,\xi) d\xi + \phi_0\right), \quad 0 \leq t \leq T_s \tag{1.1}$$

where

- $V_0 > 0$ is the chirp amplitude,
- $s \in \{0, \cdots, M-1\}$ is the modulation symbol;
- $\Delta f(s,t)$ is the symbol-dependent instantaneous frequency-offset, ranging in the interval $[-\frac{BW}{2}, \frac{\mathrm{BW}}{2}]$,
- ϕ_0 is the signal phase at the initial instant $t = 0$,
- T_s is the chirp duration.

In particular, LoRa uses M differently-shaped chirps, each of which is in one-to-one correspondence with the M symbols of the modulation alphabet $\mathcal{S} = \{0, \cdots, M-1\}$: given a modulation symbol $s \in \mathcal{S}$, the corresponding $\Delta f(s,t)$ linearly increases starting from $-\frac{\mathrm{BW}}{2}$. Then, when the maximum frequency-offset $\frac{\mathrm{BW}}{2}$ is reached, $\Delta f(s,t)$ wraps around to $-\frac{\mathrm{BW}}{2}$ and keeps on increasing linearly until $\Delta f(s, t = T_s) = \Delta f(s, 0)$. T_s, which represents the chirp duration, is usually referred to as *symbol interval*.

The modulation parameters are chosen such that

- $\mathrm{BW} \in \{125, 250, 500\}$ kHz,
- $M = 2^{\mathrm{SF}}$, with SF denoting the Spreading Factor,
- SF $\in \{7, 8, 9, 10, 11, 12\}$,
- $T_s \mathrm{BW} = M$.

By operating with high SF values, LoRa transmitters increase their communication range [24]; robustness against channel impairments and interference from third systems, frequency selectivity and Doppler effect, is also enhanced by increasing SFs. On the other hand, this results in a low data rate and long Time on Air (ToA), which are the main drawbacks of operating with high SFs.

The physical layer bit rate R_b depends on SF, the sweep interval frequency-sweep interval (BW) and the coding rate CR of the Forward Error Correction (FEC) mechanism (which encodes 4 data bits into codewords of 5..8 bits for $CR \in [1, .., 4]$, respectively) and is given by [25]:

$$R_\text{b} = \text{SF} \cdot \frac{\frac{4}{4+\text{CR}}}{\frac{2^\text{SF}}{\text{BW}}} \quad [\text{bit/s}]. \tag{1.2}$$

The indicative values (denoting the maximum instantaneous data rate) are provided in Table 1.1 assuming, as an example case, BW=125 kHz and CR=1.

Table 1.1: Bit Rate with BW=125 kHz and CR=1

Spreading Factor	7	8	9	10	11	12
Bit rate [bit/s]	5468	3125	1757	976	537	293

The bit rate has a direct impact on the ToA of transmissions, which is the time needed to transmit a packet on the wireless channel and it is computed [26] as:

$$T_\text{symbol}(\text{SF}) = \frac{2^\text{SF}}{\text{BW}} \quad [\text{s}] \tag{1.3}$$

$$T_\text{preamble}(\text{SF}) = (L_\text{preamble} + 4.25) \cdot T_\text{symbol}(\text{SF}) \quad [\text{s}] \tag{1.4}$$

$$L_\text{payload} = 8 + \left\lceil \frac{(8B - 4\text{SF} + 28 + 16 - 20H)}{(4\text{SF})} \right\rceil \cdot (\text{CR} + 4) \tag{1.5}$$

$$T_\text{payload}(\text{SF}) = L_\text{payload} \cdot T_\text{symbol}(\text{SF}) \quad [\text{s}] \tag{1.6}$$

$$\text{ToA}(\text{SF}) = T_\text{preamble}(\text{SF}) + T_\text{payload}(\text{SF}) \quad [\text{s}] \tag{1.7}$$

where B is the payload size in bytes, $H = 0$ when the header is enabled and $H = 1$ when no header is present, L_preamble and L_payload are the length in symbols of the preamble and the payload respectively.

1.2.2 LoRaWAN

On top of the LoRa PHY, the LoRaWAN protocol builds up the upper protocol layers. Communication between EDs and GWs is implemented via LoRa, whereas GWs are connected via standard Internet Protocol (IP) connections, such as Wi-Fi, Ethernet, or 4G/5G, to the NS, resulting in a star-of-stars topology. EDs are not associated to a specific GW but rather to a NS and all GWs receiving data from an ED forward them to the NS, which is then in charge of discarding duplicates, sending acknowledgements (ACKs) and managing the overall network.

The access to the radio channel is ALOHA-based complemented by selecting one of the available frequency channels, so an ED sends a packet whenever it has data to send. By standard, the network channels can be freely attributed by the network operator. However, in the EU 868 MHz band, three default channels must be implemented in every ED [19]. Those channels are the minimum set that all network gateways should

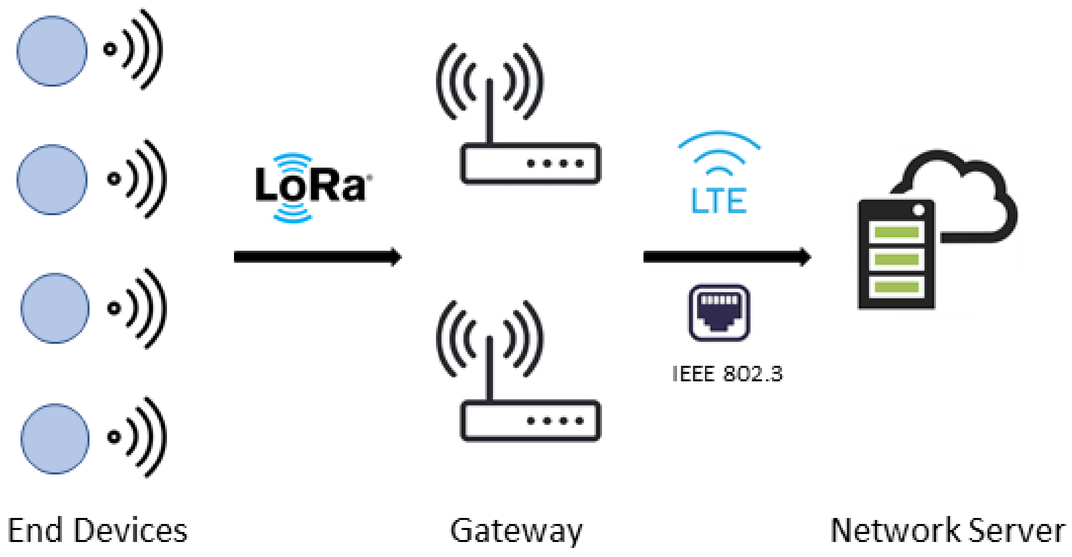


Figure 1.1: LoRaWAN Network Architecture

Channel Frequency [MHz]	868.1	868.3	868.5
Bandwidth [kHz]	125		

Table 1.2: EU868 Default Channels

always be listening on. They are reported in Table 1.2. In addition to these, extra channels (up to 16 in total) can be optionally configured.
LoRaWAN defines three classes of devices, which differ primarily with respect to the support of the downlink communication capabilities and which are labeled A, B and C.

- Class A devices' uplink transmission is followed by up to two short downlink receive slots (denoted Receive Window 1 (RX1) and Receive Window 2 (RX2), respectively) after two different fixed RECEIVE_WINDOW_DELAY intervals (the standard suggests using 1 s delay between the end of an uplink and RX1, and 2 s delay between the end of an uplink and RX2), as it can be seen in Figure 1.2. Class A is primarily intended for EDs with limited energy availability (e.g., battery-powered ones), such as sensors since it provides the lowest power consumption.

- Class B devices allow for more than two receive slots, opening extra receive windows at scheduled times. Synchronization between EDs and GWs is kept via periodic beacons broadcast by a GWs.

- Class C devices have a nearly continuously open receive window except for the time spent in uplink transmissions, so they are always reachable by the network, which is attained at the cost of increased power consumption.

Furthermore, two transmission modes, which can be selected on a per-packet basis, are defined by the LoRaWAN specification:

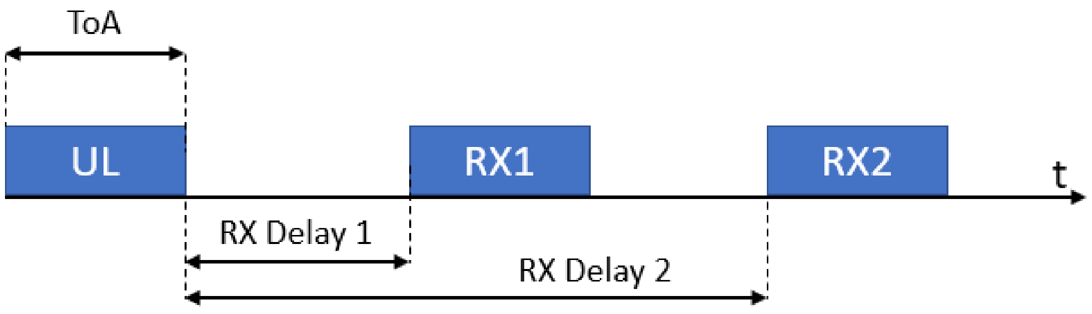


Figure 1.2: RX Windows in Class A

- Confirmed Mode: when an ED sends an uplink packet, it expects to receive an ACK packet from the NS (through the GW) after its transmission, and it may continue transmitting the same message if no ACK is received.
- Unconfirmed Mode: no ACK is sent by the NS, so the ED does not know if the packet has been correctly received.

1.2.3 Standard ADR Algorithm

One of the key operations performed by the NS is to implement an ADR algorithm to dynamically set the data rate and transmit power to be used by a given ED during its communication. In the following, the most widespread implementation is considered, that is the one used by The Things Network or ChirpStack, which is based on Semtech's recommended algorithm [27]. This algorithm is specifically designed to allow connectivity between EDs and GWs. However, the standard algorithm [27] does not take into account the interference problems that may be present when the offered traffic increases. Indeed, since a simple ALOHA-based protocol is used by EDs to access the channel, collisions, and packet losses may dramatically impair the network throughput, especially when low data rates are used for transmissions [21]. The standard ADR tries to assign the lowest value of SF, allowing connectivity between the ED and the GW, in order to reduce as much as possible energy consumption.

In addition to the ADR algorithm working at the NS, which is designed by the server developer, there exists another ADR algorithm, working at the ED side, specified by LoRa Alliance [9], designed to keep track of the connection only.

EDs are in charge of deciding if ADR should be used or not. When activated, the NS will control the transmission parameters of the device sending ADR-specific commands. Besides, the device should periodically check whether the NS still receives its uplink frames, otherwise, it should autonomously set its SF.

Algorithm 1 shows the implementation at the NS used to decide about the transmission parameters (specifically, SF and P_{T}) each ED connected to the NS should set. The NS runs the algorithm each time an uplink packet from a given ED is received and decisions on data rates are based on measurements performed over the last k packets received from the ED (by default, $k = 20$). In particular, at first, the $\mathrm{SNR}_{\mathrm{margin}}$ is measured

Algorithm 1: Network Server Adaptive Data Rate Algorithm

```
   Input: $\mathrm{SF}^{(\mathrm{temp})} = \mathrm{SF}(t)$, $P_{\mathrm{T}}^{(\mathrm{temp})} = P_{\mathrm{T}}(t)$, $\mathrm{SF}_{\mathrm{min}} = 7$
   $\mathrm{SNR}_{\mathrm{max}} = \max\{\text{last k uplink packets received}\}$,
   $\mathrm{SNR}_{\mathrm{min}}$ given in Table 1.3 by setting SF$(t)$,
   $\mathrm{SNR}_{\mathrm{margin}} = \mathrm{SNR}_{\mathrm{max}} - \mathrm{SNR}_{\mathrm{min}} - M$,
   $N_{\mathrm{step}} = \left\lfloor \frac{\mathrm{SNR}_{\mathrm{margin}}}{3} \right\rfloor$
   Output: SF$(t+T)$, $P_{\mathrm{T}}(t+T)$
1  if $N_{\mathrm{step}} > 0$ then
2      while $N_{\mathrm{step}} > 0$ & $\mathrm{SF}^{(\mathrm{temp})} > \mathrm{SF}_{\mathrm{min}}$ do
3          $\mathrm{SF}^{(\mathrm{temp})} = \mathrm{SF}^{(\mathrm{temp})} - 1$
4          $N_{\mathrm{step}} = N_{\mathrm{step}} - 1$
5      while $N_{\mathrm{step}} > 0$ & $P_{\mathrm{T}_{\mathrm{temp}}} > P_{\mathrm{T}_{\mathrm{min}}}$ do
6          $P_{\mathrm{T}}^{(\mathrm{temp})} = P_{\mathrm{T}}^{(\mathrm{temp})} - 3\,\mathrm{dB}$
7          $N_{\mathrm{step}} = N_{\mathrm{step}} - 1$
8  else
9      while $N_{\mathrm{step}} < 0$ & $P_{\mathrm{T}_{\mathrm{temp}}} < P_{\mathrm{T}_{\mathrm{max}}}$ do
10         $P_{\mathrm{T}}^{(\mathrm{temp})} = P_{\mathrm{T}}^{(\mathrm{temp})} + 3\,\mathrm{dB}$
11         $N_{\mathrm{step}} = N_{\mathrm{step}} + 1$
12 return SF$(t+T) = \mathrm{SF}^{(\mathrm{temp})}$, $P_{\mathrm{T}}(t+T) = P_{\mathrm{T}}^{(\mathrm{temp})}$
```

as reported in algorithm 1, where $\mathrm{SNR}_{\mathrm{max}}$ is the maximum Signal-to-Noise Ratio (SNR) among the last k collected packets, M is a margin set a priori ($M = 10$ dB by default), and $\mathrm{SNR}_{\mathrm{min}}$ is the minimum SNR required to correctly demodulate the received signal, computed considering the initial value of SF (see Table 1.3). Then, an iterative process starts. $\mathrm{SNR}_{\mathrm{margin}}$ is used to compute N_{step}, which indicates how many times the iteration will run. In particular, if N_{step} is greater than 0, both N_{step} and the SF are decremented by one unit at each step, until either the minimum SF is reached (SF=7) or N_{step} reaches zero. If the iteration has not been completed yet ($N_{\mathrm{step}} > 0$), then the transmitted power is also decremented by 3 dB at each further step, until either it reaches the minimum value (2 dBm by standard) or the iteration terminates ($N_{\mathrm{step}} = 0$). On the other hand, if the initial value of N_{step} is lower than 0, then both N_{step} and the transmitted power are incremented at each step (by one unit and 3 dB, respectively) until either the maximum power (14 dBm) or the last iteration ($N_{\mathrm{step}} = 0$) is reached. In both cases, the algorithm stops at most after N_{step} steps.

As far as the ED is concerned, it will receive information from the NS about the transmission parameters to be used, but then it will continuously track the connection with the NS. Algorithm 2 reports the ADR implementation at each ED. The description is valid in the case of confirmed transmission mode (extension to the unconfirmed mode is straightforward [9]). Each time an ED sends an uplink packet without receiving an ACK, the counter ADR_ACK_CNT is incremented; after ADR_ACK_LIMIT (by default 64) messages without any downlink response, the device sends a request to NS, which must respond within the next ADR_ACK_DELAY

Data Rate	**Spreading Factor**	$\mathrm{SNR}_{\mathrm{min}}$ **[dB]**
0	12	-20
1	11	-17.5
2	10	-15
3	9	-12.5
4	8	-10
5	7	-7.5

Table 1.3: $\mathrm{SNR}_{\mathrm{min}}$ values for different SFs with BW=125 kHz.

(by default 32) frames with a downlink frame. If no reply is received, the ED must try to reconnect to the network, by first setting the transmitted power to its default value and then possibly switching to the next lower data rate. The device must lower its data rate every time the ADR_ACK_DELAY expires.

Algorithm 2: End Device Adaptive Data Rate Algorithm

```
Input: $\mathrm{SF}_{\mathrm{temp}}$ set to the current SF value assigned, $\mathrm{ADR\_ACK\_CNT} = 0$
Output: New value of SF
1  while uplink transmission do
2      if no ACK received then
3          ADR_ACK_CNT=ADR_ACK_CNT+1
4          if ADR_ACK_CNT=ADR_ACK_LIMIT then
5              request downlink response from NS
6          if ADR_ACK_CNT ≥ ADR_ACK_LIMIT+ADR_ACK_DELAY then
7              $\mathrm{SF}_{\mathrm{temp}} = \mathrm{SF}_{\mathrm{temp}} + 1$
8      else
9          $ADR\_ACK\_CNT = 0$
10 return $\mathrm{SF}_{\mathrm{temp}}$
```

1.3 Literature Overview

Given the LoRaWAN potential, the research community has been dedicating significant efforts to its study, both theoretically and experimentally [28]. A number of open-source network simulators have been developed over the recent years [29]; however, most of them have been created for validating one specific research target. Therefore, they tend to be focused on a specific aspect and are often limited by a number of assumptions, and cannot properly capture the operation of the LoRaWAN protocol stack as a whole. Another somewhat limiting factor is the steep learning curve required to use these tools, which is due to the use of highly effective, but rather complex, discrete-events simulation frameworks.

One of the most popular LoRaWAN simulators available today is LoRaSim [30], a discrete-event simulator realized using SimPy. It can emulate a network of devices and gateways randomly positioned in a 2-dimensional grid and implements a channel model based on the well-known log-distance path loss. Two performance metrics are provided as outputs: Data Extraction Rate and Network Energy Consumption, which refer to the overall network and not to the behavior of the individual ED. LoRaSim, though, lacks a number of features, such as accounting of the imperfect SF orthogonality, downlink traffic and Duty Cycle (DC) limitation. A closed-source expansion of LoRaSim is presented in [31]. The main improvement concerns bidirectional communication, but the other features are still missing. An ns-3[3] [32]. based simulator implementing class A is presented in [33], and it has been extended by the authors themselves to include also downlink traffic [34]. However, there is no characterization of the energy performance. LoRaFREE [35] implements downlink communications and SF imperfect orthogonality in a network with one full-duplex GW and EDs generating periodic traffic only; in addition, the channel is based on the log-distance path loss model. LoRaEnergySim [36], instead, is a tool based on the WiMOD iC880A GW, which focuses on energy consumption, even though it still assumes perfect SF orthogonality.

Many recent studies focus on the evaluation of the performance of LoRaWAN networks. In [17] and [18], LoRaWAN technology was compared to other LPWAN technologies, that are Sigfox and NB-IoT, to point out the advantages of LoRaWAN in terms of battery lifetime and capacity in many scenarios. The coverage of LoRa communications has been addressed in [37], where authors show that the maximum communication range can reach up to 10 km with a small percentage of packets lost. A theoretical analysis of the achievable uplink throughput has been carried out in [38], where the effect of the SF allocation and the impact of SF imperfect orthogonality on the overall throughput are taken into account.

There exist also a number of works focused on the ADR mechanism. In [22, 23] it is proposed to estimate the link quality to be used as input to the ADR algorithm based on the average SNR, rather than the maximum SNR as done in the standard solution. This approach is also considered a benchmark to evaluate the CA-ADR algorithm. In [23], authors also propose a hysteresis algorithm to mitigate the problem of a link for which the link margin lies approximately at the midpoint between two decision levels and the ADR algorithm may lead to oscillations between a sub-optimal and an optimal solution for the transmission parameters. Another study worth mentioning is [30], where authors propose that each ED chooses its transmission parameters locally, to minimize the ToA and maximize lifetime. This approach, though, does not take into account the possible connection problem the ED could experience without a proper mechanism to set such parameters according to the conditions the ED is working in. Authors in [39] propose a way to optimize the throughput with respect to the standard implementation by increasing the number of devices using small SFs, even though this leads to lower packet success probability. In [40] two SF allocation mechanisms are proposed: EXP-SF, where SF are equally distributed among nodes (i.e., each cluster of $N/6$ nodes is assigned a specific SF, where N is the total number of nodes), whereas in EXP-AT, SFs are assigned to EDs so as to attempt to achieve the same ToA for each group of (potential) interferers.

[3]ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for research and educational use. ns-3 is free software, licensed under the GNU General Public License, and is publicly available for research, development, and use

In contrast with the above-cited works, CA-ADR takes into account the collision probability at the network level in order to improve the overall packet success probability. This is something other works do not focus on, since their goal is to improve performances in terms of energy consumption and link-level throughput but eventually reduce the network success probability. This new approach, instead, allows achieving fairness among all nodes distributed in the network, whatever the SF they are using, reducing the number of collisions they will experience, which is particularly suitable for large dense networks.

The advantages of deploying LoRaWAN network components on edge and fog computing infrastructures have also been investigated, especially in the case of latency-critical services or limited connectivity toward the Internet. An edge-assisted IoT architecture has been proposed and demonstrated with proof-of-concept implementation, showing how to enable advanced services and distributed storage closer to the end-devices [41]. Another approach consists in augmenting the existing LoRaWAN architecture with a middleware solution that enables IoT data analytics at the edge, allowing application data sharing among multiple customers [42]. To this purpose, a proof-of-concept prototype named IoTRACE has been implemented, which allows multiple data subscribers to deploy their analytics applications at the edge, demonstrating how edge analytics can be combined with cloud analytics. An edge-fog-IoT architecture has also been proposed and applied to the use case of urban traffic management and monitoring [43]. Devices located at the edge and fog layers are used to offload time-sensitive data processing tasks and to provide localized gateway and storage functions. Another work presents the design and deployment of a LoRaWAN infrastructure capable of providing novel applications in a smart campus [44]. The architecture enables the deployment of fog computing nodes throughout the campus to support physically distributed, low-latency, and location-aware applications that decrease the network traffic and the computational load of traditional cloud computing systems. Being in charge of collecting data from the GWs and forwarding them to relevant applications, the NS is typically deployed in a cloud computing environment, which is rarely located close to the source of data. Therefore, in case of latency-critical services or limited connectivity toward the Internet, deployments of LoRaWAN network components based on edge and fog computing solutions have also been considered [45, 46], bringing services, such as advanced analytics and distributed storage, closer to the EDs [41]. However, in most of the previous works the NS functionality is still deployed in the cloud, with processing-enabled GWs located at edge or fog nodes. Although the IoTRACE architecture assumes that NS can be deployed even at the edge, this solution has not been quantitatively evaluated. In general, no previous studies have been carried out to characterize the difference between a cloud-based and a fog-based deployment of a LoRaWAN NS in terms of the performance of the ADR algorithm.

1.4 LoRaWAN Simulator

As stated before, the simulator includes two main components: the PHY-layer and the MAC-layer simulators. The PHY-layer simulator implements a complete LoRa transceiver (transmitter+receiver), thus generating the modulated signal and performing the demodulation tasks, whereas the MAC layer simulator manages the channel multiple access, thus implementing the data traffic (who transmits to whom and when), accounting for mutual interference and the presence of multiple gateways. Specifically, the simulator considers class A LoRaWAN devices (support of class C is also available) operating in the EU 863-870 MHz ISM band and supports a number of features, including both uplink and downlink communications, imperfect SF orthogonality, capture effect, full/half-duplex GW operation, uplink-downlink interference, DC limitations and energy consumption estimation. The simulation tool is extremely flexible, as it allows to tune a large number of parameters, related to the PHY layer, the LoRaWAN protocol and the network itself. The outcomes are provided in terms of the overall Delivery Rate, for both uplink and downlink, and Energy Consumption values for the individual devices and the network as a whole.

In the following, the function of each block shown in Figure 1.3 is discussed.

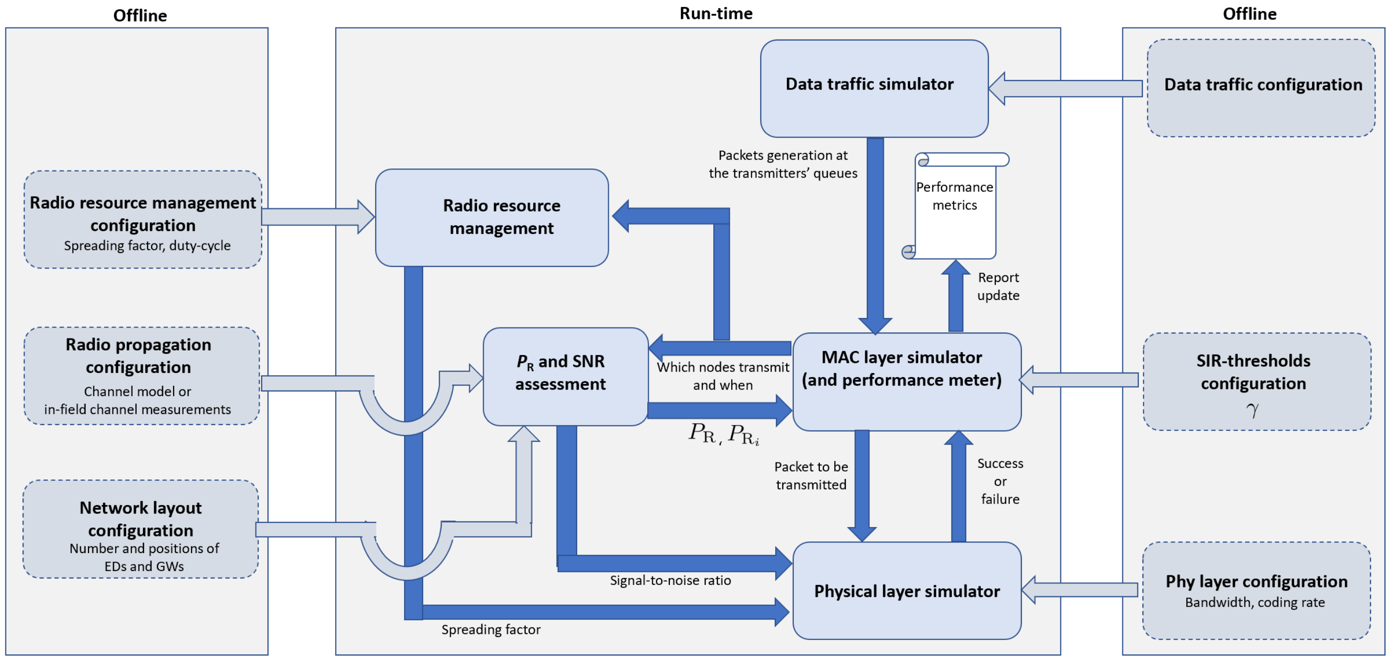

Offline
Run-time
Offline
Data traffic simulator
Data traffic configuration
Packets generation at the transmitters' queues
Performance metrics
Radio resource management configuration
Spreading factor, duty-cycle
Radio resource management
Report update
Radio propagation configuration
Channel model or in-field channel measurements
P_R and SNR assessment
Which nodes transmit and when
MAC layer simulator (and performance meter)
SIR-thresholds configuration
γ
P_R, P_{R_i}
Packet to be transmitted
Success or failure
Network layout configuration
Number and positions of EDs and GWs
Signal-to-noise ratio
Physical layer simulator
Phy layer configuration
Bandwidth, coding rate
Spreading factor

Figure 1.3: LoRaWAN simulator block scheme

1.4.1 Offline configuration operations

Prior to the simulation execution, a configuration step must be carried out offline, which consists of the definition of the scenario (e.g., the network layout) and of a number of parameters, which rules the network behavior. This step is discussed hereafter.

Network layout configuration. For each simulation, an area of interest is defined by the user (by default, a circular shape is assumed), in which N EDs and G GWs are deployed either manually (e.g., where they are actually located in a real deployment), or automatically, in fixed (e.g., on a grid) or in random positions.

Radio propagation configuration. An Okumura-Hata channel model [47] is implemented as the default model in the simulator, both for urban and rural scenarios, even though other channel models can be defined by the user. The power received, P_R, is computed as a function of the transmit power, P_T, as $\mathrm{P_R[dBm] = P_T[dBm] + G_T[dB] + G_R[dB] - L[dB]}$, where G_T is the transmitting antenna gain, G_R is the receiving antenna gain, L represents the path loss in dB, which differs when considering an urban or a rural scenario:

$$\begin{aligned} L_{\text{urban}} =& 69.55 + 26.16 \log_{10}(f) - 13.82 \log_{10}(h_b) - \\ & 3.2(\log_{10}(11.75\,))^2 - 4.97 + \\ & (44.9 - 6.55 \log_{10}(h_b)) \cdot \log_{10}(d) + s, \end{aligned} \tag{1.8}$$

$$\begin{aligned} L_{\text{rural}} =& 69.55 + 26.16 \log_{10}(f) - 13.82 \log_{10}(h_b) - \\ & 0.8 - (1.1 \log_{10}(f) - 0.7)h_m + 1.56 \log_{10}(f) + \\ & (44.9 - 6.55 \log_{10}(h_b)) \cdot \log_{10}(d) - \\ & 4.78(\log_{10}(f))^2 + 18.33 \log_{10}(f) - 40.94 + s, \end{aligned} \tag{1.9}$$

where f [MHz] is the carrier frequency; h_b [m] is the height of the GW antenna above the ground surface; h_m [m] represents the height of the ED above the ground surface; d [km] is the transmitter-receiver distance. Finally, the parameter s represents random channel fluctuations due to shadowing, modeled via a Gaussian random variable, with zero mean and standard deviation σ, that is, $s \sim \mathcal{N}(0, \sigma^2)$. The user can choose the transmit powers P_{T}, σ, and the heights of the EDs/GWs antennas.

Note, that the discussed layout and propagation models can be defined by the user manually based, e.g., on empirical measurements. The respective functionality is supported by the simulator.

Radio resource management configuration. The policy adopted for the choice of the SF can be defined by the user, along with the DC configuration. As for the former, the user can dictate a specific SF for each ED or let the simulator make a decision based on the ADR. With reference to the DC, instead, the DC constraints can be disabled or enabled; in the latter case, these are automatically satisfied according to the regional limitation reported in [19]. In particular, DC limitations are specified according to LoRaWAN specification version 1.0.1, which dictates that a device (i.e., ED or GW), after sending a packet of duration ToA seconds, must not use the same frequency sub-band for the next $\text{ToA}(\frac{1}{\text{DC}} - 1)$ seconds. In addition, the user can also define the transmission parameters related to both RX windows. For RX1, the RX1DROffset parameter, which is the difference between the SF used in uplink and in RX1, is chosen by default according to Table 1.4 (the user can introduce a different table if needed). For RX2, instead, SF is fixed to 12 by the standard, even though the user can decide to change such a parameter.

Data traffic configuration. The traffic generation models, for both the uplink and the downlink, are also configured by the user, along with the size of uplink and downlink packets in bytes (B_{UL} and B_{DL}, respectively), the presence of the packet header H and the length of the preamble L_{preamble}. When the default configuration is adopted, data packets are generated by EDs periodically every T seconds. The user can also define the probability of generating a downlink message after an uplink packet; by default, this is set to 1.

SIR threshold configuration. The **MAC layer simulator** is in charge of reproducing the ALOHA-based access protocol adopted by LoRaWAN, which does not depend on the parameters to be configured. However, within the simulator, this block also evaluates the Signal-to-Interference Ratio (SIR) experienced by a receiver

Uplink SF \ **RX1DROffset**	0	1	2	3	4	5
7	7	8	9	10	11	12
8	8	9	10	11	12	12
9	9	10	11	12	12	12
10	10	11	12	12	12	12
11	11	12	12	12	12	12
12	12	12	12	12	12	12

Table 1.4: Downlink SF according to different RX1DROffset values[19]

due to simultaneous transmissions, in order to check whether the packet is correctly received or not. As detailed in Section 1.4.3, the SIR is compared with a threshold γ, dubbed *SIR-threshold*, which can be defined by the user. Clearly, evaluating the success/failure of transmission in the presence of interference is by no means a MAC layer task; in the simulator perspective, however, the **MAC layer simulator** block is the most appropriate for the SIR assessment, because only this block knows which nodes (if any) are simultaneously transmitting hence might interfere each other.

Physical layer configuration. The bandwidth BW of the modulated signal is a user-defined parameter, along with the coding rate CR of the forward-error-correcting code adopted to reveal/correct transmission errors. Indeed, the encoding process should be carried out at the Logical Link Control (LLC) sub-layer but, for the sake of simplicity, in the simulator, it is carried out by the PHY-layer block. Note, that in the current LoRaWAN specification both BW and CR are pre-specified for each region. Such models allow to update these parameters, thus adapting the simulator to other regions and/or novel modulation-coding schemes.

Table 1.5 summarizes the key parameters a user can define.

1.4.2 Run-time operations

During the execution, the **Data traffic simulator**, which oversees the whole network, provides the **MAC layer simulator** with the time instants when data packets, generated either by EDs or GWs, enter the respective transmission queues. The **MAC layer simulator**, in turn, reproduces the behavior of the ALOHA channel access protocol for all devices in the network, thus deciding which nodes, among those with queued packets, transmit and when.

This information is passed to the **SNR assessment** block that, given the positions of the transmitting nodes and the adopted channel model/channel measurements, derives the SNR experienced by receivers. Such information, along with the current SF value provided by the **Radio resource management**, is used by the **Physical layer simulator**, which generates the modulated signal corrupted by noise and demodulates it in order to assess if the packet is correctly received or not. Clearly, all accompanying operations, such as interleaving/de-interleaving, Gray coding/decoding and channel coding/decoding, are carried out as well. The transmission outcome, either success or failure, is passed to the **MAC layer simulator** for any consequent

Network layout parameters			
N	Number of EDs	G	Number of GWs
R	Circular area radius [km]		
Radio propagation parameters			
P_{T}^{ED}	Transmit power of the ED [dBm]	P_{T}^{GW}	Transmit power of the GW [dBm]
G_{ED}	ED Antenna Gain [dB]	G_{GW}	GW Antenna Gain [dB]
h_m	Height of the ED [m]	h_b	Height of the GW [m]
σ	Shadowing standard deviation [dB]		
Radio resource management parameters			
SF	Spreading Factor	RX1DROffset	Shift between Uplink and RX1 SF
DC	Duty Cycle Limitation		
Data traffic parameters			
B_{UL}	Uplink Payload Size [bytes]	B_{DL}	Downlink Payload Size [bytes]
H	Packet Header presence	L_{preamble}	Length of the preamble
T	Uplink packet periodicity [s]		
SIR threshold parameters			
γ	SIR Threshold [dB]		
Physical layer parameters			
CR	Coding Rate	BW	Sweep interval [kHz]

Table 1.5: List of key parameters defined by the user

action. The **MAC layer simulator** also gets from the **SNR assessment** block the values of the useful and interfering power at the receiver under investigation. This information is used to assess whether the interference level is such to prevent the correct reception. As detailed in Section 1.4.3, this assessment is carried out by comparing the SIR experienced by the receiver under investigation and the SIR-threshold γ. Given the result of such comparison and the success/failure outcomes of the **Physical layer simulator** (which does not account for interference), the **MAC layer simulator** updates the performance counters, which are then used to compute the final performance metrics.

1.4.3 Performance Metrics

At the end of the simulation, three default performance metrics are provided, which are discussed hereafter.

Uplink Delivery Rate

The delivery rate provided by the simulator refers to the packets received by GWs. Two impairments are taken into account that might prevent GWs from correctly receiving transmitted packets, namely noise and interference.

As mentioned above, the LoRa PHY simulator implements the whole transmitter $\rightarrow$ Additive White Gaussian Noise (AWGN) channel $\rightarrow$ receiver chain. In particular, the PHY-layer simulator reproduces the operations carried out by the transmitter (channel coding, interleaving, gray coding and modulation), the addition of AWGN noise in the channel and the receiver behavior (demodulation, deinterleaving, decoding). Thus, given the frame of data bits to be transmitted, the adopted SF, CR and BW as well as the actual SNR that characterizes a given link, the PHY-layer simulator assesses whether the currently transmitted packet is correctly received or not.

The possible presence of interference is also considered, taking into account the possibility that, even though a collision happens between two LoRa packets (considering both uplink and downlink), one of them could be correctly received if one of the signals is strong enough. In this case, the simulator supports the capture effect mechanism, so the packet is correctly received provided that the SIR, that is the ratio between the useful received power and the sum of the interfering powers, is above a given threshold, γ:

$$SIR = \frac{P_{\mathrm{R}}}{\sum_i P_{\mathrm{R}_i}} \geq \gamma \tag{1.10}$$

where P_{R} is the received power of the target ED, P_{R_i} is the received power of the i-th interfering signal and γ depends on the SF used, as specified by a Table 1.6 [21] (the table can also be modified by a user, if desired). In particular, since the simulator accounts for inter-SF interference, the inequality should be verified for all the interfering signals by summing the received power for each SF.

Int / Ref	SF7	SF8	SF9	SF10	SF11	SF12
SF7	1	-8	-9	-9	-9	-9
SF8	-11	1	-11	-12	-13	-13
SF9	-15	-13	1	-13	-14	-15
SF10	-19	-18	-17	1	-17	-18
SF11	-22	-22	-21	-11	1	-20
SF12	-25	-25	-25	-24	-23	1

Table 1.6: SIR Thresholds [dB] [48]

Condition (1.10) is used by the **MAC layer simulator** to decide whether the reception of a packet by GWs is prevented by the interference. In addition, since the simulator supports both full-duplex and half-duplex GWs, interference between transmissions takes into account also the potential uplink/downlink interference and the very possibility of the GW transmitting and receiving packets simultaneously. The type of the GWs is specified as a part of network pre-configuration.

Energy Consumption

The simulator is able to estimate the LoRaWAN energy performance by computing the energy consumed by each ED by assuming an ED working in class A (or class C) and transmitting an uplink packet, followed, optionally, by the reception of a downlink packet. Therefore, the simulator takes into account the energy spent in: uplink transmission, RX Delay 1, downlink reception in RX1, RX Delay 2, downlink reception in RX2, sleep until the next uplink transmission, as reported in (1.11).

$$\begin{aligned} E =& E_{\text{UL}} + E_{\text{RX}_{\text{Delay1}}} + E_{\text{DL}_{\text{RX1}}} + E_{\text{RX}_{\text{Delay2}}} + E_{\text{DL}_{\text{RX2}}} + E_{\text{Sleep}} = \\ & V\ I_{\text{TX}}\ T_{\text{TX}} + V\ I_{\text{RX}_{\text{Delay}}}\ T_{\text{RX}_{\text{Delay1}}} + V\ I_{\text{DL}_{\text{RX1}}}\ T_{\text{DL}_{\text{RX1}}} + \\ & V\ I_{\text{RX}_{\text{Delay}}}\ T_{\text{RX}_{\text{Delay2}}} + V\ I_{\text{DL}_{\text{RX2}}}\ T_{\text{DL}_{\text{RX2}}} + V\ I_{\text{Sleep}} + T_{\text{Sleep}} \end{aligned} \tag{1.11}$$

In (1.11), V represents the voltage, I is the current and T is the time spent in each state. In particular, the time passed in transmission and reception, T_{TX} and T_{RX} (in case a downlink packet has been sent), are considered equal to the ToA of the packet, which is computed from (1.7). The time intervals spent in $T_{\text{RX}_{\text{Delay1}}}$ and $T_{\text{RX}_{\text{Delay2}}}$ are defined by the standard, as reported in Section 1.3. If a downlink packet has not been sent or the packet has not been correctly received due to interference, the time intervals spent in reception during RX1 and RX2 are defined based on Table 1.7, which corresponds to the duration of five preamble symbols, required to effectively detect the downlink packet preamble. The other parameters used to characterize the consumption in the different phases have been taken from [49, 50]; they refer to the Microchip RN2483 LoRa Mote [51], and they are reported in table 1.8.

Table 1.7: RX1 and RX2 duration when no packets are received by the ED [49]

Parameter	**Value**					
SF	7	8	9	10	11	12
T_{RX1} [s]	12.29	24.58	49.14	98.3	131.02	262.14
T_{RX2} [s]	1.28	2.3	4.35	8.45	16.64	33.02

Table 1.8: Energy Consumption parameters [50]

Parameter	**Value**						
P_{T} [dBm]	14	12	10	8	6	4	2
I_{T} [mA]	38	35.1	32.4	30	27.5	24.7	22.3
I_{R} [mA]	38						
$I_{\text{RX}_{\text{Delay}}}$ [mA]	27						
I_{Sleep} [mA]	0.0016						
V [V]	3.3						

Downlink Response Rate

The simulator allows also to estimate the rate of correctly delivered downlink packets. In particular, a downlink packet can be generated for each uplink packet sent by an ED with a user-defined probability. The simulator schedules the downlink packets for each ED based on the number of GWs which have received this packet and their availability (accounting for the DC restrictions). The Downlink Response Rate is then computed as the ratio between the number of downlink packets delivered and those generated.

When modeling the delivery of a downlink, a procedure similar to that discussed above in the case of an uplink is used. However, there are some differences between the RX1 and RX2 cases. According to the LoRaWAN specification, in the case of RX1, data are sent in the same frequency channel used for the uplink, therefore both downlink-to-downlink (i.e. caused by simultaneous downlink transmission from the other GW) and uplink-to-downlink (i.e. caused by simultaneous uplink transmissions of the other ED) are considered. Meanwhile, in the case of RX2, which is typically carried within a standalone frequency channel, only the downlink-to-downlink interference between different GWs is considered.

1.4.4 Validation

To check the accuracy of the simulator results, several tests have been carried out (results are averaged over 10000 iteration cycles), and their results have been mainly checked against that of the analytical models and the state-of-the-art literature, as summarized in this section. Comparisons with experimental results have been also carried out, as discussed in the following.

PHY-Layer Simulator

The behavior of the PHY-layer simulator has been validated through the comparison with the results reported in [52] and [53]. The symbol error rate (i.e., the chirp error rate) as a function of the $\mathrm{SNR} = \frac{P_u}{P_w}$, where

- P_u denotes the average power of the signal,
- $P_w = N_0\mathrm{BW}$ denotes the noise power within the nominal signal bandwidth BW.

is presented in Figure 1.4. Even the visual comparison shows a very close match between the two.

Packet Delivery Rate

The default setting of the parameters adopted in simulations is summarized in Table 1.9. Note, that T there stands for the average period between the uplink packets for each ED. As for the scenario, the GW is considered placed at the center of a circular area of radius R, and the EDs were randomly and uniformly distributed within this area.

Figure 1.5 shows the Uplink Delivery Rate as a function of the Offered Traffic, defined as:

$$O = \frac{8 \cdot B \cdot N}{T} \quad [\mathrm{bit/s}] \tag{1.12}$$

The Uplink Delivery Rate has been defined as the percentage of packets correctly received by the GW according to the conditions described in Section 1.4.3. As a reference, the figure shows the rate computed

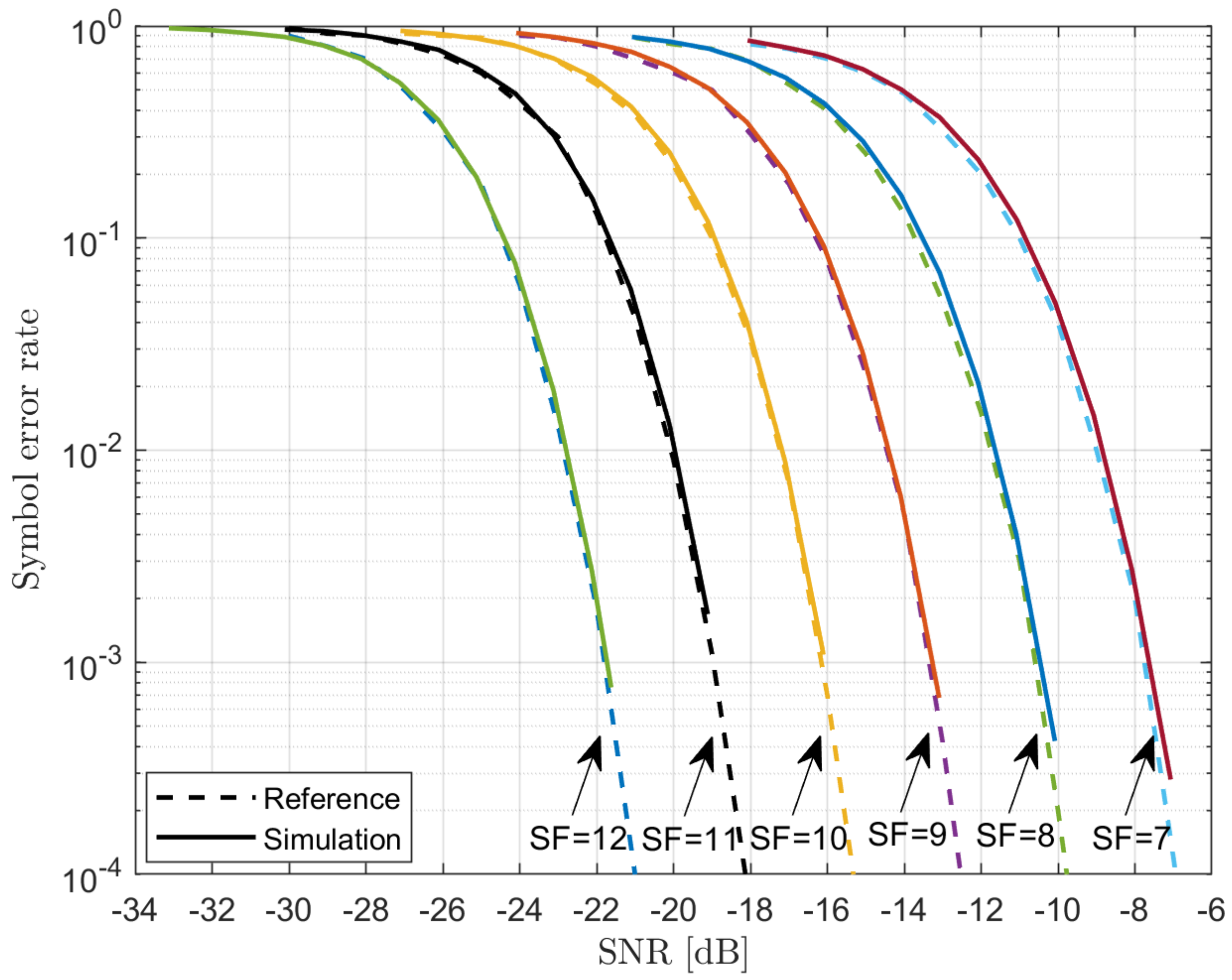


Figure 1.4: Symbol error rate as a function of the SNR

with the well-known equation for ALOHA Success Probability, P_s, by Abramson [54]:

$$P_s = e^{-\frac{2*N*B}{T}} \tag{1.13}$$

It can be seen that the two curves match very closely. Note that, since the analytic equation accounts only for intra-SF interference and it implies the loss of all collided packets, specifically for this comparison, the capture effect and the inter-SF interference computation have been disabled.

Besides validating the simulator by means of analytical models, experiments have been carried out by setting up a small LoRaWAN network. To this purpose, Idesio Rigers Boards 1.0 equipped with the *Microchip RN2483* radio transceiver have been used, fully certified 433/868 MHz SX1276 LoRa module, that supports LoRaWAN Class A, and one GW, placed 50 meters away from the devices. Five boards have been programmed to send a packet of $B_{\text{UL}} = 16$ bytes every $T = 60$ seconds for 1 hour. The Uplink Delivery Rate has been estimated as the ratio between the number of packets received over the number of the sent packet. All devices used the same SF and the experiment has been replicated with SF=7 and SF=10. The results reported

Table 1.9: Simulation Parameters

Parameter	**Value**	**Parameter**	**Value**
N	$\{1, 25, 50, 100, 250, 500\}$	P_{T}^{ED}	14 [dBm]
G	1	P_{T}^{GW}	16 [dBm]
G_{ED}	0 [dB]	G_{GW}	0 [dB]
B_{UL}	20 [bytes]	B_{DL}	20 [bytes]
R	4 [km]	CR	1
T	$\{10, 30, 60, 150, 300\}$ [s]	BW	125 [kHz]
h_m	1 [m]	h_b	30 [m]
H	1	L_{preamble}	8
σ	3 [dB]	RX1DROffset	0
DC	1%		

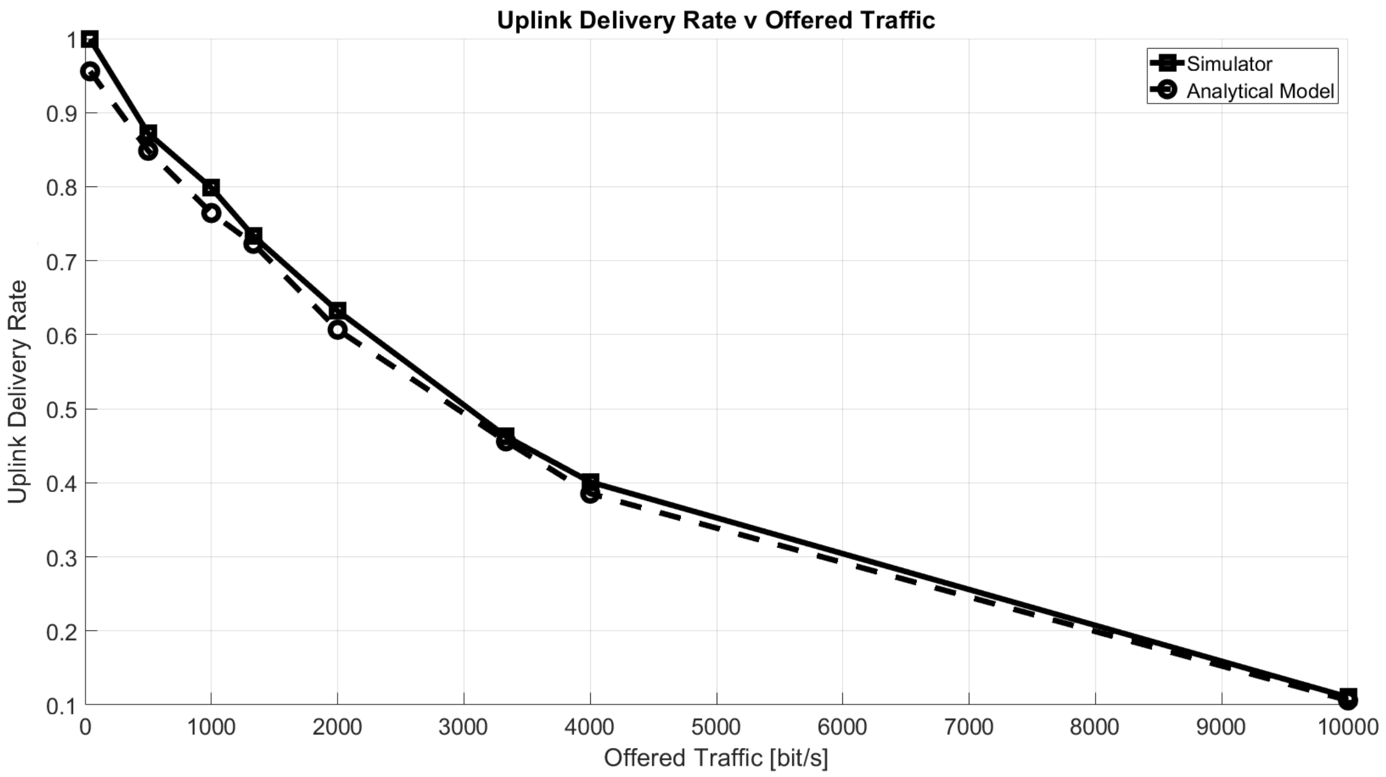


Figure 1.5: Uplink Delivery Rate as a function of Offered Traffic

in Table 1.10 show the good accuracy of the simulator. Note, that the minor difference in the results (i.e., the additional packet losses) may have been caused by interference from the third-party systems.

Table 1.10: Experimental results and comparison with analytical, simulated and experimental results

Results	**SF=7**	**SF=10**
Analytical Result	0.991	0.959
Experimental Result	0.968	0.914
Simulated Result	0.997	0.978

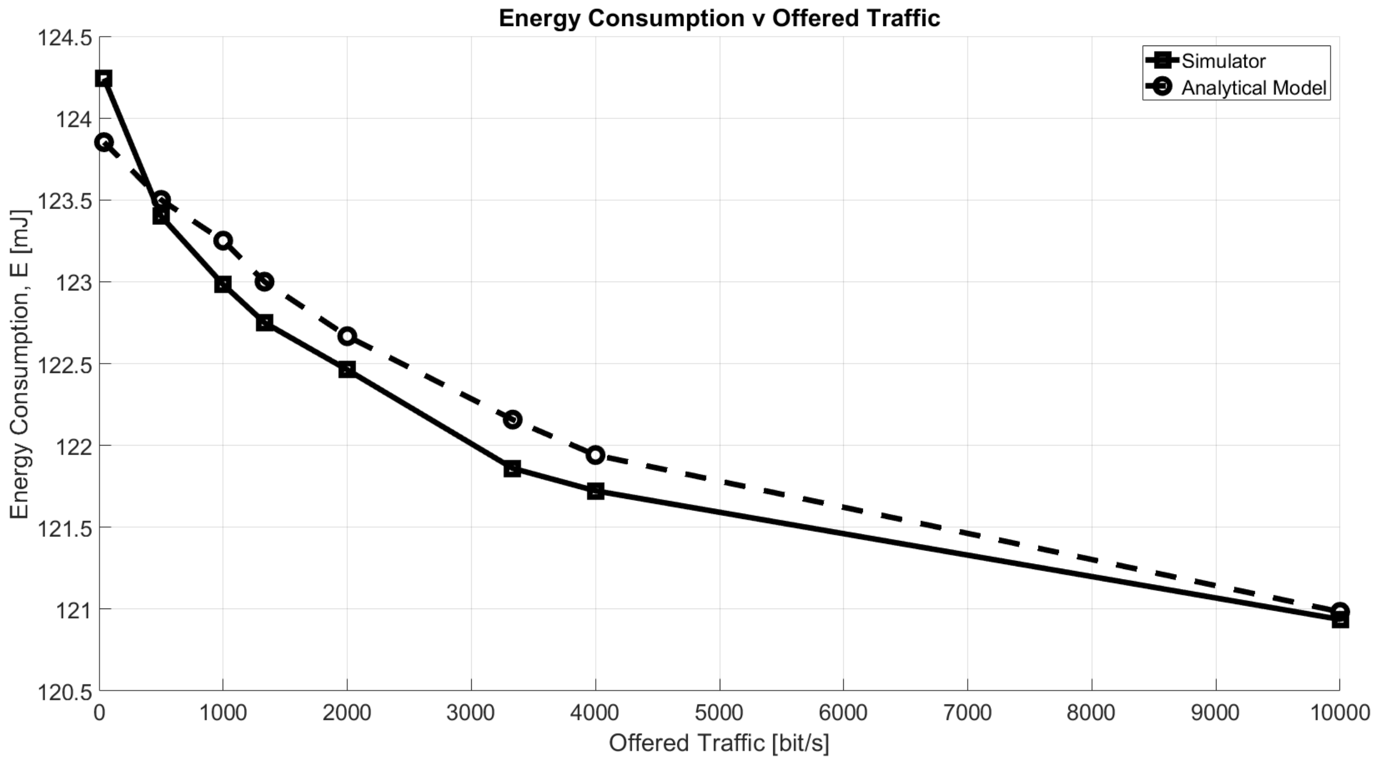


Figure 1.6: Energy Consumption as a function of Offered Traffic

Energy Consumption

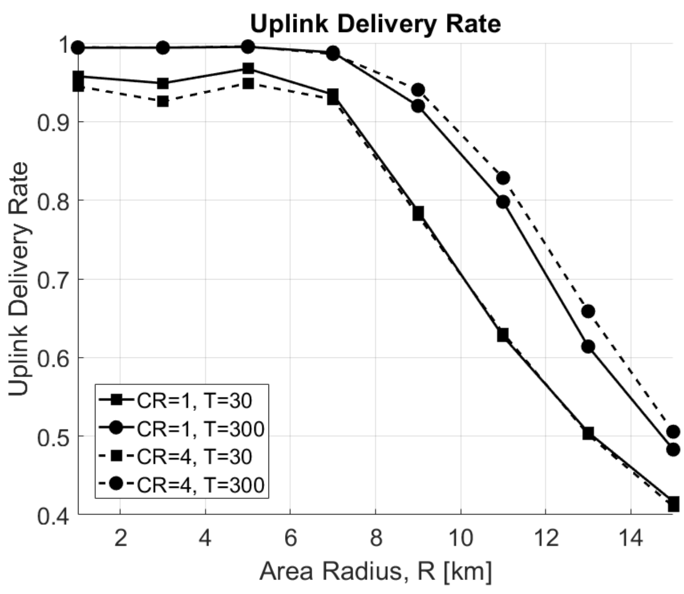
The validation of the energy consumption results provided by the simulator has been carried out by configuring the GW to respond in RX1 to each packet received in uplink and removing the DC restrictions. The results are presented in Figure 1.6. For comparison, a simple analytical model given by

$$E = E_{\mathrm{UL}} + E_{\mathrm{RX_{Delay1}}} + E_{\mathrm{DL_{full}}} \cdot P_s + E_{\mathrm{DL_{empty}}} \cdot (1 - P_s) \quad [\mathrm{J}] \tag{1.14}$$

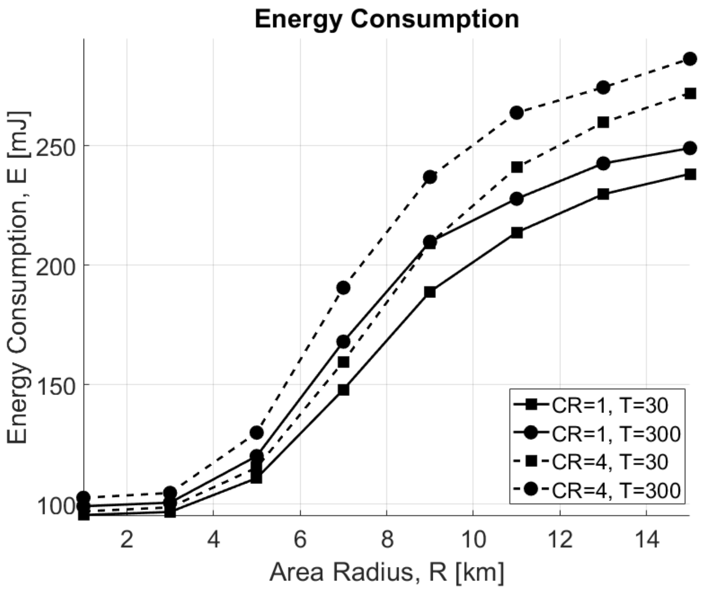
has been used, where E_{UL} is the energy required for the uplink transmission, $E_{\mathrm{RX_{Delay1}}}$ is the energy consumed between the uplink and RX1 (having duration of 1 s), while, in downlink, $E_{\mathrm{DL_{full}}}$ is the energy spent if a packet is received, and, finally, $E_{\mathrm{DL_{empty}}}$ denotes the consumption of a device when it opens the RX Window, but no packet is received. The NS will send a packet in downlink only if it receives an uplink packet, whose success probability is equal to P_s, so it knows the device has transmitted something and it will open the Receive Window. It should be highlighted that in this case, neither duty cycle constraints nor the possibility the send a downlink message in RX2, which would require a more complex model, are taken into account. As can be seen from the presented charts, the results are almost coincident.

1.4.5 Numerical Results

Having validated the simulator in simple scenarios, some initial insight into less trivial cases, which show some notable trade-offs, are provided in the following. Unless stated otherwise, for these simulations, the same parameters and configurations which have been used in the previous section are kept. In all the considered scenarios, GWs operate in full-duplex mode.



(a) Uplink Delivery Rate as a function of Area Radius, R.



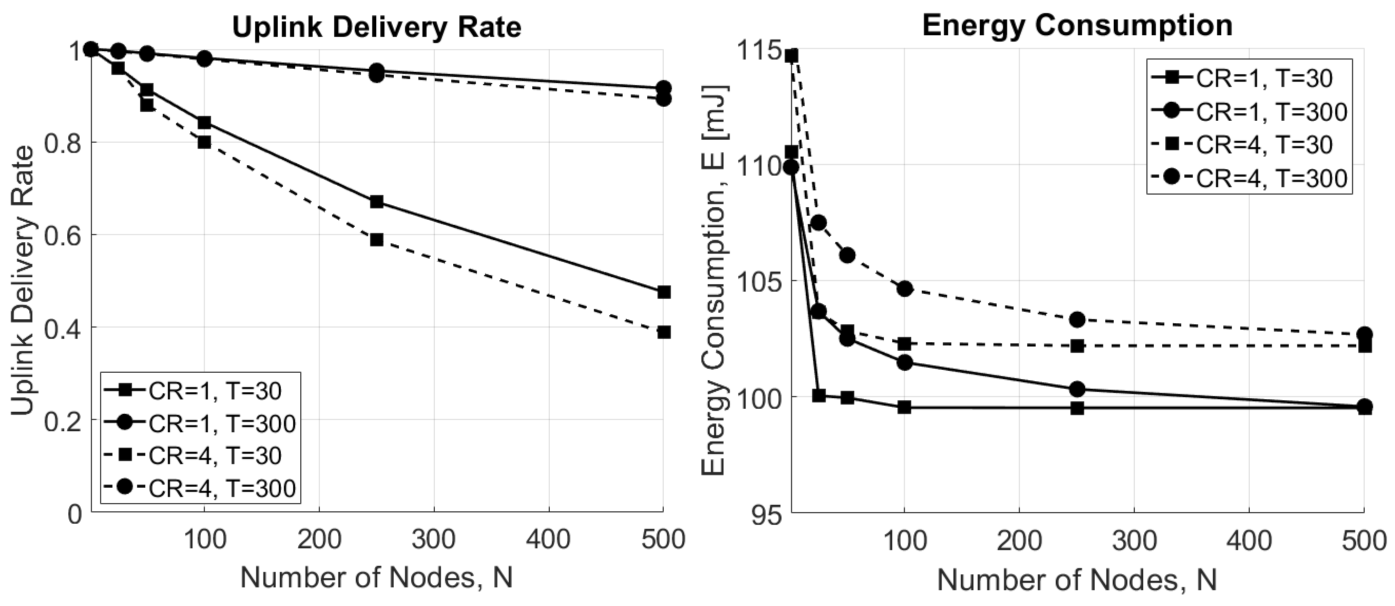
(b) Energy Consumption as a function of Area Radius, R.

Figure 1.7: Impact of the Area Radius, R [km].

Figure 1.7a depicts the Uplink Delivery Rate as a function of the radius of the circular area where $N = 50$ nodes are distributed for both CR=1 and CR=4 when varying T. As expected, when the area dimension increases, the delivery rate decreases. In particular, in an interference-limited scenario ($T = 30$ s), performance does not depend on CR, but for very small areas, CR=1 is the best case due to lower ToA and, therefore, collisions. On the other hand, when the scenario is limited by noise ($T = 300$ s), CR=4 performs better for bigger areas, thanks to the stronger encoding it is able to offer.

Observing the energy performance in this scenario, reported in Figure 1.7b, it can be noticed that the consumption increases with the area size. This happens because EDs are forced to use higher SFs when they are far from the GW, which causes higher energy consumption. In addition, nodes using CR=4 consume more energy mainly because the ToA is higher, therefore the time spent during transmission is higher.

Figure 1.8a shows the delivery rate as a function of the number of EDs in the network for both CR=1 and CR=4. As a matter of fact, the higher the number of EDs in the network, the higher the rate of collisions, which results in the degradation of the overall performance of the network. In addition, when the scenario is



(a) Uplink Delivery Rate as a function of Number of EDs, N

(b) Average Energy Consumption as a function of Number of EDs, N.

Figure 1.8: Impact of the Number of End Devices, N

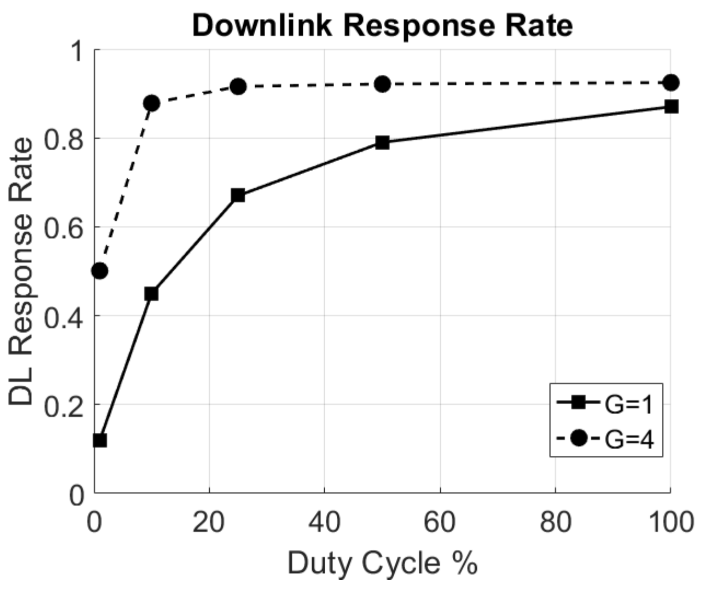
limited by interference ($T = 30$ s), CR=1 shows better performance. This happens because the ToA is smaller w.r.t CR=4 case, therefore there are fewer collisions. On the other hand, when the scenario is limited by noise ($T = 300$ s) and all EDs can successfully reach the GW since no interference is present, no difference between the two approaches is highlighted. The respective energy consumption is revealed in 1.8b, which shows that the average energy consumption decreases when the number of ED increases. This result may seem counter-intuitive; however, it is explained by the increase of the uplink collisions, resulting in fewer packets being correctly received by the GW, and, consequently, fewer packets sent in downlink to the ED. As a matter of fact, if the uplink message is lost, no downlink packet will be sent to the ED, which will then consume less power. In addition, also in this case, nodes using CR=4 consume more energy on average.

For the sake of completeness, an analysis of the Uplink Delivery Rate in presence of class C downlink traffic has been carried out, whose functioning has been described in Section 1.2.2. In this scenario, $N = 50$ EDs transmit every $T = 60$ seconds. 2 GWs have been placed at $(x, y) = [R/2, R/2], [-R/2, -R/2]$ of a circular area of radius R. More specifically, it is assumed that one GW may be transmitting a class C downlink packet with probability p at an instant $t \in [0, T]$ and, therefore, it may be interfering with the uplink packet sent to the other GW, according to the capture effect mechanism already described in Section 1.4.3. As expected, the higher the probability p, the higher the interference, so the Uplink Delivery Rate decreases. Results are provided in Table 1.11.

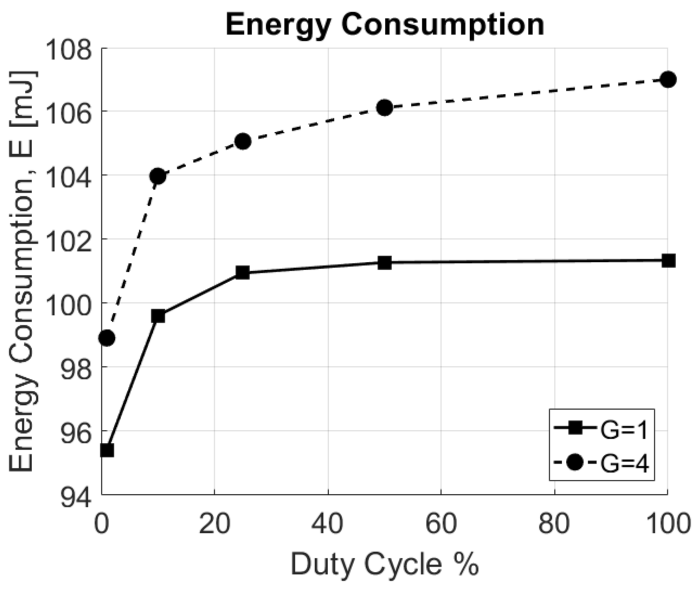
Finally, an analysis of the Downlink Response Rate, as described in section 1.4.3, has been carried out. Two scenarios have been considered: one with a single GW, placed at the centre of the circular area of radius R, and one with four GWs, placed respectively at coordinates $(x, y) = [R/2, R/2]$, $[-R/2, -R/2]$,

Table 1.11: Uplink Delivery Rate with Class C Downlink Interference

p	Uplink Delivery Rate
0	0.9748
0.1	0.9662
0.3	0.9428
0.5	0.894



(a) Downlink Response Rate



(b) Average Energy Consumption

Figure 1.9: Impact of the Duty Cycle, DC

$[R/2, -R/2]$, $[-R/2, R/2]$, with RX2 disabled. In Figure 1.9a the Downlink Response Rate as a function of the duty cycle for RX1 window is reported. The higher the DC, the higher the number of packets sent in the downlink. Per intuition, with more GWs present in the network, the Downlink Response Rate is higher. However, this results in increased average energy consumption, since the reception of a downlink packet often requires more energy than that needed for checking the two empty receive windows.

1.5 Collision-Aware ADR Algorithm

1.5.1 CA-ADR Description

In this section, a new ADR algorithm will be introduced and evaluated. The idea behind the new algorithm is to assign SF in order to guarantee a given success probability in delivering a packet, accounting not only for the link-level performance (connectivity with the GW), but also for the collision probability (MAC-level performance). The algorithm exploits the orthogonality among signals transmitted with different SFs, as assumed in previous work (see, e.g., [21, 48]).

Algorithm 3: Collision-Aware ADR Algorithm

```
   Input: $\hat{p}_{\text{mac}} = 1, \mathcal{SF} = \{7, 8, 9, 10, 11, 12\}$
   Output: SF(n) for $n \in \mathcal{N}$
1  foreach $\text{SF} \in \mathcal{SF}$ do
2      $n_{\text{count}}(\text{SF}) = 0$
3      $n_{\text{max}}(\text{SF}) = \left\lfloor \left( \frac{1}{2} \log_{(1-\frac{\text{ToA(SF)}}{T})} (\hat{p}_{\text{mac}}) \right) + 1 \right\rfloor$
4  foreach $n \in \mathcal{N}$ do
5      $\text{SF}_{\text{min}}(n) = min\{\text{SF}\, s.t.\, P_{\text{R}}(n) \geq P_{\text{R}_{\text{min}}}(\text{SF})\}$
6      $\text{SF}_{\text{temp}}(n) = \text{SF}_{\text{min}}(n)$
7  foreach $n \in \mathcal{N}$ do
8      if $n_{\text{count}}(\text{SF}_{\text{temp}}(n)) < n_{\text{max}}(\text{SF}_{\text{temp}}(n))$ then
9          $\text{SF}_{\text{res}}(n) = \text{SF}_{\text{temp}}(n)$
10         $n_{\text{count}}(\text{SF}_{\text{temp}}(n)) = n_{\text{count}}(\text{SF}_{\text{temp}}(n)) + 1$
11     else
12         $\text{SF}_{\text{temp}}(n) = \text{SF}_{\text{temp}}(n) + 1$
13         if $\text{SF}_{\text{temp}}(n) \leq 12$ then
14             go to 8
15 if $\sum_{\text{SF} \in \mathcal{SF}} (n_{\text{count}}(\text{SF})) = N$ then
16     return $\text{SF}(n) = \text{SF}_{\text{res}}(n)$ for $n \in \mathcal{N}$
17 else
18     if $\hat{p}_{\text{mac}} > 0.01$ then
19         $\hat{p}_{\text{mac}} = \hat{p}_{\text{mac}} - 0.01$
20         go to 1
```

In the algorithm $p_{\text{mac}}(\text{SF})$ denotes the packet success probability for an ED using spreading factor SF, taking into account collisions at the MAC layer. In particular, $p_{\text{mac}}(\text{SF})$ is the probability that no other transmissions will occur during the vulnerability period of ALOHA, which coincides with $2 \cdot \text{ToA}$, being ToA given by equation (1.7). Therefore, $p_{\text{mac}}(\text{SF})$ is defined as [54]:

$$p_{\text{mac}}(SF) = \left(1 - \frac{ToA(SF)}{T}\right)^{2(n(SF)-1)} \tag{1.15}$$

where $n(\text{SF})$ is the number of devices using spreading factor SF. In the algorithm, for the sake of simplicity, the capture effect is not accounted; therefore, it tries to assign SFs in order to limit as much as possible the collision probability. When running the simulation, after the SF values assignment, the simulator then accounts for the capture effect, as described in section 1.5.2, to evaluate if a packet is correctly received by the GW.

The CA-ADR algorithm is reported in Algorithm 3 and it works as follows. By first setting $p_{\text{mac}}(\text{SF})$ equal to a target value, $\hat{p}_{\text{mac}} = 1$, the maximum number of EDs that can use a specific SF, $n_{\text{max}}(\text{SF})$, is computed according to the equation reported in line 3 of the algorithm (i.e., reversing equation (1.15)). Then, for each node n the minimum value of SF the ED can use is derived, that is the minimum SF satisfying $P_{\text{R}}(n) \geq P_{\text{R}_{\text{min}}}(\text{SF})$ (see line 5 of algorithm 3). $P_{\text{R}}(n)$ is the average power received by the GW when the ED is transmitting (averaged over the last k packets received) and $P_{\text{R}_{\text{min}}}(\text{SF})$ is reported in Table 1.12. The algorithm then tries to assign the obtained minimum SF to each ED, as long as the number of nodes already assigned that SF value ($n_{\text{count}}(\text{SF})$) does not exceed the maximum allowed ($n_{\text{max}}(\text{SF})$). If the limit is reached, then the algorithm tries to assign a higher SF following the same process and keeps increasing it up to the maximum value (SF=12). When $n_{\text{count}}(\text{SF}) = n_{\text{max}}(\text{SF})$ even for the highest SF value, no SF assignment is carried out for that specific ED. At the end of the process, the algorithm checks if it was able to assign a SF to each ED, by comparing the sum of the $n_{\text{count}}(\text{SF})$ to N. If they are equal, it means that each ED has a correct SF assigned, otherwise $\hat{p}_{\text{mac}}$ is decremented by 0.01 and the algorithm runs again from the beginning, as long as $\hat{p}_{\text{mac}} > 0.01$.

Differently from the reference ADR implementation, the CA-ADR algorithm running on the NS goes through a number of iterations to make sure that all EDs are assigned the best SF values such that $n_{\text{max}}(\text{SF})$ is never exceeded. It is thus important to assess the complexity of the CA-ADR algorithm. In the worst case when all the iterations must be executed, the number of operations to be performed is given by $(|\text{SF}| + N \cdot |\text{SF}| + N + |\text{SF}|) \cdot 100$. Considering that the set SF is very small, the CA-ADR complexity is $\mathcal{O}(N)$.

Table 1.12: SF sets for different $P_{\text{R}_{\text{min}}}$ with BW=125 kHz.

$P_{\text{R}_{\text{min}}}$[dBm]	SF_{min}	$\mathcal{SF}$
-137	12	$\{12\}$
-134.5	11	$\{11, 12\}$
-132	10	$\{10, 11, 12\}$
-129.5	9	$\{9, 10, 11, 12\}$
-127	8	$\{8, 9, 10, 11, 12\}$
-124.5	7	$\{7, 8, 9, 10, 11, 12\}$

1.5.2 Benchmarking the CA-ADR algorithm

In this section assumptions and models according to the scenario considered to evaluate the new CA-ADR are reported. A set $\mathcal{N}$ of EDs is considered, whose size is $N = |\mathcal{N}|$, randomly and uniformly distributed in a square area of side D [m]. A set $\mathcal{G}$ of GWs are deployed in the area, with $G = |\mathcal{G}|$. EDs generate a packet of payload B [bytes] in an instant that is randomly and uniformly distributed within a time period T [s]. As a result, the offered throughput, O, defined as the number of bits per second generated in a network of N EDs, is given by:

$$O = \frac{8 \cdot B \cdot N}{T} \quad [\mathrm{bit/s}] \tag{1.16}$$

Note that duty cycle constraints are not explicitly assumed, but the results shown are valid even in the presence of duty cycle, provided that the offered throughput O given by equation (1.16) is generated each time an ED wakes up to transmit its data.

The power received by a GW, P_{R}, as a function of the transmit power, P_{T} is computed according to the Okumura-Hata model[47] described in Section 1.4 for large urban areas. In addition, both uncorrelated and correlated shadowing samples are considered in the following. For the correlated case, the well-known Gilbert-Elliot model [55, 56] is considered, since it gives a negative exponential correlation function, which is exactly the correlation function that can be observed for shadowing in outdoor environments [56]. According to [55] the channel is modeled via a two-state Markov chain (see Figure 1 in [55]), where the two states represent: *Good* channel, where shadowing is characterized by a low value of the standard deviation, σ_{G} and *Bad* channel, where shadowing is characterized by a high value of the standard deviation, σ_{B}. The probability of remaining in a given state (*Good* or *Bad*) is set equal to p and performance are analyzed by varying p.

At the receiver side, a packet is assumed correctly received if:

1. $P_{\mathrm{R}} \geq P_{\mathrm{R_{min}}}(SF)$ and
2. $P_{\mathrm{R}}/\sum_i P_{\mathrm{R}_i} \geq \gamma$

where P_{R} is the useful received power, that is the power received by the GW when the ED is transmitting (in case of multiple GWs, the strongest one, that is the one receiving the largest received power, is considered); $P_{\mathrm{R_{min}}}$ is the receiver sensitivity; $P_{\mathrm{R_i}}$ is the power the GW is receiving from the $i - th$ interfering ED.

The second condition represents the capture effect: even in the presence of collisions, there is a possibility that the receiver (i.e., the GW) is able to capture the frame if the SIR is above a given threshold (capture threshold) [57]. γ has been measured in [21] via experiments performed with the same devices as the ones used in this chapter (see Table III of [21] for details). Finally, note that, as stated above, LoRaWAN uses a simple ALOHA-based multiple access protocol, meaning that an ED sends its data whenever available and interferers will be represented by all the other transmitters in the area whose packets are partially or entirely overlapping in time with the useful one.

According to specifications [26], the receiver sensitivity, $P_{\mathrm{R_{min}}}$, is defined as:

$$P_{\mathrm{R_{min}}}[dBm] = 10\, log_{10}(k \cdot BW \cdot T_0(F-1)) + SNR_{\mathrm{min}} \tag{1.17}$$

where the first term represents the noise power, given that k is the Boltzmann constant, BW is the bandwidth, $T_0 = 290$ Kelvin degrees is the standard noise temperature, $F = 6$ dB is the receiver noise figure; $\mathrm{SNR_{min}}$ is reported in Table 1.3 and it represents the minimum SNR required for demodulation [27].

1.5.3 Numerical Results

The performance of the CA-ADR algorithm are evaluated in terms of i) Packet Success Rate, P_{s}; ii) Network Throughput, S; iii) Latency, L, and iv) Round Trip Time, RTT.

P_s is the percentage of packets that are correctly received at the GW, given that both conditions 1 and 2 in Section 1.5.2 are satisfied. Consequently, the Network Throughput, S, has been defined as:

$$S = \frac{B \cdot N \cdot P_s}{T} \quad [\text{bit/s}] \tag{1.18}$$

Latency L has been defined as the interval of time between the generation of a packet at the ED and its reception at the NS and it can be expressed as

$$L = T_{\text{ED}}^{(proc)} + \text{ToA}_{\text{UL}}(\text{SF}) + T_{\text{GW}}^{(proc)} + \tau_{GW-NS} + T_{\text{NS}}^{(proc)} \quad [\text{s}] \tag{1.19}$$

where $T_{\text{ED}}^{(proc)}$ is the time needed by an ED to generate a packet, $\text{ToA}_{\text{UL}}(\text{SF})$ is the ToA for transmitting the uplink data for a given value of SF, $T_{\text{GW}}^{(proc)}$ is the processing time needed by the GW (which just performs packet forwarding), τ_{GW-NS} is the propagation delay needed to reach the NS from the GW (via Internet), and $T_{\text{NS}}^{(proc)}$ is the processing time required at the network server.

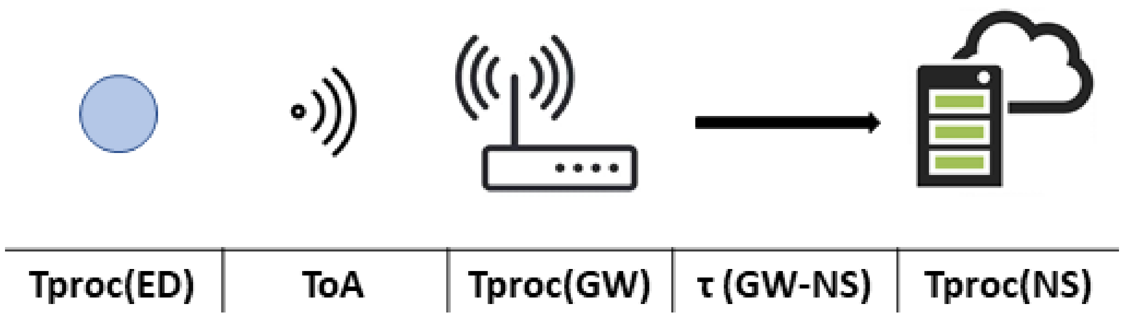


Figure 1.10: Latency

Finally, RTT has been defined as the interval of time between the generation of a query at the NS, to be sent in downlink to a given ED, and the instant when the NS receives the reply in uplink from the ED itself. This delay strongly depends on the operating class used by the ED. In particular, since in Class A the receive window is opened only after an uplink message, the NS is able to send a downlink message (which contains the query) only after the correct transmission by the ED. This means that if an uplink packet is lost, the NS will not know that the receive window of the ED is open and it will not send the downlink message, waiting for the next uplink packet. Therefore, in the case of Class A devices, the packet to be sent in downlink remains at the NS for a certain amount of time, denoted as $T_{\text{NS}}^{(wait)}$, which depends on the periodicity with which packets are generated and on the probability that they are successfully received at the GW/NS. Therefore, RTT for Class A devices, RTT_{A}, is given by:

$$RTT_{\text{A}} = T_{\text{NS}}^{(wait)} + \text{ToA}_{\text{UL}}(\text{SF}) + T_{\text{ED}}^{(RXwind)} + \text{ToA}_{\text{DL}}(\text{SF}) + L \quad [\text{s}] \tag{1.20}$$

where $T_{\text{NS}}^{(wait)} = \frac{T}{2} + T \cdot (1 - P_s)$ is the average waiting time of the packet at the NS, given that EDs generate packets in uplink every T and these packets have a probability to be correctly received P_s, and $T_{\text{ED}}^{(RXwind)}$ is the interval between the uplink transmission and the beginning of the first receive window opened by the ED, denoted in the standard as RECEIVE_DELAY1.

In Class C, instead, as described in Section 1.2.2, the NS can send a packet to the ED at almost every instant (except during ED transmissions and taking into account possible duty cycle limitation), therefore, there is no need to wait for an uplink message from the ED. The RTT for Class C devices, RTT_{C}, is given by:

$$RTT_{\mathrm{C}} = \tau_{NS-GW} + \mathrm{ToA}_{\mathrm{DL}}(\mathrm{SF}) + L \quad [\mathrm{s}] \tag{1.21}$$

CA-ADR algorithm has been tested and compared with the standard solution, considering an integrated approach exploiting simulations and experiments.

As far as the simulation is concerned, a proprietary MATLAB® simulator, implementing the models presented in the previous section and the standard ADR algorithm, has been developed. At each transmission, after the SF assignment, the simulator checks if collisions among packets sent with the same SF happen and if they result in a loss (see condition 2 in Section 1.5.2). This allows computing P_{s} and S.

The considered scenario is shown in Figure 1.11, where two GWs are deployed in a fixed position and EDs are randomly distributed in a squared area. The parameters used in the simulation are reported in Table 1.13. If not otherwise specified, the case of uncorrelated shadowing with standard deviation σ is considered.

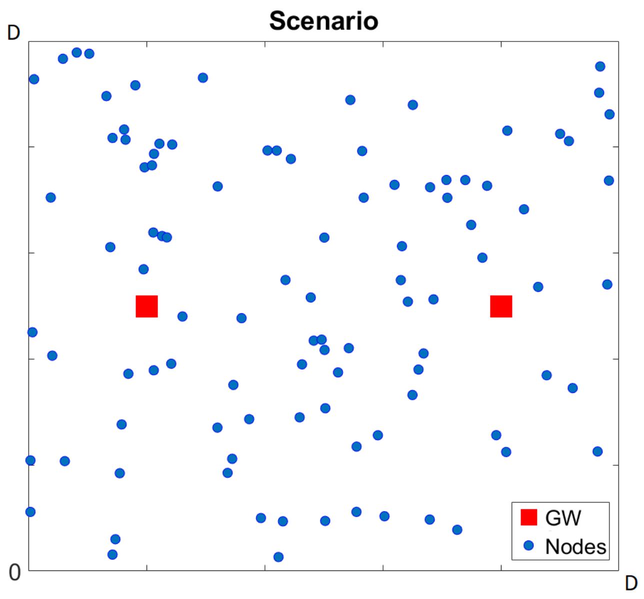


Figure 1.11: Reference Scenario

In order to understand the limitations of the new algorithm in terms of the number of devices it is able to handle, the Packet Success Rate and the Network Throughput are plotted as a function of N. In Figure 1.12, the Packet Success Rate as a function of the number of the EDs in the network is presented. As expected, the success probability decreases with N due to the increase in the collision probability. CA-ADR algorithm performs notably better because it takes into account the overall distribution of EDs in the network, assigning

Parameter	Notation	Value	Parameter	Notation	Value
Carrier frequency [MHz]	f	$868, 5$	Bandwidth [kHz]	BW	125
Coding rate	CR	4/5	Transmit power [dBm]	P_{T}	14
Number of EDs	N	100	Number of GWs	G	2
Packet periodicity [s]	T	5	Area side [km]	D	3
SIR Threshold [dB]	γ	1.5	Payload size [bytes]	B	16
Propagation constant	β	3	Shadowing std. dev. [dB]	σ	3
Maximum P_{T} [dBm]	$P_{\mathrm{T_{max}}}$	14	Minimum P_{T} [dBm]	$P_{\mathrm{T_{min}}}$	2
Preamble length	L_{preamble}	8	Header	H	1
GW antenna height [m]	h_b	30	Shadowing std. dev. (bad) [dB]	σ_{B}	6
ED antenna height	h_m	1 m	Shadowing std. dev. (good) [dB]	σ_{G}	1

Table 1.13: CA-ADR Simulation Parameters

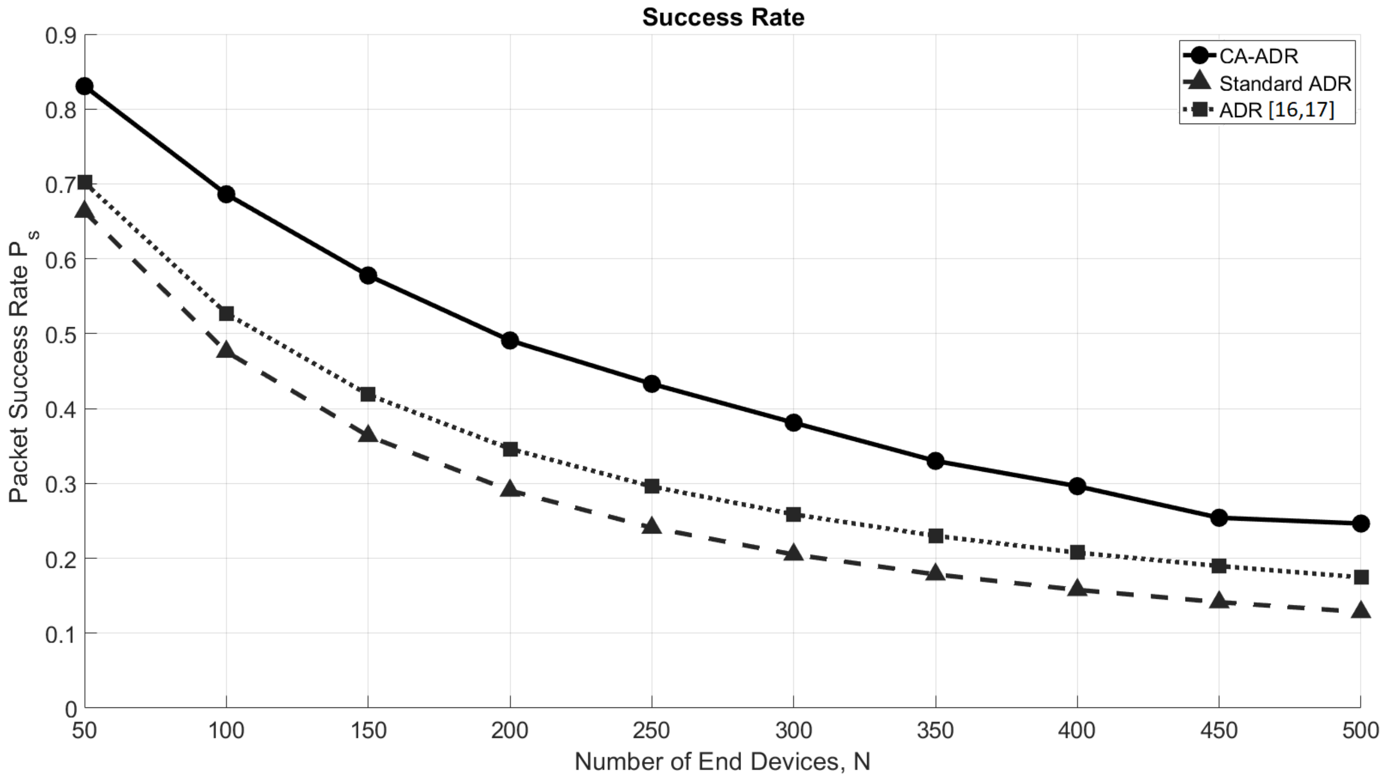


Figure 1.12: Packet Success Rate, P_{s}, as a function of the number of EDs, N.

SFs in a more fair way. Collisions are not taken into account by the standard algorithm, which assigns the lowest SF possible to all devices by looking only at the link-level performance. The new algorithm, instead, reduces as much as possible the set of nodes using the same spreading factor, by keeping track of the number of EDs already using a given SF and assigning (when possible from the connectivity viewpoint) a different value of SF, reducing the collision probability. In the figure results obtained with the ADR algorithm proposed in [22, 23] are considered as another benchmark. As expected, it performs better than the standard one, since it assigns SFs based on the measure of the average SNR, rather than the maximum one. However, the CA-ADR solution outperforms also this benchmark, since it uses average SNR measure as done in [22, 23] and, in addition, includes features to reduce collisions, not considered in [22, 23].

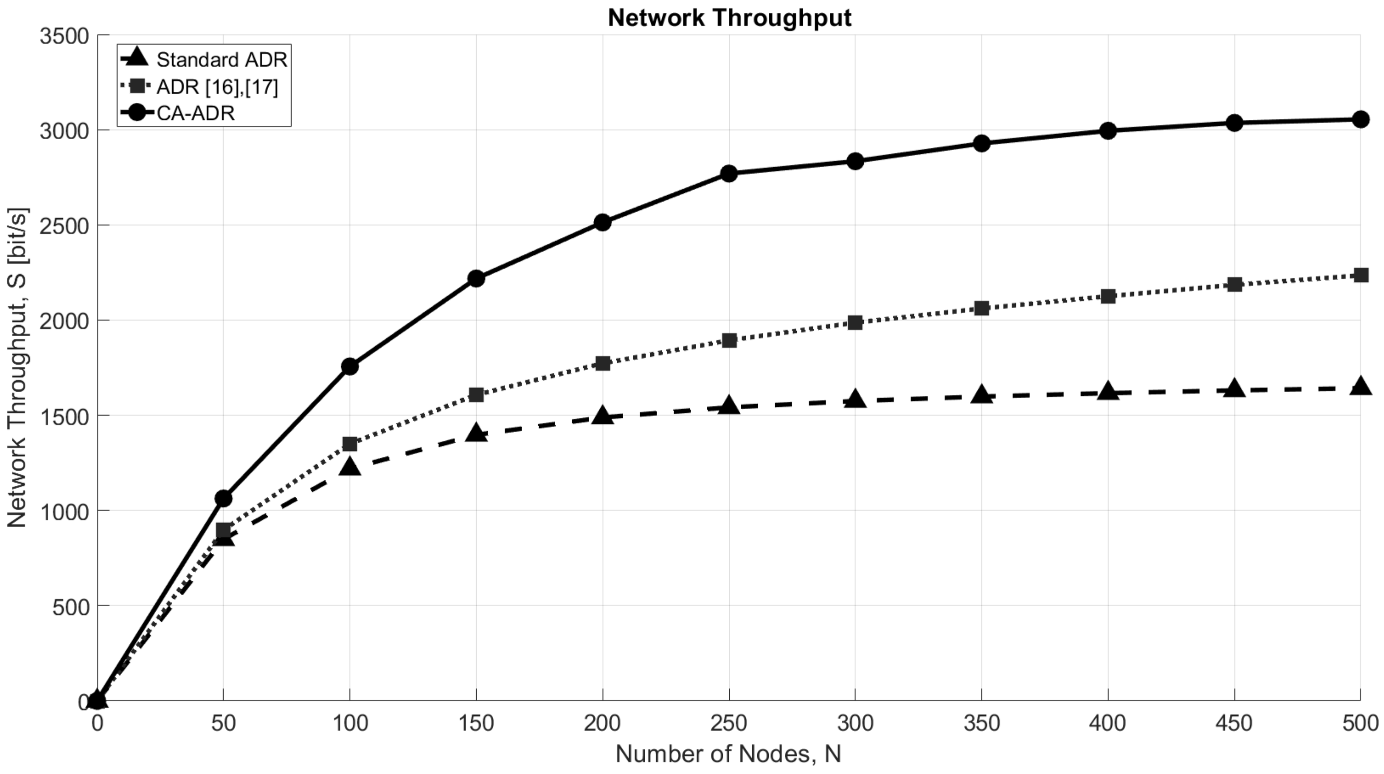


Figure 1.13: Network Throughput, S, as a function of the number of EDs, N.

Figure 1.13 shows the Network Throughput as a function of the number of EDs. Performance, in this case, is a direct consequence of the behavior of the Packet Success Rate. Indeed, for low values of N, S increases with N, being proportional to it (see equation (1.18)) and being P_s high; on the contrary, when N becomes too large, P_s starts decreasing dramatically, resulting in a saturation effect for S. Therefore, the improvement of the proposed solution w.r.t. the standard one increases by increasing N.

Figure 1.14 shows the behavior of the Network Throughput by varying the area size. The new algorithm performs better w.r.t. the standard solution until a given area size is reached, after which they converge. S for the CA-ADR case decreases with D, since this algorithm is only limited by connectivity problems, so its performance is maximized for small areas. With increasing values of D, the algorithm needs to assign SF=12 to more and more nodes in order to reach the GW, resulting in an increased number of collisions. On the contrary, the standard solution presents an optimum value of D maximizing S: for small area sizes, it tends to assign SF=7 to all nodes, which is enough to reach the GW, and this results in many collisions (many nodes using the same SF), while for large areas, as in the case of CA-ADR, the algorithm tends to assign SF=12 to all nodes to overcome connectivity problems, resulting again in many collisions (this is the reason why the two algorithms converge for $D > 8000$ m). The peak value of S for the standard solution is reached for $D = 6000$

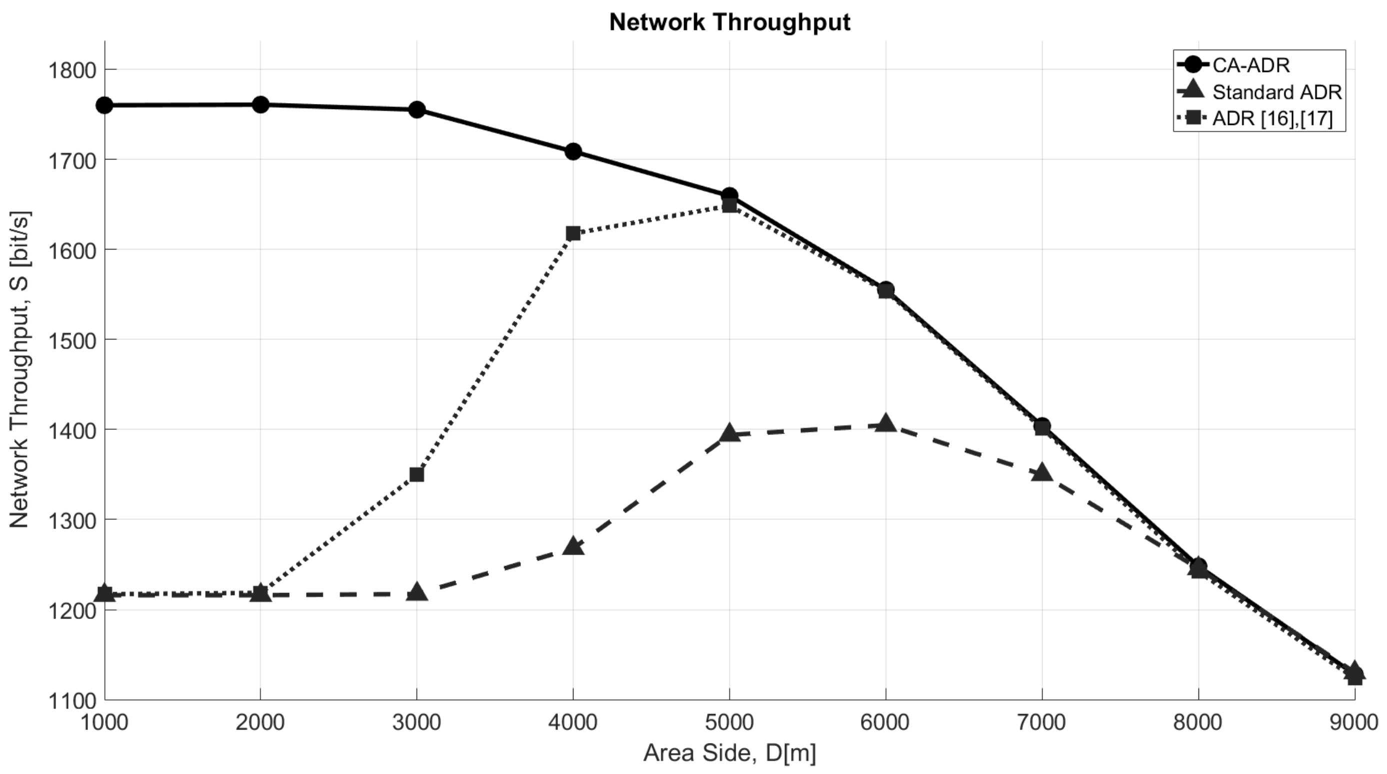


Figure 1.14: Network Throughput, S, as a function of the area side, D [m].

m, where some EDs use a higher SF to maintain connectivity, so the overall number of nodes using the same SF is lower with respect to smaller areas, resulting in fewer collisions. The solution in [22, 23] shows some improvements, because it takes into account the average of the SNR values, so it is more conservative and it assigns a low SF value with less probability w.r.t. the standard. However, CA-ADR still performs better for areas with side $D < 5000$ m.

Figure 1.15 shows the impact of the shadowing on the performance of the algorithms. The inclusion of shadowing strongly reduces performance for both solutions, standard and CA. However, performance does not change significantly when considering different channel models in the presence of shadowing, that is for the uncorrelated or correlated channels by varying p (i.e., the probability of remaining in *Good* or *Bad* channel conditions). However, it is important to notice that, in all cases, the proposed solution is always better than the standard one, because the assignment of the SF is always carried out in order to reduce collisions, when possible.

Figure 1.16 shows the Packet Success Rate as a function of T. Since the CA-ADR algorithm is specifically designed to reduce collisions, the lower T, the higher will be the offered throughput, and the larger will be the gap w.r.t. the standard because collisions are better managed by CA-ADR. In addition, the figure shows the impact of the presence of external interference. A situation where $N_{\mathrm{i}} = 50$ interfering nodes are randomly and uniformly deployed in the area has been simulated; they generate a data packet, assumed to be transmitted with the same transmit power used by LoRa devices and assuming that the packet occupies the channel for 100 ms. Results have been obtained by modifying the frequency of generation of packets from the interferers,

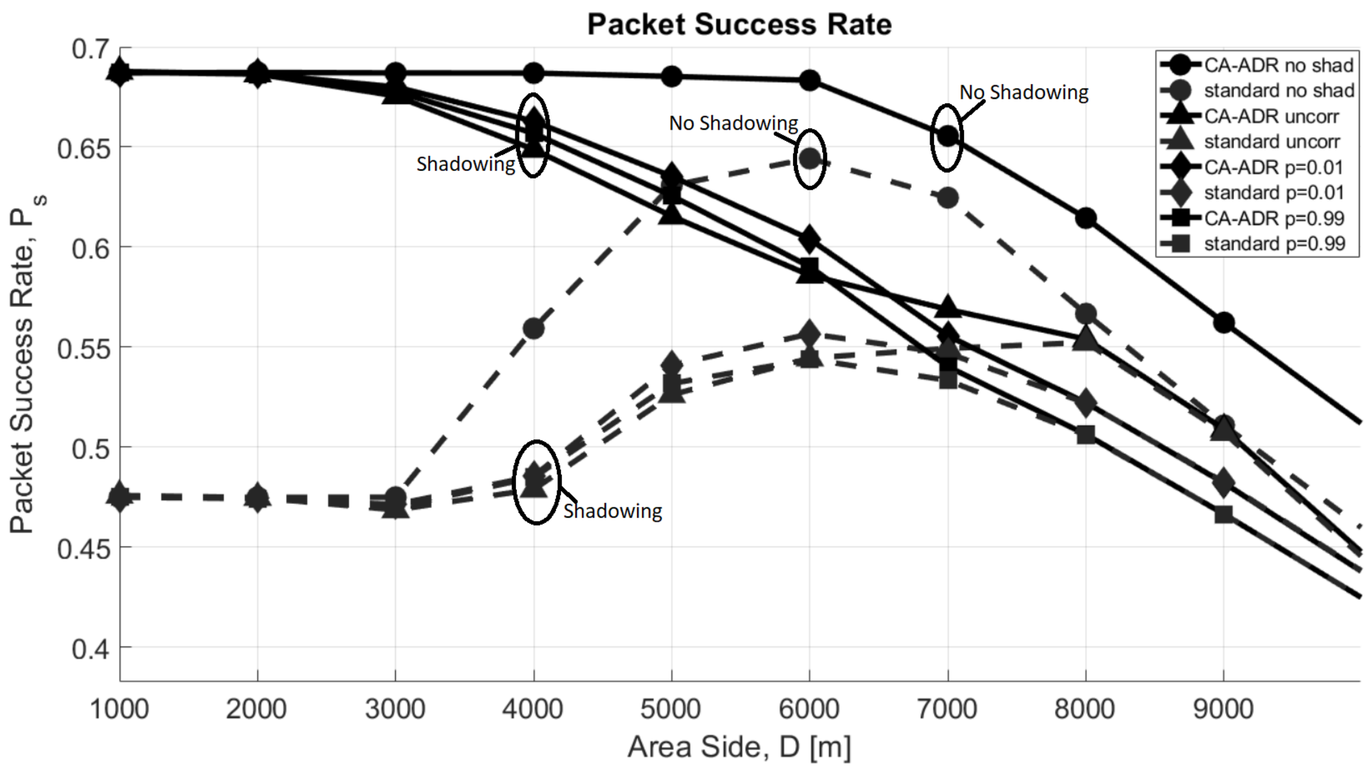


Figure 1.15: Packet Success Rate, P_s, as a function of the area side, D [m] with shadowing analysis

that is varying the level of interference generated in the network; in particular, they are assumed to transmit every $T_\mathrm{i} = 5$ s and $T_\mathrm{i} = 15$ s. As it can be seen, even though the performance is generally worse due to the presence of external interference, the CA-ADR still performs better, although the difference with the standard algorithm becomes smaller.

Differences in terms of delay the two algorithms provide as a result of their decision have been also investigated. Figure 1.17 shows the RTT as a function of the number of EDs. In the case of Class A, the factor which has more relevance is the collision probability, because an uplink packet needs to be received by the NS before it could be able to transmit a packet in downlink. Therefore, the CA-ADR algorithm performs better (i.e., the RTT is lower) w.r.t. the standard solution. In the case of Class C, instead, RTT does not depend on P_S because the ED is always listening, so the NS can send downlink packets even if it has not received anything from the ED. Therefore, the standard solution performs better, because it tends to assign lower SFs (i.e., the minimum value of SF allowing connectivity), resulting in lower transmission times, on average.

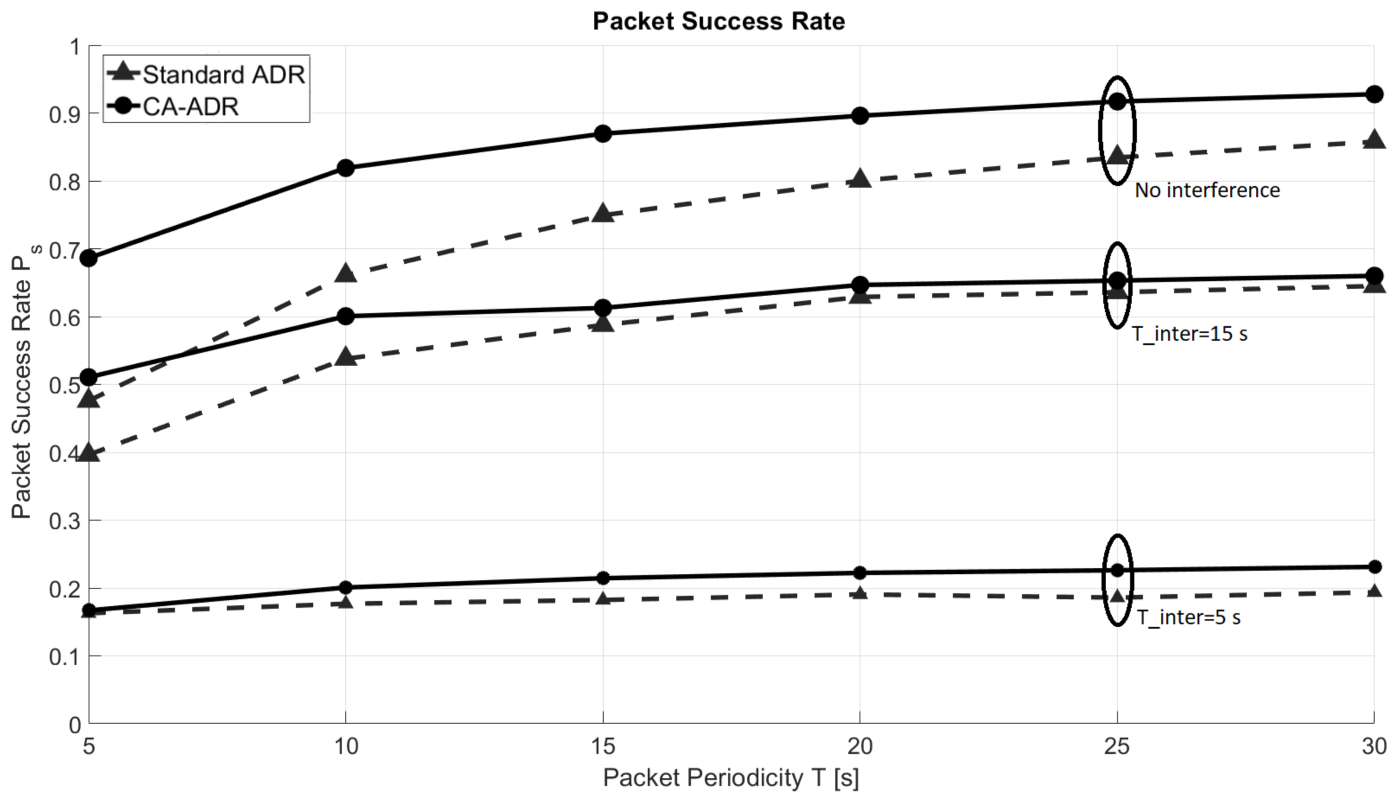


Figure 1.16: Packet Success Rate, P_{s}, as a function of the packet periodicity, T [s]

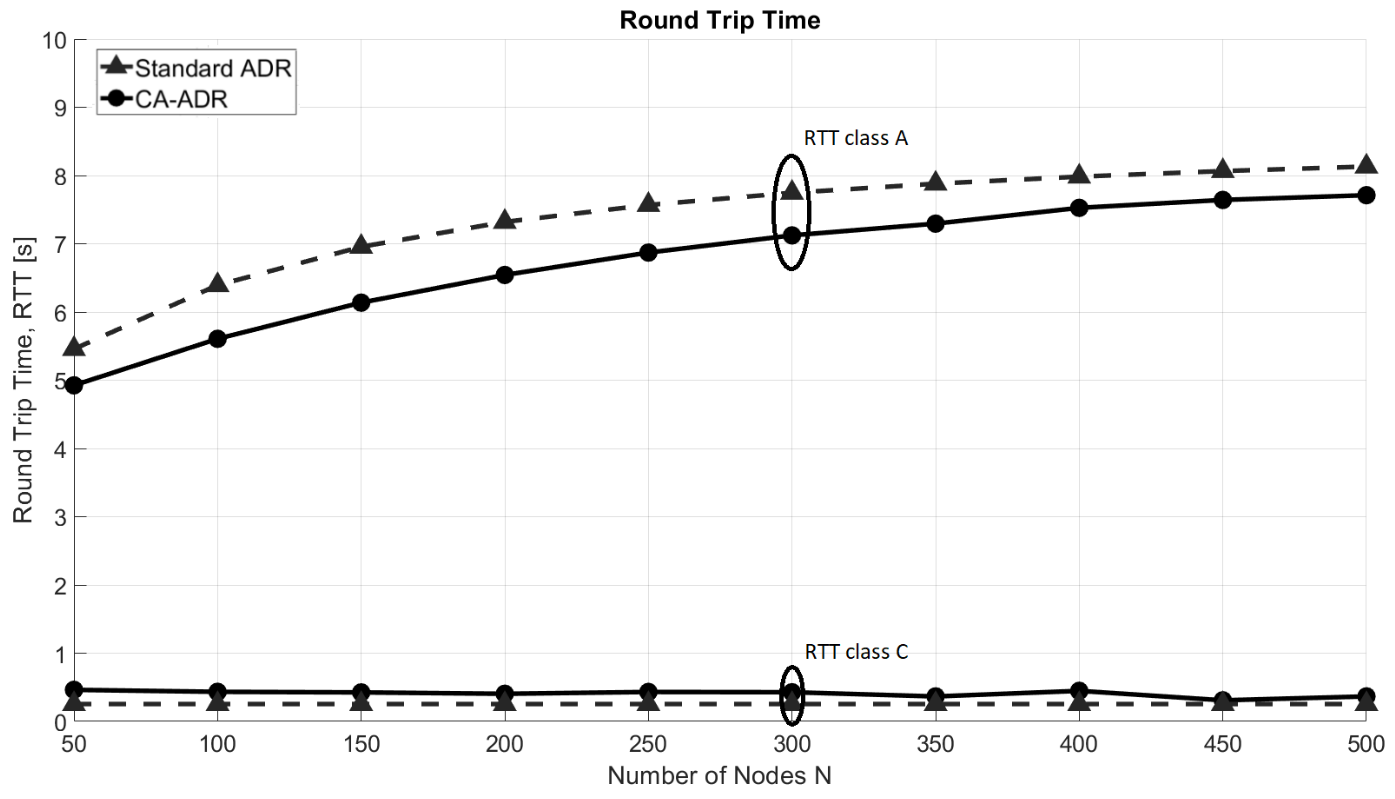


Figure 1.17: Round Trip Time, RTT, as a function of the number of EDs, N.

1.6 Comparing Cloud and Fog Architectures

1.6.1 Methodology and Experimental Results

Experiments have been conducted to characterize delays in two different network architectures considering the same scenario described in the previous section. For this purpose, Chirpstack has been exploited to set up a LoRaWAN network (i.e., the network server). As for the GW, the LoRaWAN^{TM} EMB-GW1301-O gateway has been used, based on the *Semtech SX 1301* chipset working at 868 MHz, with network connectivity provided via Ethernet. Finally, concerning EDs, Idesio Rigers Board 1.0 equipped with the Microchip RN2483 radio transceiver, fully certified 433/868 MHz SX1276 LoRa module and supporting LoRaWAN Class A devices, have been used.

The above-described setup has been replicated in two different architectures: i) cloud-based case, where the NS and all its components were installed on a powerful computing machine, assuming the NS was located in the cloud (see below); ii) fog-based case, where the NS was running on a Raspberry Pi 3 Model B, connected via the University Local Area Network (LAN) to the GW.

The propagation delay, τ_{GW-NS}, introduced when considering the two architectures has been measured. For this purpose, *ping* sessions of 30 minutes each have been carried out considering a public remote NS provided by A2A Smart City[4], which can be reasonably assumed as a cloud server, as well as the Raspberry Pi server implementation, connected via the university LAN to the GW for the fog case. Results of the measurements are provided in Table 1.14.

τ	**Average**	**Standard Deviation**
τ_{GW-NS} (Cloud)	26.2 ms	16.9 ms
τ_{GW-NS} (Fog)	0.4 ms	0.2 ms

Table 1.14: Propagation Delays

Spreading Factor	**Average**	**Standard Deviation**
SF7	61,6 ms	4,26 ms
SF8	68,5 ms	1,9 ms
SF9	90,8 ms	2,78 ms
SF10	132,6 ms	2,33 ms
SF11	260,3 ms	2,6 ms
SF12	389,9 ms	0,98 ms

Table 1.15: $T_{\text{ED}}^{(proc)}$ values

The processing time at the EDs, $T_{\text{ED}}^{(proc)}$, has been also measured: results are reported in Table 1.15 and, as

[4]https://www.a2asmartcity.it/

it can be seen, this time increases with SF. As for the processing time at the GW and NS, $T_{\text{GW}}^{(proc)}$ and $T_{\text{NS}}^{(proc)}$, negligible times have been measured in all traffic conditions analyzed.

Finally, in order to understand the limits, in terms of computing capabilities, of the Raspberry Pi implementation for the NS, situations characterized by high traffic conditions have been considered. Having at disposal only 15 real EDs during experiments, the Lorhammer traffic emulator[5] has been used, which is an open-source tool able to emulate LoRaWAN traffic and redirect it to the NS, that was implemented on the Raspberry Pi. Such a tool offers the possibility to specify the number of GWs and EDs in the network, the frequency with which EDs send packets, and the size of the packets, that is the offered throughput as defined in equation (1.16).

The outcomes of the above experiments have been provided as input to a simulator, characterizing the network-level performance, to derive the latency and RTT as defined above. Figure 1.18 shows such a methodology.

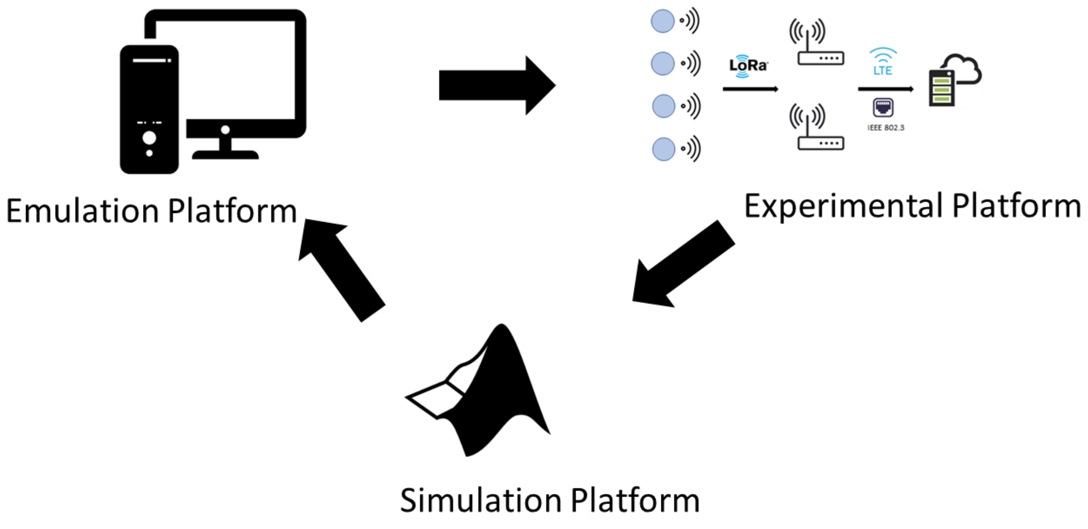


Figure 1.18: Evaluation methodology

1.6.2 Comparison

In this scenario two GWs are considered, which can either be managed by a single centralized NS (cloud case) or are associated with two separated NSs (fog case). Therefore, in the first case, CA-ADR algorithm runs considering all EDs in the area, while in the second case, EDs are divided into two subsets according to the Received Signal Strength Indicator (RSSI) value (for the sake of simplicity, the strongest GW is selected) and the algorithm runs on the two separated subsets. No substantial differences have been observed in terms of SF allocation to EDs, therefore no significant differences in terms of P_s can be observed (see Table 1.16).

[5]http://lorhammer.itk.fr/

N	P_s Cloud	P_s Fog
10	0.997	0.986
50	0.962	0.960
100	0.922	0.921
150	0.884	0.884
200	0.852	0.849

Table 1.16: Success rate in Cloud and Fog architectures

In particular, it can be seen that the advantage achievable in the cloud configuration (where a joint optimization of the entire area is considered) is negligible. On the other hand, it is expected that the cloud architecture will bring larger latency, and this is analyzed in the following.

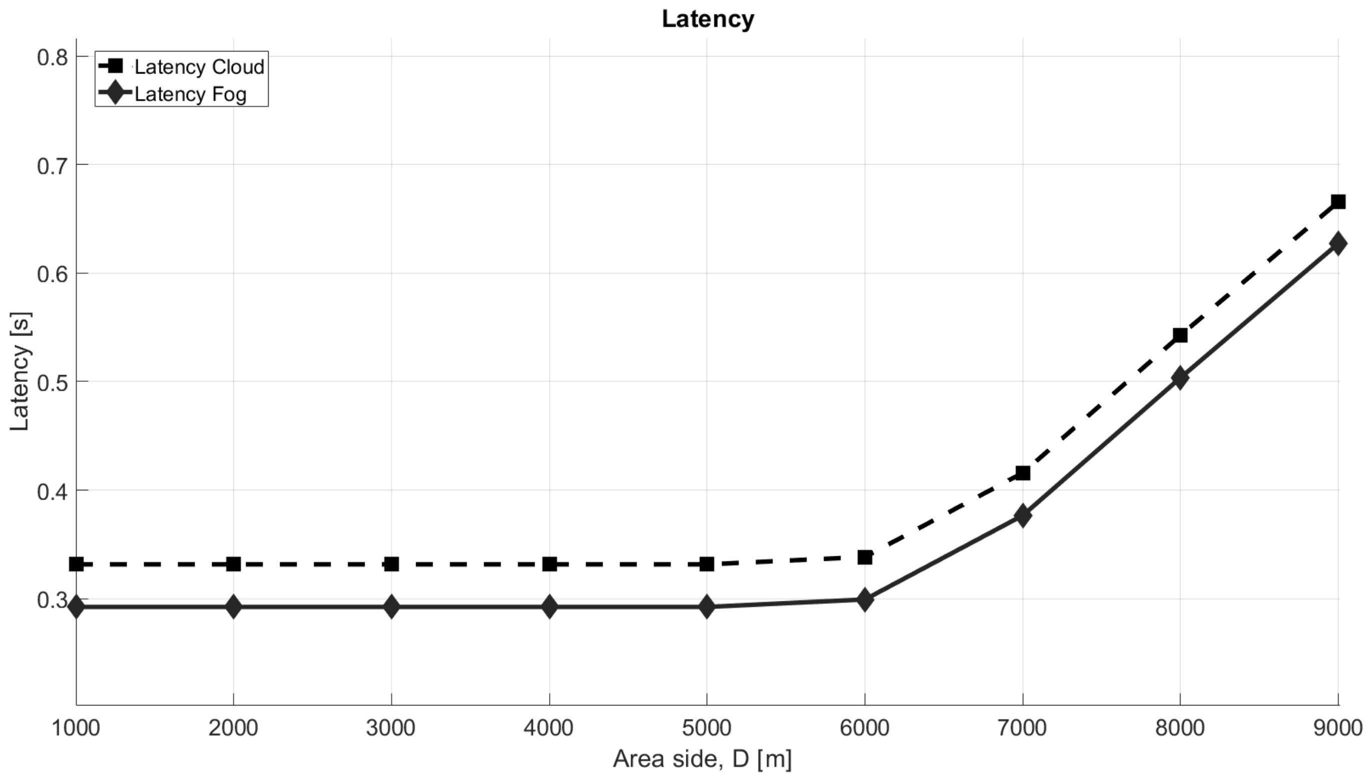


Figure 1.19: Latency, L, as a function of the area side, D

Figure 1.19 shows how the latency evolves when increasing the area size, for the two architectures. By increasing D, higher SF are needed to reach the GWs, therefore the ToA of the packet and then the latency increase. In addition, as expected, the cloud architecture results in larger latency, due to the time needed to reach the cloud infrastructure, where the NS is deployed. The above results show that the cloud architecture works worsening in terms of latency, while slightly improving the network throughput performance. However, to draw final conclusions, it is necessary also to check if a fog-based solution is feasible also in terms of computing capabilities. To this aim, an upper bound in terms of the maximum amount of traffic (in bit/s) the network server is able to process without crashing or causing a relevant delay for the user has been derived. Fixing the packet size to 100 bytes and the packet periodicity to 0.1 s, the number of EDs generating data have been varied. For each value of N, 10000 packet transmissions are emulated to check the CPU usage and to check possible crashing of the NS. The Raspberry Pi implementation proved to handle up to an offered traffic of 40 Mbit/s, reached by setting $N = 5000$, with almost 100% of CPU usage. Such limit is largely compliant with all LPWAN applications, as identified in [21], which demonstrates that the fog solution is sufficiently powerful to manage a LoRaWAN network and this architecture does not present particular drawbacks.

1.7 Conclusions

With the number of LoRaWAN connections approaching 200 million worldwide [8], there is significant interest from both scientific and industrial communities in LoRaWAN technology. However, despite the apparent simplicity of such a protocol, assessing its performance and finding the proper trade-offs by means of analytical tools is not always possible. Therefore, in this chapter, a new MATLAB-based simulator tool covering the physical and upper layers of the LoRa/LoRaWAN protocol stack has been described. The simulator implements a number of unique features such as support of multiple gateways both for uplink and downlink, a detailed interference analysis, receive window prioritization, and energy consumption computation.

In addition, the simulator has been used to provide illustrative results, which demonstrate exciting and not always intuitive trade-offs. The impact of different coding rates in interference-limited and noise-limited scenarios has been investigated. Also, the use of more powerful coding rates turns out to be counterproductive as the number of EDs increases significantly, thus making the network operate in heavy interference conditions. In addition, the impact of downlink transmissions on the average energy consumption of EDs has been assessed.

After the description of some results, a new ADR algorithm for LoRaWAN network, called Collision-Aware ADR, has been proposed. Simulation and experimental approaches have been jointly used in order to compare the new algorithm with two benchmark solutions, considering different performance metrics. Results show that CA-ADR outperforms the standard solution thanks to the orthogonality of signals emitted with different data rates, a fact that allows for drastically reduced collisions among transmissions.

Finally, cloud- and fog-based architectures have been set up and compared in terms of network throughput, latency and processing capabilities. Results demonstrate that the fog architecture based on a common Raspberry Pi is still able to manage a sufficient amount of traffic for many real-life IoT applications. It has also been underlined that the lower latency achievable with the fog architecture can help also in reacting to dynamic environmental changes, i.e. the NS can quickly send commands to EDs to change transmission parameters.

Chapter 2

Narrowband-IoT

2.1 Introduction

NB-IoT was introduced in Release 13 of 3GPP specification documents, emerging as an alternative solution to the LPWAN technologies already available on the market (e.g., LoRaWAN). It is designed to provide efficient connectivity in cellular IoT scenarios and attain a long battery life for a massive distribution of machine terminals and it offers a large number of connections per cell and robust coverage, enabling connectivity even in underground and indoor environments [58]. NB-IoT leverages the LTE standard numerology, but it is adapted for low-cost MTC, supporting a massive number of devices per cell. From LTE, it retains the synchronization, radio access, resources definition and assignment mechanisms. Modifications to regular LTE procedures have been introduced to enhance the link budget and significantly reduce energy consumption, complexity and costs.

NB-IoT and LoRaWAN feature enormous potential to support the development of many different IoT applications. Preliminary analyses and comparisons between the two have been carried out during the last few years. To give an example, the authors of [18] analyze the advantages and disadvantages of choosing one technology over the other according to different indicators, such as battery lifetime, capacity, and reliability. Coverage comparison between different LPWAN technologies is carried out in [59], where authors point out that NB-IoT provides the best coverage probability, even though EDs experience a link loss which on average is 3 dB higher than LoRaWAN. In [60] it is shown that NB-IoT provides uplink and downlink connectivity with less than 5% failure rate, whereas LoRaWAN struggles to provide sufficient indoor coverage. The qualitative analysis of the different KPIs and the results of the energy consumption measurements are reported in [61]. A coverage analysis in a real-life scenario has been carried out in [62], where NB-IoT is shown to outperform LoRaWAN thanks to directional antennas, thus providing better coverage to EDs. The propagation models and the coverage have been investigated in [24], based on the data of an extensive empirical city-wide measurement campaign. In addition to a discussion on the capability of each technology to support links of tens of kilometers, the paper provides some insight into the characteristics of commercial infrastructure deployments by NB-IoT and LoRaWAN operators. The comparative field trials measuring the performance of LoRaWAN and NB-IoT in several propagation-challenging scenarios are reported by the authors of [63]. Results on the coverage provided by three LPWAN technologies, including NB-IoT and LoRaWAN, in a city, are reported and discussed as a part of [64]. In [65], the MAC protocols of several IoT technologies are investigated, specifically in the Smart City domain. The survey [66] suggests that LoRaWAN is more advantageous in terms of energy efficiency and cost, whereas NB-IoT offers more benefits in terms of reliability, resilience to interference, and latency. A comparison between several LPWAN technologies is carried out in [67], which classifies them according to different performance metrics, indicating LoRaWAN as the best solution with respect to energy efficiency and NB-IoT as the best one for high data rate services. Performance comparisons of wireless technologies for specific application domains are carried out in [68], where Smart Grid use cases are taken into account, and in [69], where Smart Water Metering is considered. The results of these studies reveal that NB-IoT provides better scalability compared to LoRaWAN and, thus, it is able to support a huge number of devices with a low packet error rate.

However, most of the cited works provide only qualitative comparisons and focus on one aspect of the two technologies (e.g., energy consumption, scalability, or coverage). Another critical aspect is the different setups considered for the two technologies (e.g., with respect to the infrastructure density or antenna characteristics). In this chapter, instead, a more systematic, accurate, and comprehensive analysis and comparison of the performance of two technologies are introduced:

– employing novel simulators accounting many important aspects of the PHY layer and the Link Layer (LL) of the two technologies,

– addressing within one single study a number of relevant metrics under a number of variables for identical scenarios,

– highlighting the PHY and LL features and mechanisms causing the observed trends and results.

For the purpose of the investigation, two simulators have been developed, one for NB-IoT and one for LoRaWAN (described in Section 1.4), both fed with the same input scenarios and parameters.

The remainder of the chapter is organized as follows: Section 2.2 details the NB-IoT technology. Note that in these sections, for the sake of conciseness, the focus is on the PHY and LL layers procedures that are relevant to the proposed studies and do not attempt to provide a complete description of the technology. Section 2.3 compares the two solutions. Finally, Section 2.4 concludes the chapter.

2.2 Technology background

With respect to the spectrum usage, NB-IoT can operate in three different modes: i) a stand-alone mode, ii) within the guard bands of LTE carriers, or iii) within LTE carriers (in-band mode). As in LTE, NB-IoT evolved Node-Bs (eNBs) employ Orthogonal Frequency Division Multiple Access (OFDMA) in the downlink, and User Equipments (UEs) (the term used in LTE to denote an ED or a user terminal) use Single Carrier Frequency Division Multiple Access (SC-FDMA) in the uplink. However, the modulation schemes are limited to Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK) to reduce complexity and ensure a better link budget.

Hybrid Automatic Repeat Request (HARQ) is implemented in both the uplink and the downlink by default, even though this requirement was relaxed in Release 14, and half-duplex operations are now allowed. In addition, UEs (cat NB1/NB2) implement power control in the uplink, in order to reduce power consumption when possible.

Traditionally, in Release 13 and Release 14, NB-IoT was limited to Frequency Division Duplexing (FDD) operations implying the use of different frequency bands for uplink and downlink transmissions. However, in Release 15 (2019) a new option - Time Division Duplexing (TDD) - has been introduced, allowing to use of the same frequency band both for uplink and downlink. It supports a nominal system bandwidth of 180 kHz (equal to the one of an LTE Physical Resource Block (PRB)) in both uplink and downlink. However, most of the commercial NB-IoT networks are still based on FDD.

In the uplink, the resource grid is composed of multiple subcarrier frequencies with a step (the so-called frequency separation - Δf) of either 3.75 kHz or 15 kHz [70]. The time slots have a duration of 0.5 ms and 2 ms in case of Δf=15 kHz and Δf=3.75 kHz, respectively. On top of this, NB-IoT introduces the notations of Resource Element (RE), which is the smallest time-frequency resource (i.e. one subcarrier and one-time slot), and the Resource Unit (RU), denoting a combination of a specific number of consecutive subcarriers (i.e., 1, 3, 6 or 12) and a specific number of time slots.

In the downlink, the frequency separation is fixed at 15 kHz, and the concept of PRB is used. A PRB spans over 12 subcarriers and 7 Orthogonal Frequency Division Multiplexing (OFDM) symbols and a pair of PRBs is the smallest schedulable unit, which is referred to as a single subframe (thus having the total duration of one millisecond).

The Coverage Enhancement (CE) is mainly achieved by introducing repetitions and thus exploiting time diversity. The transmission of control information and data is repeated a number of times. Each replica has a different coding, and multiple replicas can be combined at the receiver to increase the probability of successful reception. Repetitions increase energy and time consumption, so UEs are often subdivided into three coverage classes, characterized by a different number of repetitions and specific configurations that impacts the coverage. In general, the higher the Radio Signal Received Power (RSRP), the fewer repetitions are used by the UE. The coverage classes are named, from higher to lower RSRP, Normal (N), Robust (R) and Extreme (E).

Table 2.1 offers a summary of the key parameter values.

2.2.1 NB-IoT Device Operation Sequence

To better illustrate NB-IoT procedures, hereafter the operations carried out in an FDD network are detailed [72].

Once powered up, a UE typically starts the cell search procedure, through which it acquires time and frequency synchronization with a cell, identifies it, and receives from the periodically-broadcast information

Table 2.1: NB-IoT Key Parameters Values

Parameter	**Value**
Bit Rate	up to 253.6 kbit/s
Frequency Bands	[400, 2700] MHz
Bandwidth	180 kHz
Link budget	up to 164 dB
TX Range	up to 10 km [71]

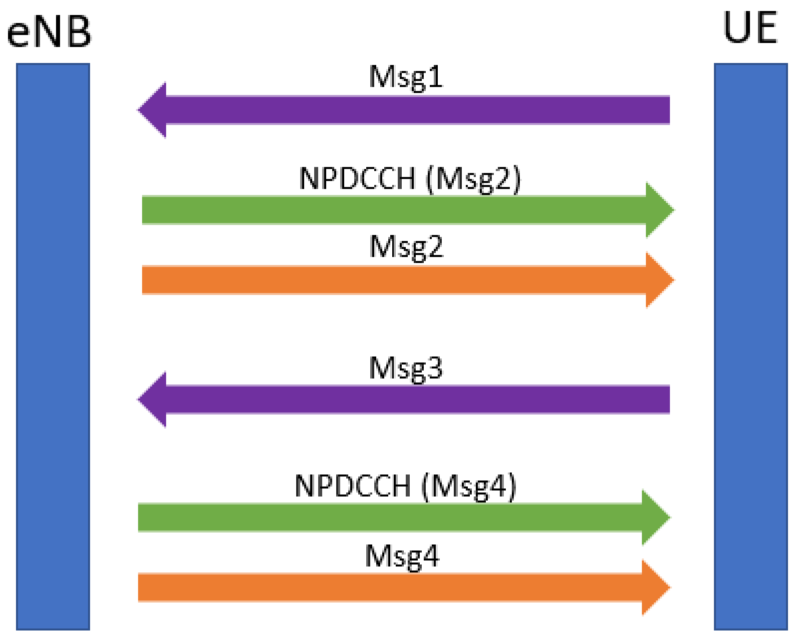


Figure 2.1: Random Access Procedure in NB-IoT

blocks all the information about the configuration of the network, including, e.g., the radio resource configuration.

Once possessing all the required information, the UE may try to establish the connection to the network. For this, it has to execute the Random Access (RA) procedure (see Figure 2.1) to gain access to a radio channel. Specifically, the UE waits for a scheduled Random Access Channel (RACH) window and it transmits a randomly selected preamble (Msg1 in Figure 2.1). A preamble is composed of four groups of symbols and for each of the four groups the carrier is changed. Three different Narrowband Physical Random Access Channel (NPRACH) preamble formats are currently defined (formats 0 and 1 introduced in Release 13 and format 2 added in Release 15), featuring different trade-offs between the on-air time and the maximum communication range. Up to three periodic NPRACH windows can be configured in a single cell, each associated with a CE level and characterized by a different number of preamble symbols repetitions (ranging from 1 to 128 [73]). The selection of the coverage class to be used is made by the UE based on its estimation of the RSRP, the network configuration, and the number of previous unsuccessful RA attempts. The NPRACH transmission is sometimes referred to as Msg 1 since this is the first message in RA procedure.

If multiple UEs choose the same preamble, the transmissions will overlap, but, initially, the eNB is not aware of it. Thus, the eNB delivers the scheduling for Random Access Response (RAR) (or Msg2) in the Narrowband Physical Downlink Control Channel (NPDCCH). The RAR itself is sent using the Narrowband Physical Downlink Shared Channel (NPDSCH) by the eNB, which allocates the resources and specifies the Modulation and Coding Scheme (MCS) as well as the number of repetitions for the next uplink transmission for each of the received RA preambles. In this phase, the UEs that have chosen the same preamble will receive the same RAR. These UEs will send Msg3, which carries the Buffer Size Report (BSR) and the unique data identifying the device, i.e., the UE Contention Resolution Identity, using Narrowband Physical Uplink Shared Channel (NPUSCH) resources and then they wait for Radio Resource Control (RRC) connection setup (or Msg4), scheduled again via NPDCCH. Msg4 is therefore used to grant resources for data transmission and solve possible collisions. A similar procedure has to be repeated each time an unconnected UE requires to access the radio resources to transmit the data.

2.3 Comparing NB-IoT and LoRaWAN

In the following, a thorough comparison between NB-IoT and LoRaWAN will be discussed.

In this section, in particular, assumptions and models are reported. A NB-IoT network simulator has been realized following intentionally the structure of the LoRaWAN simulator (already described in Section 1.4) to enable fair comparison of the two technologies. The structure is illustrated in Figure 2.2.

2.3.1 Network Layout and Data Traffic Configuration

A set $\mathcal{N}$ of EDs, whose size is $N = |\mathcal{N}|$, is randomly and uniformly distributed in a circular area of radius R [km]. The single LoRaWAN GW, as well as the NB-IoT eNB, share the same location in the center of such area. EDs are assumed to generate periodically, with period T [s], a frame with a payload of B [bytes]. All these parameters are configurable and the same scenario is fed as an input to the LoRaWAN simulator as well as to the NB-IoT one.

The well-known Okumura-Hata model (described in Section 1.4) is adopted in both simulators to take into account the attenuation introduced by the propagation. Both urban and rural scenarios are addressed.

2.3.2 Physical layer simulations

For a given position of an IoT node (either ED or UE) in the considered scenario, the SNR experienced in the link is assessed, based on the propagation model and the receiver characteristics (e.g., noise figure). Then, given the transmitter configuration (e.g., BW, SF, CR for LoRa and BW, Δf, MCS for NB-IoT) the signal corrupted by the noise is generated (either LoRa or NB-IoT) and passed to the corresponding receiver, which assesses if the transmitted frame has been correctly received. More precisely:

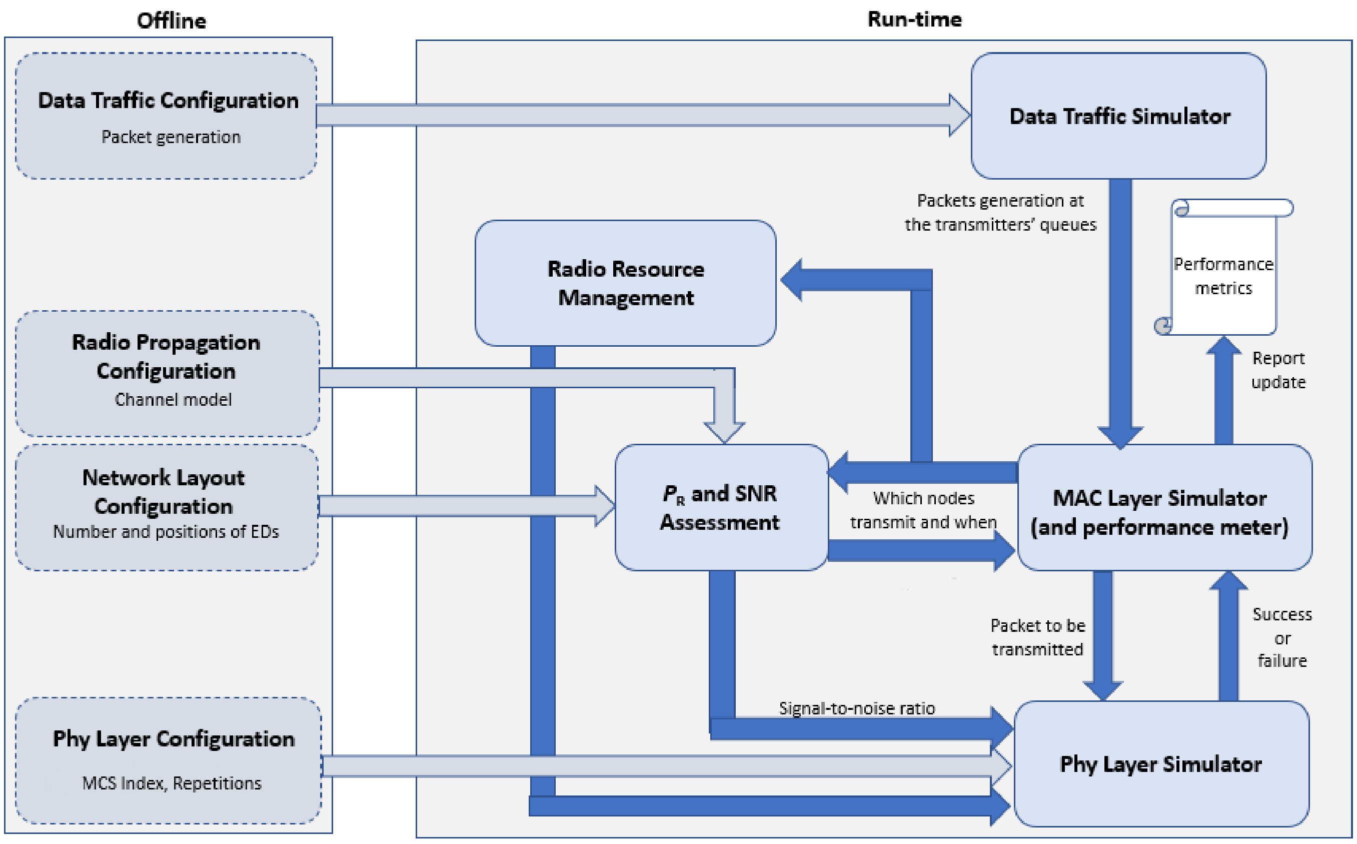

Offline
Run-time
Data Traffic Configuration
Packet generation
Radio Propagation Configuration
Channel model
Network Layout Configuration
Number and positions of EDs
Phy Layer Configuration
MCS Index, Repetitions
Data Traffic Simulator
Packets generation at the transmitters' queues
Performance metrics
Report update
Radio Resource Management
P_R and SNR Assessment
Which nodes transmit and when
MAC Layer Simulator (and performance meter)
Packet to be transmitted
Success or failure
Signal-to-noise ratio
Phy Layer Simulator

Figure 2.2: Block scheme of the NB-IoT simulator.

- **LoRa**. The LoRa simulator reproduces the operations carried out by the transmitter (channel coding, interleaving, gray coding and modulation), the addition of AWGN in the channel and the receiver behavior (demodulation, deinterleaving, decoding). Thus, given the frame of data bits to be transmitted, the simulator assesses whether the currently transmitted frame has been correctly received or not according to different parameters, such as SF, CR and BW and the SNR that characterizes a given link.

- **NB-IoT**. For NB-IoT, the success/failure of NPUSCH transmissions is assessed by means of a NB-IoT PHY Layer simulator, which is based on the LTE Toolbox [74] provided by MATLAB. In particular, for each frame to be transmitted, the corresponding baseband waveform of the SC-FDMA modulated signal is generated by the simulator and passed through the noisy AWGN channel. At the receiver side, the simulator performs SC-FDMA demodulation, decoding and assesses if the frame has been correctly received or not.

In case of collision (i.e., overlapping of different transmissions in frequency and time), the capture effect for both technologies is also taken into account. Specifically, the receiver has still a chance to capture the frame, provided that the SIR is above a given technology-specific threshold:

$$\frac{P_{\mathrm{R}}}{\sum_i P_{\mathrm{R}_i}} \geq \gamma \tag{2.1}$$

where P_{R} is the received power of the wanted signal, $P_{\mathrm{R_i}}$ is the power the GW/eNB is receiving from the i-th interfering node and γ is the threshold. More precisely

- **LoRa**. For LoRa, the possibility that a collision may occur between two LoRa frames (whether they are sent in uplink or downlink) is considered, even though the two frames do not overlap entirely, since the access to the channel is ALOHA-based. In addition, as described in Section 1.4, the simulator is designed to take into account SF *quasi*-orthogonality, so collisions may happen even between EDs using different SF and a frame has still a chance to be correctly received if the SIR is higher than a given threshold which depends on the SFs considered.
- **NB-IoT**. For NB-IoT, collisions may happen only during the RA procedure, since each UE has its own NPUSCH resources for uplink data. As described in Section 2.2.1, two or more UEs choosing the same NPRACH preamble will collide when sending Msg3. However, one of the UE may still complete its procedure and receive Msg4, given that, during Msg3 transmission, its SIR is sufficiently high, which in the simulator has been fixed to $\gamma = -4.6$ dB [58].

2.3.3 Protocol-specific parameters and operations

LoRaWAN

EDs are considered to have already joined the network using one of the two activation procedures supported by LoRaWAN. Therefore, each ED generates a data frame to be sent in uplink periodically every T seconds. The DC limitations are implemented according to LoRaWAN specification version 1.0.1, which imposes that a device (i.e., ED or GW), after sending a frame of duration ToA seconds, computed with the formula reported in Section 1.2, must not use the same frequency band for the next $\mathrm{ToA}(\frac{1}{\mathrm{DC}} - 1)$ seconds. An ED can use a specific SF or the one resulting from the ADR algorithm (in the presented results, EDs are assumed to use ADR, if not stated otherwise). After selecting an SF, it is checked whether a transmitted frame has been correctly received according to the procedure described in Section 2.3.2 and the related performance metrics are derived. No ACK is considered in this case.

The key LoRaWAN parameters used during simulations are summarized in Table 2.2.

Table 2.2: LoRaWAN Parameters

f	868 [MHz]	BW	125 [kHz]
H	1	L_{preamble}	8
DC	1%	CR	1

NB-IoT

In the simulator UEs are assumed already synchronized to the network, so each UE starts by determining a CE level according to the measured RSRP and its set of operations begins from the RA procedure. Each ED is configured according to the CE level to which it belongs (the configurations used in the simulator are reported in Table 2.3 and Figure 2.3) by comparing its RSRP with the $\mathrm{RSRP}_{\mathrm{min}}$ in Table 2.3.

Timely feedback is assumed from the eNB to the UEs during the RACH procedure, in the case of Msg2 and Msg4 transmissions, as uplink transmissions can be more challenging than the downlink ones [75]. Therefore, in the simulator time intervals between the end of such messages, as well as the time needed to transmit them are fixed, as considered in [76, 77].

Table 2.3: NB-IoT Coverage Parameters [58]

	CE0	CE1	CE2
$\mathrm{RSRP}_{\mathrm{min}}$ [dBm]	-101	-111	-121
NPRACH Periodicity [ms]	320	640	640
NPRACH Subcarriers	24	12	12
NPRACH Format	0	0	1
NPRACH Repetitions	2	8	32
NPUSCH Repetitions	2	8	32

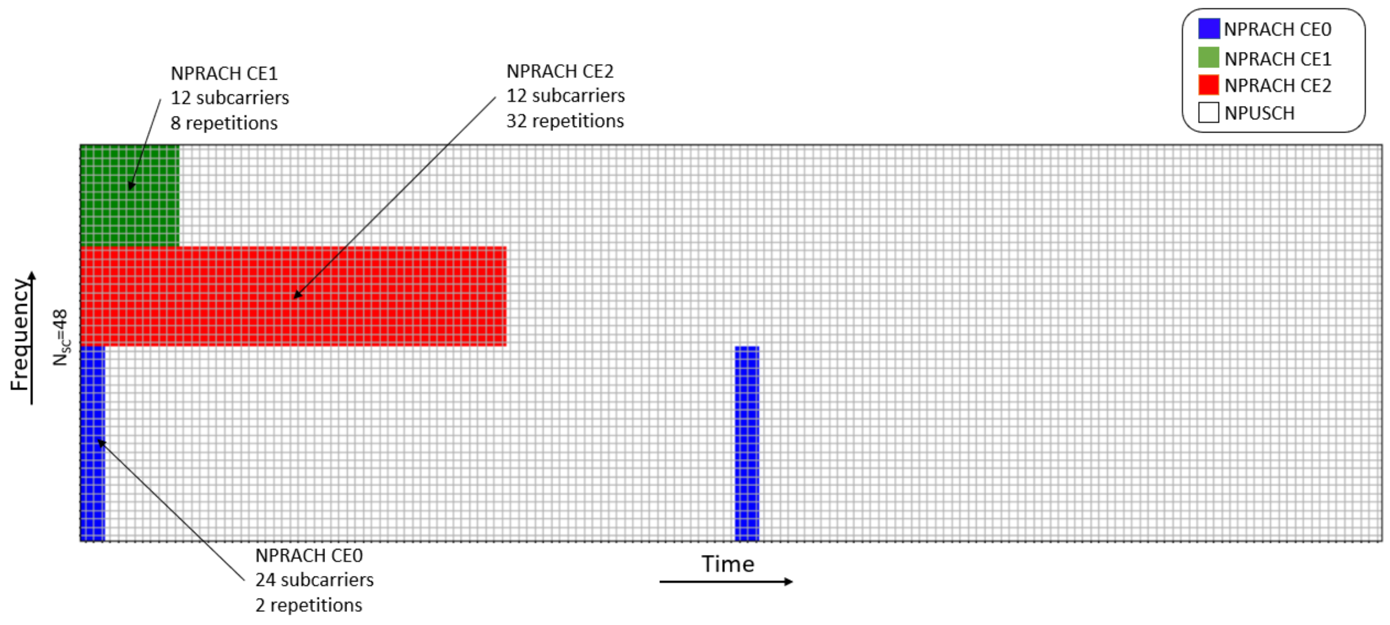


Figure 2.3: NB-IoT Uplink Resources Structure (illustration of one possible configuration)

Once the CE level is selected, the UE sends the random NPRACH preamble at the first NPRACH occurrence and the RA procedure is carried out as described in Section 2.2.1, by checking if every single message is correctly received. If not, the procedure fails and the UE must start from the beginning. Each time the RRC connection is set up correctly, an ED sends uplink data during the scheduled NPUSCH, after the assignment of the resources. A UE asks for a total amount of resources equal to:

$$\text{Transport Block Duration} = N_{\mathrm{RU}} \cdot N_{\mathrm{rep}} \cdot \tau_{\mathrm{RU}} \tag{2.2}$$

where N_{RU} is the number of RUs needed to send a frame, which depends on the MCS selected, N_{rep} is the number of repetitions of the transmission, defined according to the CE level and, finally, τ_{RU} is the duration of one RU.

After sending uplink data, the UE goes in Power Saving Mode (PSM) until a new frame is generated, so the connection to the network can be released and the UE disconnected from the eNB. The NB-IoT related parameters used during the simulation are summarized in Table 2.4.

Table 2.4: NB-IoT Parameters

f	800 MHz	BW	180 [kHz]
Δf	3.75 kHz	MCS Index	6

2.3.4 Performance Metrics

This section presents selected numerical results that were obtained to compare the two technologies.

The comparison has been carried out in terms of Block Error Rate (BLER), network throughput, energy consumption, and latency.

BLER

Having defined as block the MAC level payload, which is the data frame of B bytes passed to the physical layer for modulation and transmission[1], the BLER is defined as the ratio between the number of erroneously received blocks and the total number of transmitted blocks.

$$\text{BLER} = \frac{\text{number of erroneous blocks}}{\text{number of transmitted blocks}} \tag{2.3}$$

Network Throughput

The network throughput, S, has been defined as the number of data bits correctly received by the GW/eNB from all EDs/UEs over the duration of the simulation divided by the duration of the simulated time:

$$S = \frac{8N_{\text{received}} \cdot B}{T_{\text{sim}}} \quad \text{[bit/s]} \tag{2.4}$$

where N_{received} is the number of received frames, and T_{sim} is the duration of the simulation in seconds.

[1]The $8 \cdot B$ bits of a block are processed (channel coding, interleaving, etc.) according to the technology considered to obtain the payload of the physical layer frame to be transmitted.

Parameter	**LoRaWAN**	**NB-IoT**
P_T [dBm]	14	14
I_{TX} [mA]	38	220
I_{RX} [mA]	38	46
I_{Idle} [mA]	27	6
I_{Sleep} [mA]	0.0016	0.003
V [V]	3.3	3.6

Table 2.5: Energy consumption parameters

Energy Consumption

Finally, the mean energy consumption E of a single ED/UE has been defined as:

$$E = E_{\text{TX}} + E_{\text{RX}} + E_{\text{idle}} + E_{\text{sleep}} = V\ I_{\text{TX}}\ T_{\text{TX}} + V\ I_{\text{RX}}\ T_{\text{RX}} + +V\ I_{\text{idle}}\ T_{\text{idle}} + V\ I_{\text{sleep}}\ T_{\text{sleep}} \quad [\text{J}] \tag{2.5}$$

The current, voltage and time duration of the different phases used to compute the energy consumption are reported in Table 2.5 and have been taken from the state-of-the-art works [50, 78]. In (2.5), V represents the voltage, I is the current and T is the time spent in each state, respectively. More specifically, for LoRaWAN, T_{TX} coincides with the ToA of the frame, T_{idle} coincides with a RECEIVE_WINDOW_DELAYs excluding the reception time, T_{RX} is the time spent in reception and T_{sleep} coincides with the time interval between the end of the transmission of a frame and the beginning of the next transmission. For NB-IoT, T_{TX} is the sum of the time spent sending MSG1, MSG3 and NPUSCH data, T_{RX} is the time spent receiving MSG2 and MSG4 and downlink control information (DCI), T_{idle} indicates the remaining time spent during the different phases without transmitting/receiving and, finally, T_{sleep} is the time the UE spends sleeping between the end of the transmission of NPUSCH data and the beginning of the next RA procedure.

Latency

For LoRaWAN, the average latency, L, of a successful transmissions is:

$$L = \text{ToA}_{\text{UL}}(\text{SF}) + \sum_{i=1}^{\infty} (1 - P_s)^i \cdot T \quad [\text{s}] \tag{2.6}$$

where $ToA_{\text{UL}}(SF)$ is the Time on Air of the uplink frame which depends on the chosen SF, P_s is the probability of correctly receiving a frame given the two conditions described in Section 2.3.2 and T is the frame periodicity. It is worth highlighting that, in case of no collisions, the minimum latency experienced by an ED coincides with the ToA of the frame.

For NB-IoT, the average latency depends on the success of the RA procedure as well as the time needed to send uplink data using NPUSCH resources, which depends on the resources assigned to the UE by the eNB.

The minimum latency L_{min} is defined as:

$$L_{min} = T_{\text{MSG1}} + \tau_{\text{MSG1}} + T_{\text{MSG2}} + \tau_{\text{MSG2}} + T_{\text{MSG3}} + \tau_{\text{MSG3}} + T_{\text{MSG4}} + \tau_{\text{MSG4}} + T_{\text{NPUSCH}} \quad [\text{s}] \tag{2.7}$$

where $T_{\text{MSG}i}$ is the time needed to send the i-th MSG, $\tau_{\text{MSG}i}$ is the time interval between the end of MSG i and the beginning of MSG $i+1$ and T_{NPUSCH} is the time needed to send the actual uplink data.

Consequently, the average latency L is equal to:

$$L = L_{min} + \sum_{i=1}^{\infty} (1 - P_s)^i \cdot (T_{\text{NPRACH}} + L_{min}) \quad [\text{s}] \tag{2.8}$$

where P_s is the probability of correctly receiving each message needed for the RA procedure, given the two conditions described in Section 2.3.2, and T_{NPRACH} is the time the UE spends waiting for the next NPRACH occasion (in case the previous RA procedure failed).

The results obtained via simulation are based on the configurations reported in Table 2.6. Each simulation run covers 5 minutes of simulated time and 10000 iterations of simulation are carried out for each run. Results are presented assuming an urban scenario, if not otherwise specified.

Table 2.6: Simulation Parameters

Parameter	**Notation**	**Value**
Number of EDs	N	1,50,100,250,500,750,1000
Number of GWs/eNBs	GW/eNB	1
Area radius [km]	R	1,3,5,7,9
Packet periodicity [s]	T	10,20,30,40,50,60
Payload size [bytes]	B	10,20,25,50,100
GW/eNB antenna height [m]	h_b	30
ED antenna height [m]	h_m	1
Shadowing st. dev. [dB]	σ	3
Carrier frequency [MHz]	f	868 for LoRaWAN 800 for NB-IoT

2.3.5 Numerical Results

The LoRa BLER is plotted in Figure 2.4 as a function of the signal-to-noise ratio for all SFs and CRs, considering a block size of $B = 20$ bytes. One can observe that, for a given SF, passing from CR1 (CR2) to CR3 (CR4) allows a gain of about 1.5 dB in terms of SNR. One also observes that, as expected, passing from CR1 to CR2 or from CR3 to CR4 does not provide any benefit. In fact, CR1 and CR2 do not provide any bit correction capability, whereas both CR3 and CR4 allow correcting only one bit in a codeword.

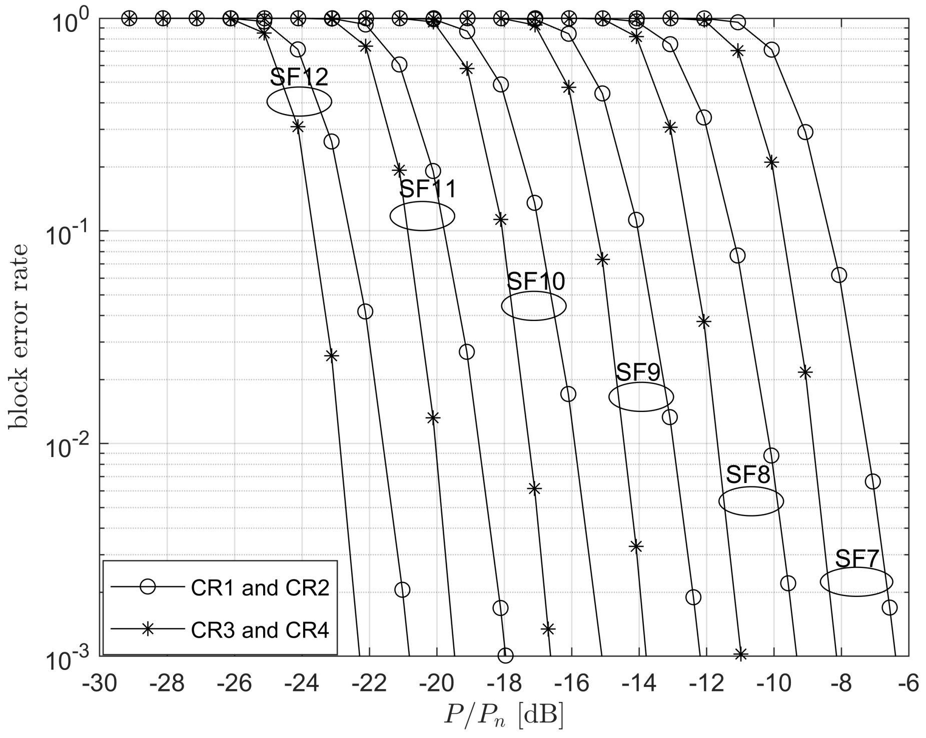


Figure 2.4: LoRa block error rate. BW $= 125$ kHz.

The NB-IoT NPUSCH BLER is shown in Figure 2.5, considering again a block size of $B = 20$ bytes, for different numbers of repetitions. It is clear that, by increasing the number of repetitions, a gain in terms of SNR is achieved.

Comparing the two technologies, it can be stated that NB-IoT is more robust to noise. Considering, for instance, an application requiring a maximum BLER of 10^{-2}, the minimum SNR is in the range $\{-23, -7\}$ dB for LoRaWAN (depending on SF and CR) and in the range of $\{-31, -22\}$ dB for NB-IoT (depending on the number of repetitions).

Figure 2.6 shows the network throughput as a function of the number of EDs in the network[2]. In particular, for LoRaWAN, all EDs use SF=7, SF=12 or the SF resulting from the ADR algorithm, whereas, for NB-IoT, EDs use MCS Index=1, 6 or 13. As expected, NB-IoT almost always provides higher throughput w.r.t to LoRaWAN for medium traffic, whereas for very high traffic the two becomes comparable. However, both demonstrate the same trend, since the network throughput increases with the number of EDs until such number becomes too large and collisions worsen the performance. This can also be seen that there is an optimal number of devices that maximizes the network throughput. Comparing the results for urban and rural scenarios, it can be seen that the throughput for the rural scenario is higher than that of the urban scenario. The main reason is the lower path loss and its lower fluctuation in the rural environment. For NB-IoT specifically, using a high MCS Index (e.g., MCS Index=6 or 13) provides a higher throughput since fewer resources are

[2]In this section, the acronym ED is used to denote both LoRaWAN end-devices and NB-IoT user-terminals.

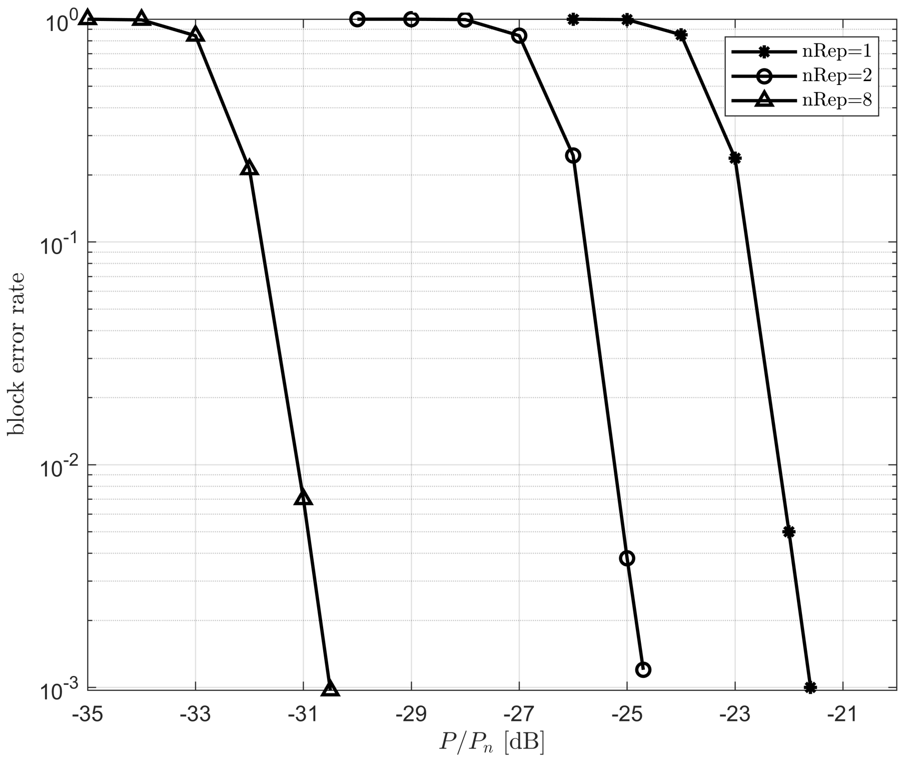


Figure 2.5: NB-IoT block error rate. MCS Index=6.

needed to send the same frame if compared to a lower MCS Index (MCS Index=1). On the other hand, for LoRaWAN, enabling ADR turns out to be better w.r.t choosing only SF=7 or 12, since this allows exploiting the quasi-orthogonality of the different SFs. In addition, it is worth noting that the performance of LoRaWAN is further reduced due to the duty cycle limitation, which particularly affects the case with SF=12.

The energy consumption is shown in Figure 2.7 as a function of the number of EDs. It can be clearly seen that NB-IoT devices feature higher energy consumption, which increases further with the increase of the number of EDs active in the network. This happens because more EDs compete for the access to resources and EDs which fail try again during the next NPRACH occurrences. In addition, lower MCS Indexes make EDs use more resources, resulting in more time spent in transmission.

On the contrary, for LoRaWAN, the energy consumption remains stable regardless of the number of EDs in the network, since, being based on ALOHA, a LoRaWAN ED transmits whenever it has to.

In Figure 2.8 the network throughput S is presented as a function of the radius R of the circular area over which EDs are distributed. As before, for LoRaWAN, EDs use SF=7, SF=12 or the SF resulting from the ADR algorithm, whereas, for NB-IoT, the possibility of having only CE0 or all 3 coverage classes is addressed.

It can be clearly seen that NB-IoT provides higher throughput if compared to LoRaWAN. In particular, when the area becomes larger, the availability of multiple CEs drastically improves the performance. Meanwhile, the performance obtained by NB-IoT when using only one coverage class appears to be worse even than that of LoRaWAN.

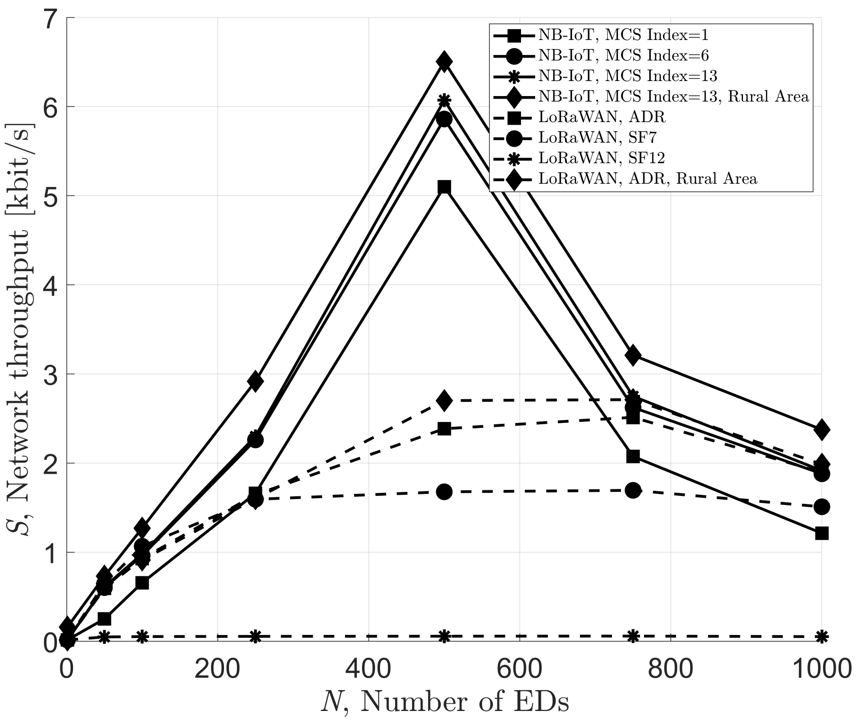


Figure 2.6: Network throughput, S [kbit/s], as a function of the number of EDs, N, with R=3 km, B=20 bytes and T=10 s

For LoRaWAN, using the ADR turns out to be the best choice, since it offers higher throughput, compared to using SF=7 and, thanks to the use of the optimal SF, is not as much affected as SF=12 by duty cycle limitation and collisions, even though using SF=12 should enable better performance in terms of coverage.

For a fixed area size, the rural scenario shows better performance with respect to the urban one because of the better link quality experienced by EDs (being the average path loss lower).

The energy consumption as a function of the area radius R is shown in Figure 2.9. LoRaWAN devices consume much less than NB-IoT ones, whose consumption, especially for large areas and when using all 3 CE classes, strongly depends on the number of transmission repetitions UEs are configured to use when sending their data if they operate in CE 1 or 2.

In Figure 2.10 the network throughput S is shown as a function of the block size, B. For LoRaWAN, EDs use the ADR algorithm, whereas, for NB-IoT, EDs using MCS Index=6 are considered, for both urban and rural scenarios. As a general trend, by increasing the block size, the throughput raises. One can also notice that NB-IoT is able to provide higher throughput w.r.t to LoRaWAN in all cases. Even though LoRaWAN specifications [19] define a maximum payload for each SF for different regions, in this case, the same packet dimensions for NB-IoT and LoRaWAN are addressed for a fair comparison.

The related energy consumption as a function of the block size, B, is reported in Figure 2.11. As expected, NB-IoT consumes more energy than LoRaWAN, with slightly better performance in a rural scenario due to the better link quality and, thus, fewer packet losses. In any case, increasing the block size boosts energy

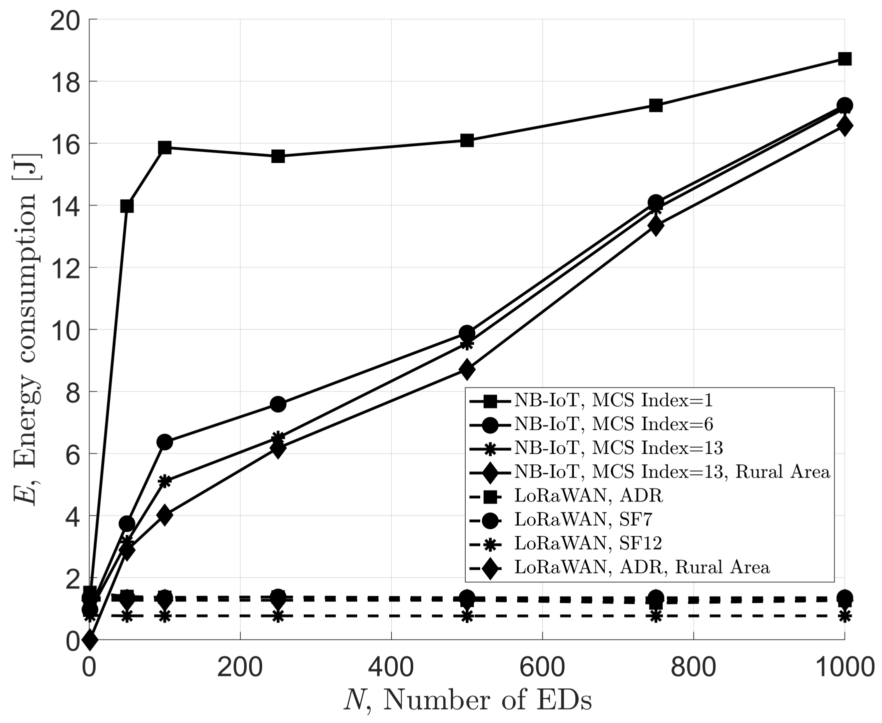


Figure 2.7: Energy consumption, E [J], as a function of the number of EDs, N, implying R=3 km, B=20 bytes and T=10 s

consumption due to the longer transmission time.

Finally, Figure 2.12 shows the latency as a function of the network throughput, S. LoRaWAN huge latency for higher throughput is the consequence of the high number of collisions due to the number of EDs transmitting since, when a frame is lost, the network should wait for the next transmission attempt, which happens T seconds after. For NB-IoT instead, if the RA procedure fails due to collisions, it is possible to try again during the next NPRACH occasion, without waiting for the next frame generation.

Qualititave Comparison

In addition to performance-related aspects, the comparison between LoRaWAN and NB-IoT, and the identification of their actual strengths and limitations, should also consider regulatory issues and business models. From the regulatory viewpoint, there is a clear difference between the two technologies. NB-IoT can be deployed over existing 4G systems. Only Mobile Network Operators (MNOs) who have a 4G license can offer NB-IoT services. This is both an advantage and a drawback. The positive side is that for MNOs, deploying the network is just a technical and investment issue. In many countries all over the world, they have already deployed NB-IoT plug-ins, and there is no other issue in exploiting it from the user viewpoint. On the opposite, LoRaWAN operates on a license-exempt ISM band which is regulated differently from country to country. In Europe, the document providing guidelines for the use of LoRaWAN (and other) technologies is CEPT Recommendation number 70 03. Different national authorities interpret it in various ways. The business model

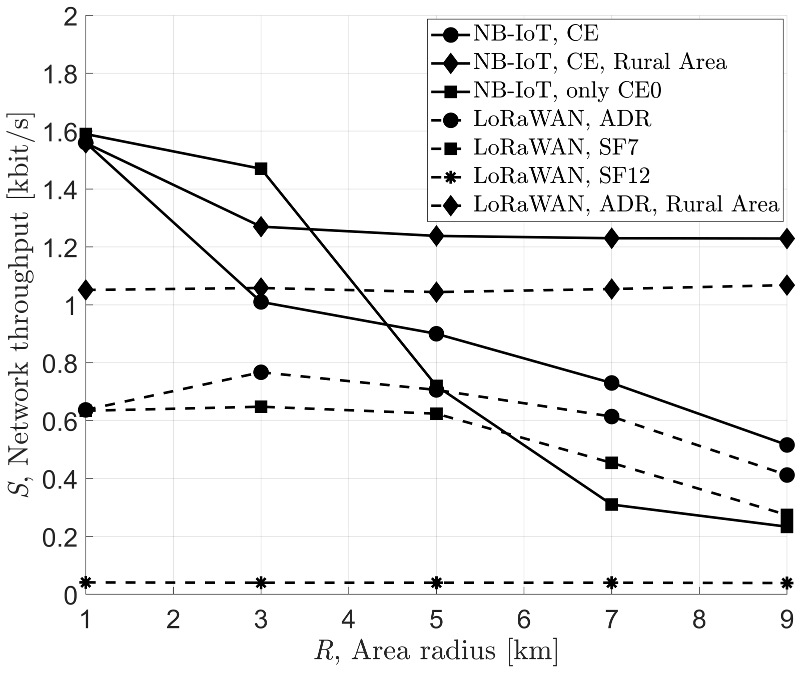


Figure 2.8: Network throughput, S [kbit/s], as a function of the Area Radius, R [km], with N=100, B=20 bytes and T=10 s

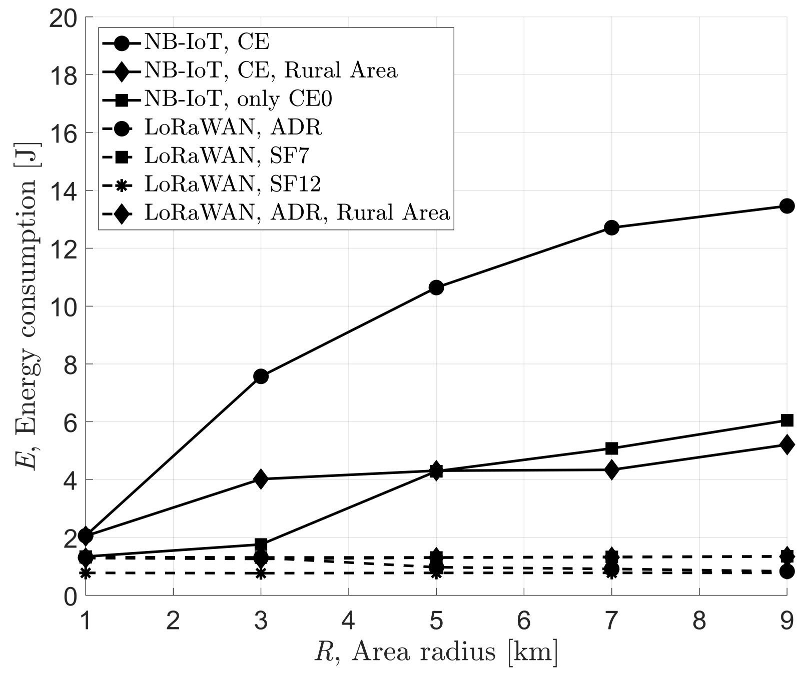


Figure 2.9: Energy consumption, E [J], as a function of the Area Radius, R [km], with N=100, B=20 bytes and T=10 s

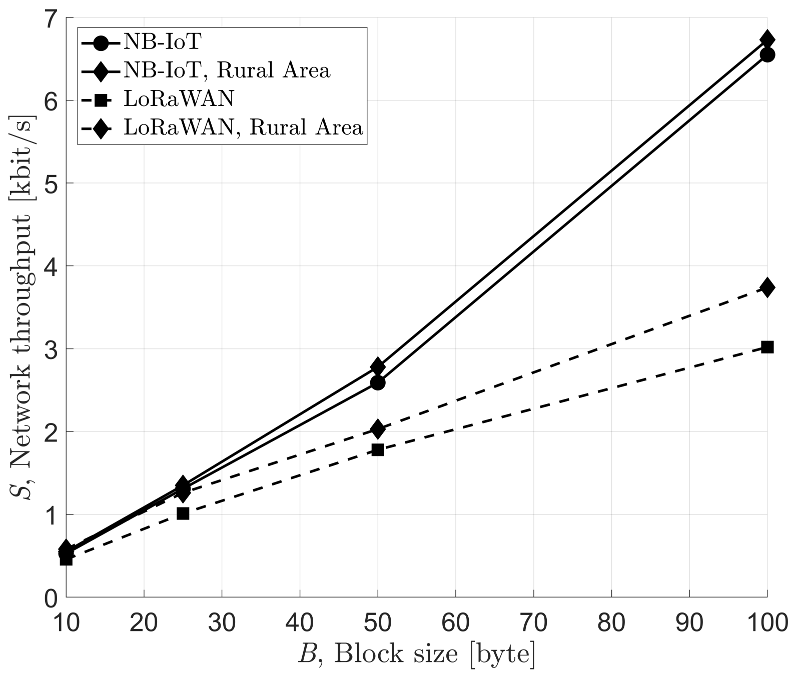


Figure 2.10: Network throughput, S [kbit/s], as a function of the block size, B [bytes], with N=100, R=3 km and T=10 s.

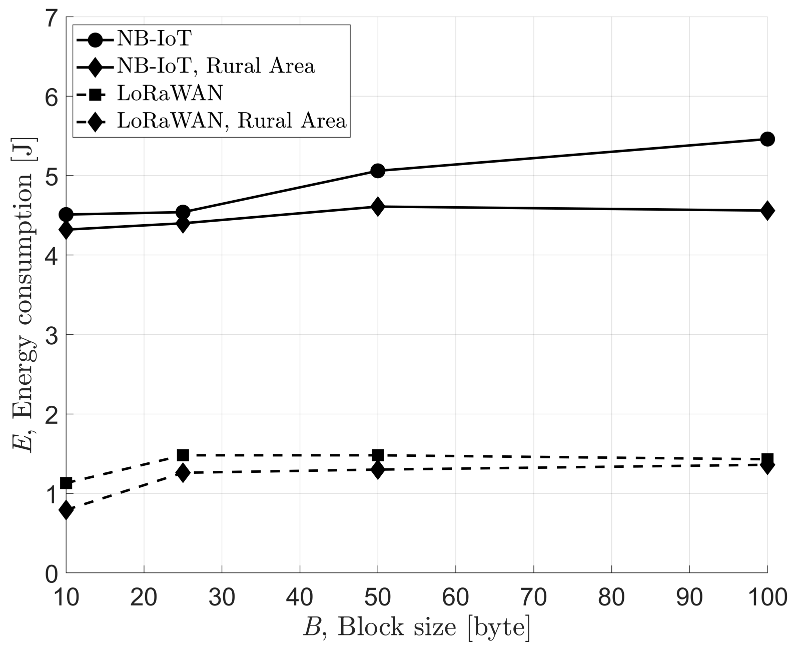


Figure 2.11: Energy consumption, E [J], as a function of the block size, B [bytes], with N=100, R=3 km and T=10 s

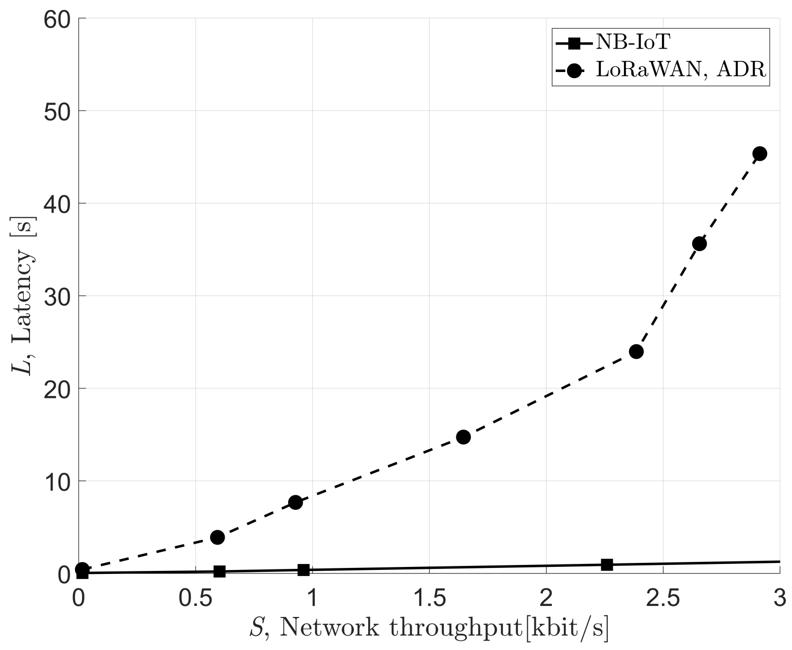


Figure 2.12: Latency, L [s], as a function of the network throughput, S [kbit/s], with R=3 km, B=20 bytes and T=10 s

behind the two technologies is totally different. NB-IoT services can only be offered by MNOs. As long as they deploy the network, it is publicly available (upon payment of a subscriber fee). On the opposite, anyone in principle could offer LoRaWAN coverage; private deployments may be useful for particular applications (especially in remote locations, which are not attractive to MNOs). LoRaWAN networks might be available for free in some areas, as it happens, e.g., with the Things Network - a community of open source LoRaWAN gateway owners.

Other technical aspects that should be considered when comparing the two technologies include the following:

- NB-IoT offers advanced security protocols compared to LoRaWAN, which is particularly insecure when using Activation by Personalization (ABP) join procedure.
- LoRaWAN is limited also from the packet size point of view (according to the SF used, as described in Chapter 1, and does not provide any fragmentation mechanism by the standard.
- Roaming: Sub-GHz ISM bands (normally used by MNOs for NB-IoT wide coverage) are not uniform around the globe, which complicates trans-ocean roaming. NB-IoT terminals supporting multiple bands can handle this. Recently, intra-continental roaming solutions for LoRaWAN (allowing to roam between networks deployed in the same bands) have been delivered. However, their widespread adoption is still underway.
- IP support: NB-IoT supports IP, and many off-the-shelf transceivers implement IP-based protocols

like TCP/UDP, FTP, HTTP, CoAP, MQTT. This enables seamless integration between NB-IoT and the Internet, whereas LoRaWAN requires some form of adaptation layer (most often handled by the NS) in between.

– Handover: LoRaWAN networks do not implement any sort of handover mechanism. NB-IoT has to handle it, though this requires additional signalling.

2.4 Conclusions

In this chapter, NB-IoT technology has been described and carefully compared with LoRaWAN, accounting for technical aspects, both at PHY and Link layers, and regulatory issues. As one can expect, the two technologies differ in many aspects and both have strengths and weaknesses but, depending on the specific application, the best solution can be identified based on the above-reported discussion and numerical results. To summarize, NB-IoT implements a more robust modulation and coding scheme, together with a highly reliable Link layer, at the cost of larger energy consumption. Therefore, NB-IoT is more suitable for applications that are demanding in terms of reliability and network throughput. In addition, it is not limited by any regulation in terms of duty cycle, thus devices can transmit more frequently or bigger data volumes. On the other hand, LoRaWAN is convenient for applications having strict requirements in terms of lifetime (i.e., for battery-constrained use cases) and where the reliability requirements can be relaxed. Notably, subject to some configurations and scenarios, either of the considered technology may outperform its counterpart. It is important to keep this in mind when considering the communication technology to be used for the particular use case scenario. Also, this motivates the further more in-depth study of the effects the different network parameters have on the technology performance as well as the development of relevant optimization mechanisms. Unfortunately, this aspect (especially for NB-IoT technology) has got somewhat limited attention so far.

Chapter 3

UAV-Aided NB-IoT Networks

3.1 Introduction

In the previous chapter the performance of a NB-IoT network have been investigated by considering a terrestrial deployment (i.e., terrestrial BSs in the area of interest). Even though statistical reports already proved a steady increase in the number of machine-type or IoT links[1], the massive presence of IoT devices might not be the only major challenge to be addressed for the future. Other key challenges lie on the more differentiated and stringent requirements on communication performance - the demand - imposed by the several applications and use cases possible. These may include autonomous vehicles, wearables, industrial IoT for Industry 4.0, data monitoring, alarm detection, municipality services and many others, in which one can observe that commonalities are few. These aspects call for new paradigms to network design. To avoid the densification and deployment of new terrestrial bases needing huge investments in capital and operational expenditures, a viable and largely foreseen solution can be found in mobile BSs.

UABSs, (i.e. UAVs where mobile BSs are mounted), represent very interesting means to add the required flexibility and scalability for future networks. UABSs have, in fact, the potential to fly on-demand and exactly where it is needed. Moreover, they are not tied to roads, not affected by traffic congestion and can feature good connectivity with both, on-ground users, and terrestrial BSs (i.e., backhaul), thanks to the large probability of being in Line of Sight (LoS). The usage of UABSs is especially suited for massive Machine-Type Communication (mMTC) and IoT links since IoT nodes are mostly static (do not change their position over time) and their traffic demand is usually predictable [79]. The knowledge of these two basic inputs allows to make decisions in advance on the trajectory to follow to maximize IoT service, plan periodical flights when the demand arises, modify the UABS behavior if needed, and so on. This may be further relevant if these mobile BSs have to direct themselves in remote and different areas where the nodes are placed. To consider a scenario with the plurality of IoT applications mentioned before, as pictured in Figure 3.1, a massive number of IoT devices scattered in different zones of a service area, requesting to transmit a periodical packet, is

[1]See, e.g., https://www.statista.com/statistics/802690/worldwide-connected-devices-by-access-technology/ or https://www.ericsson.com/en/mobility-report/reports/november-2019/iot-connections-outlook.

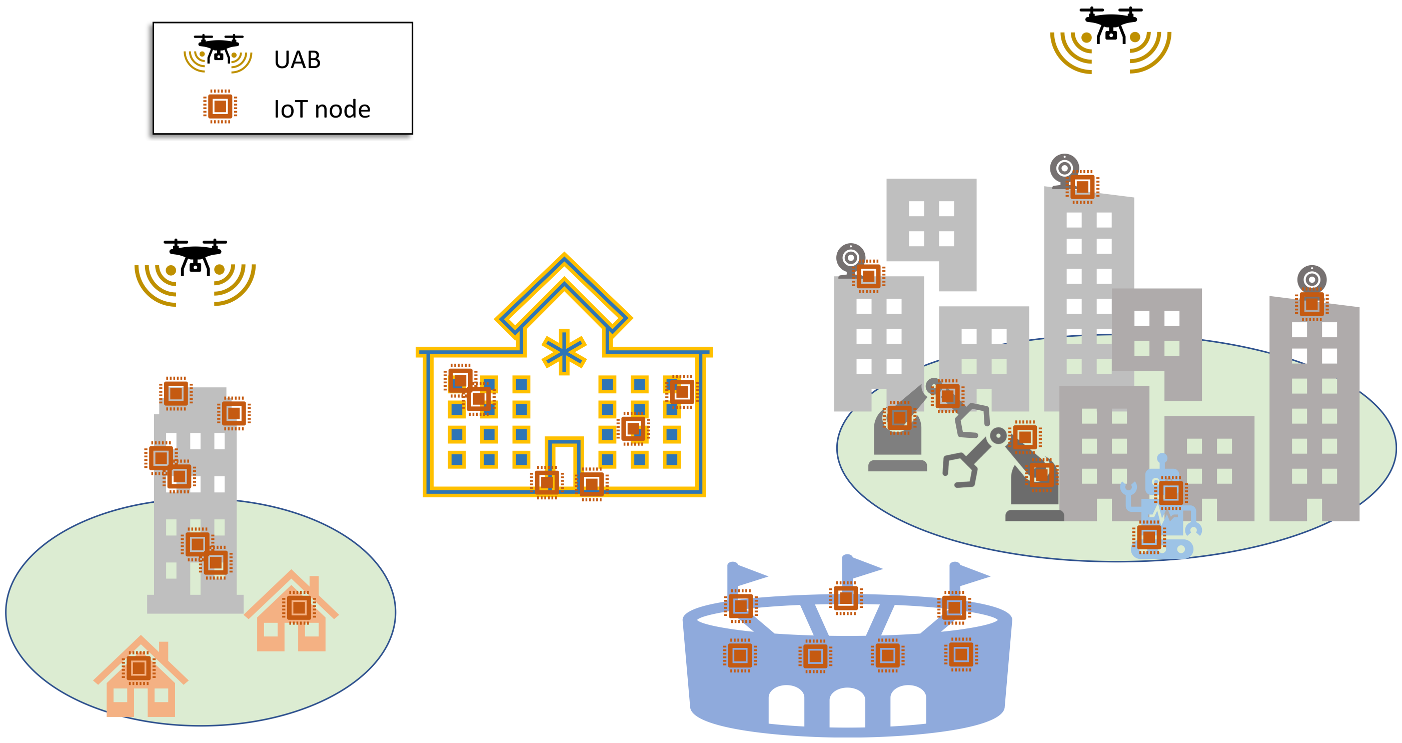


Figure 3.1: Illustrative example of a scenario with multiple IoT applications and nodes served by UABSs.

considered in this chapter. This is a realistic scenario and perfect for UABS services during a single flight.

Thanks to their versatility, flying network nodes like UAVs gained an ever-increasing interest from researchers to standardization bodies. The 3GPP, after considering the feasibility of UAVs being UE (i.e. end users of the cellular network), started approaching on-board radio access for UAVs (denoted with the acronym UxNB) at the beginning of Release 17 [80, 81]. In fact, if a UAV could have installed the same radio-frequency equipment with a similar protocol stack to target both IoT nodes and broadband users, it would be convenient for mobile operators. To this purpose, there exist a number of technologies targeting IoT applications that follow the fourth-generation (4G) numerology [58]. In particular, the focus will be on NB-IoT for the design intended to target low-end IoT applications with low data rates, delay tolerance, massive connections, and extremely wide coverage [82].

Initial studies on NB-IoT UAV-aided networks focused on the key link-level considerations, and specifically on the characterization of path loss and its impact on the so-called Air-to-Ground (ATG) channel [83, 84]. These activities on UABSs were followed by the aim of finding an acceptable trade-off between coverage, capacity, and connectivity, as in [85]. To be more specific, in [84] the effect of the user-UAV incident angle w.r.t. the ground plane as a function of drone height is studied. It is defined as *elevation angle*, and its aperture may determine whether a link is or not free from obstructions and in LoS conditions. Differently from this chapter's objective, the majority of papers address the general users, forgetting the implications and limitations of the specific protocol procedures. For example, dynamic trajectories for the 3D space are studied in [86] with the purpose to connect IoT nodes at their activation time. Authors jointly optimize the transmission power of ground nodes, the overall energy spent in movement, and the choice of the next stop of each UABS. Optimization algorithms have also been studied for UABS placement to achieve optimal energy efficiency [87, 88] by minimizing power consumption and transmission delays. Finally, [89] studies trajectories for energy minimization in a NB-IoT context. The NB-IoT protocol is considered in [82], whose research targets are low data rates, delay tolerance, massive connections, and extremely wide coverage [90]. However, these works usually lack a fine-grained protocol study and focus on the optimization of one metric above another. On the other hand, the goal is to provide a more general overview of potentials and challenges with a deeper focus on protocol details rather than the optimization of a single performance metric. Also, optimization frameworks struggle to handle large input instances (e.g. a massive number of nodes in the scenario) because of the excessive computation times, in terms of days or weeks. For example, [91] and [92] consider less than 10 users in the service area. In [93], it is optimized the 3D locations of UABSs for wireless powered NB-IoT. Another work worth mentioning is [94]. Authors propose a NB-IoT model to collect underground soil parameters in potato crops using a UAV-aided network. The analysis in this case is mostly application-dependent and therefore differs from the general evaluation with different metrics provided in the following.

However, the literature still lacks a detailed model and protocol analysis of similar scenarios and setups. Therefore, the focus of this chapter is to extend and further discuss the system dynamics of NB-IoT networks served by an UABS, rather than compare the proposed approach with other research activities. This study may help to extract the major impacts of the overall protocol stack of the NB-IoT technology on UAV-aided networks.

In this chapter, different approaches to aerial support for NB-IoT networks are proposed, in order to provide a general overview of the challenges and potentials of these systems. To properly assess the network performance of UABSs serving NB-IoT nodes, performance metrics as the percentage of completely served nodes, the throughput provided, the latency which has to be expected, and the IoT nodes' energy consumption are jointly considered. The key contributions can be summarized as follows:

- a UAV-aided NB-IoT scenario is proposed, with several hundreds of nodes located in different parts of the area to simulate diverse applications. Differently from other papers in literature, the NB-IoT technology is considered in detail as specified by the 3GPP documents, including the signaling procedures, and studied in all of its features;
- performance in terms of the number of completely served nodes, achieved network throughput, perceived latency, and energy consumption of nodes are investigated;
- the impact of using different UABS trajectories is analyzed;
- the implications of the three NB-IoT coverage classes and the impact of varying UABS parameters, such as speed, height, and trajectory selection are considered.

The chapter is organized as follows. Section 3.2 describes the scenario and the network model. Final simulation results and related discussions are reported in Section 3.3. Finally, Section 3.4 concludes the chapter.

3.2 System Model

The scenario taken into consideration has the scope to recreate a realistic deployment of IoT nodes, denser in service areas and absent in other locations. For example, IoT devices can be positioned at smart traffic junctions, in city parks, at waste collection points, in parking lots, or into buildings, to name just a few. Thus, practically, clusters of IoT nodes are distributed in the area, characterized by close vicinity when they implement the same application requirements. The spots scattered with NB-IoT nodes are considered not to be served adequately by the terrestrial infrastructure and, for this reason, an UABS equipped with NB-IoT radio access is sent to supply the service instead.

3.2.1 Network Scenario

To be specific, the scenario is modeled using a Poisson Cluster Process, namely the Thomas cluster process (TCP) [95], as proposed in [79] and conventionally done in the literature (see, e.g., [96]). The TCP is a stationary and isotropic Poisson cluster process generated by a set of offspring points independently and identically distributed (i.i.d.) around each point of a parent Poisson Point Process (PPP) [95]. In particular, the locations of parent points are modeled as a homogeneous PPP, with intensity λ_p, around which offspring points are distributed according to a symmetric normal distribution with variance σ^2 and mean value m. As a consequence, the intensity of the offspring points can be written as $\lambda = \lambda_\mathrm{p} \cdot m$. In the proposed scenario, offspring points represent the IoT nodes asking for service, whereas parent points are only reference coordinates for cluster centers.

A square area of size L x L m^2, where offspring points are located according to the description above, is simulated. A sample scenario together with possible UABS trajectories in depicted in the following (see Figures 3.2, 3.3, 3.4).

A single UABS is considered to decrease capital and operational expenditures and to simplify the final numerical evaluation. Please note that, in this model, the extension to the case of multiple UABSs do not necessitate additional complex settings, and therefore it is not a major focus of this work. However, for the sake of completeness, it will be discussed in the following.

The UABS is assumed to start its flight from a fixed position, which can be considered as a recharge station and where it has to come back at the end of the trajectory. In this way, it can recharge or change its battery for the next flight. The capacity of the UABS battery is assumed to be sufficient to enable a full round trip over any trajectory. This is a reasonable assumption provided that the UABS has no heavy payload other than the RF equipment and the flight time is no longer than half an hour [97, 98]. The UABS is assumed to fly at a constant altitude from the ground between 200 m and 300 m (not violating the regulations in EU).

3.2.2 Channel Model

Motivated by the short-sized traffic demand, the backhaul link UABS-terrestrial BS can be assumed as free-space propagation, and the capacity achieved is sufficient for both the UABS control links (for maneuverability and command and control signals) and data forward. The propagation model affects the UABS-ground node-link and is therefore known as ATG channel. Consequently, the received power, P_{rx}, as a function of the transmit power, P_{tx}, can be expressed as: $P_{\text{rx}}[dBm] = P_{\text{tx}}[dBm] - A_L[dBi] - L_{\text{ATG}}[dB]$. The ATG propagation can statistically model the loss value, L_{ATG}, as the reference considered here for drones in urban environments [83, 99]. According to this model, connections between drones and nodes can either be LoS or Non-Line of Sight (NLoS). For NLoS links, the signals travel in LoS before interacting with objects located close to the ground which results in shadowing effects. p_{LoS} denotes the probability of connection being LoS. The probability p_{LoS} at a given elevation angle, θ, is computed according to the following equation

$$p_{\text{LoS}} = \frac{1}{1 + \alpha \exp(-\beta[\frac{180}{\pi}\theta - \alpha])} \tag{3.1}$$

with α and β being environment-dependent constants, i.e. rural, urban, etc, and adopted as given in [83, 84]. Equation (3.1) determines for every link if it is in LoS or NLoS condition, impacting then the value of ξ_{LoS} in equation (3.2). The LoS path loss model is given by:

$$L_{\text{ATG}}(d)[dB] = 20 \log \left(\frac{4\pi f_c d}{c} \right) + x_{LoS}\xi_{LoS} + (1 - x_{LoS})\xi_{NLoS} + \eta \tag{3.2}$$

where x_{LoS} equals 1 in case of realization of LoS links and 0 otherwise. ξ represents the shadowing coefficient which depends on LoS or NLoS conditions as well and is set as described in [83, 84]. Then, c is the speed of light, f_{c} is the center frequency, and d is the transmitter-receiver distance in meters. An additional penetration loss, η, as for in indoor monitoring or basement applications is considered.

If the received power is above the receiver sensitivity, the node is in the connectivity range of the UABS. Because of the fact that NB-IoT may have three coverage classes, there are three sensitivity thresholds, one for each class ce, denoted as $P_{ce,\text{min}}$. Once the device is connected and synchronized to its coverage class signalling, it can attempt to access the channel through the NB-IoT NPRACH (see Chapter 2), so that, if succeeded, it may be given resources to transmit its data. The number of resources assigned determines the packet size that the node can transmit in the given time window. Note that, since IoT nodes are the devices more limited in their characteristics, for example considering the maximum transmit power, in the following some assumptions are made:

- the connectivity range is defined by the uplink,
- the downlink control communication is error-less.

3.2.3 Traffic Model and Metrics

Each node will then request to the UABS to transmit one uplink packet of size B. NB-IoT nodes are assumed to be already synchronized to the cellular network; they start their operations from the RA, exchanging the required NB-IoT signalling messages and, if completed correctly, the required uplink resources are scheduled in NPUSCH. As one can imagine, the procedure is robust but its completion is not guaranteed. In fact, the

Parameter	I_{tx} [mA]	I_{rx} [mA]	I_{idle} [mA]	I_{sleep} [mA]	V [V]
Value	220	46	6	0.003	3.6

Table 3.1: Energy Consumption Parameters

main obstacles may be found in the UABS movement, channel fluctuations and collisions. The last two are application and environment dependent, while the first may be subject to a proper trade-off of the different latency, energy consumption, throughput and success rate metrics. In the following, pedex u will indicate a generic node in the network.

Latency, that is the interval elapsing between the instant when the node has a packet to transmit to the service completion, is computed as:

$$\Delta\tau_u = \tau_{u,\text{tx}} - \tau_{u,\text{start}} \tag{3.3}$$

where $\tau_{u,\text{tx}}$ is the time instant in which the node transmits its packet and $\tau_{u,\text{start}}$ when this data is first available. For ease of evaluation, $\tau_{u,\text{start}}$ is assumed equal to the time in which the UABS starts its trajectory for all nodes u. Then, the throughput, t_u, achieved by a single node u whose transmission is considered successful depends on $\Delta\tau_u$ as:

$$t_u = \frac{B}{\Delta\tau_u} \tag{3.4}$$

If a node u_n is not able to transmit its packet demand, it would be $t_{u_n} = 0$ b/s. Finally, the energy consumption has to account for the signalling related to the RA procedure. It holds:

$$\begin{aligned} E_u =& E_{\text{tx}_u} + E_{\text{rx}_u} + E_{\text{idle}_u} + E_{\text{sleep}_u} = \\ =& V\ I_{\text{tx}}\ T_{\text{tx}_u} + V\ I_{\text{rx}}\ T_{\text{rx}_u} + V\ I_{\text{idle}}\ T_{\text{idle}_u} + V\ I_{\text{sleep}}\ T_{\text{sleep}_u} \end{aligned} \tag{3.5}$$

where V indicates the voltage with which the IoT node is powered, I_{tx} and I_{rx} the current needed in transmission and reception mode, respectively, I_{idle} the current present when the node stays in idle, and I_{sleep} the current during PSM. Similarly, T_{tx_u}, T_{rx_u}, T_{idle_u}, T_{speed_u} indicate the times spent by each node u for being in the corresponding operation mode. Of course, this depends on the message exchange described in Chapter 2 (that includes an alternating of transmission and reception modes). Some of these parameters are fixed and shown in Table 3.1 [78].

3.2.4 UABS Trajectories

As mentioned, multiple possible trajectories for one UABS flying over clusters of IoT nodes have been analyzed. These trajectories follow a predefined path since IoT nodes are placed in fixed positions and usually have a traffic demand that is easily predicted or periodical. In this way, the static positioning of multiple drones can be avoided, which would require increasing capital expenses. Moreover, a static deployment of hovering UABSs has further energy consumption issues from the UABSs side, which may be less under control. The following possible trajectories are considered:

- Circular path;
- Paparazzi-like trajectory;
- Flight following the solution of a Traveling Salesman Problem (TSP) over clusters' parent points.

Each of these trajectories has its pros and cons. Thanks to the wide coverage which can be achieved by the three coverage classes of NB-IoT, the circular path might be an option for its short path length. On the other side, if IoT nodes are not adequately covered, the paparazzi-like trajectory is able to scan the entire area. However, since the UABS has to serve clusters of fixed nodes, there is a third option. Locations of the parent points can be considered as reference coordinates to model the trajectory as a TSP [100], as proposed in [79]. In this way, it can be observed the effectiveness of these choices compared to other known alternatives. The TSP determines, for a finite set of points whose pairwise distances are known, the shortest route connecting all points. The circular path has a radius length equal to half the length of the circumscribed circle of the service area. To better adapt the circular trajectory to the nodes' deployment, the perimeter of the service area is considered as the maximum extension of the nodes' location in all directions, centering it and the circular path consequently. A similar implementation is repeated for the paparazzi trajectory, which considers also a *sensing* radius for the UABS to define the width of the serpentine, fixed to 500 m. Examples of trajectories and cluster positions for a scenario snapshot are represented in Figures 3.2, 3.3 and 3.4.

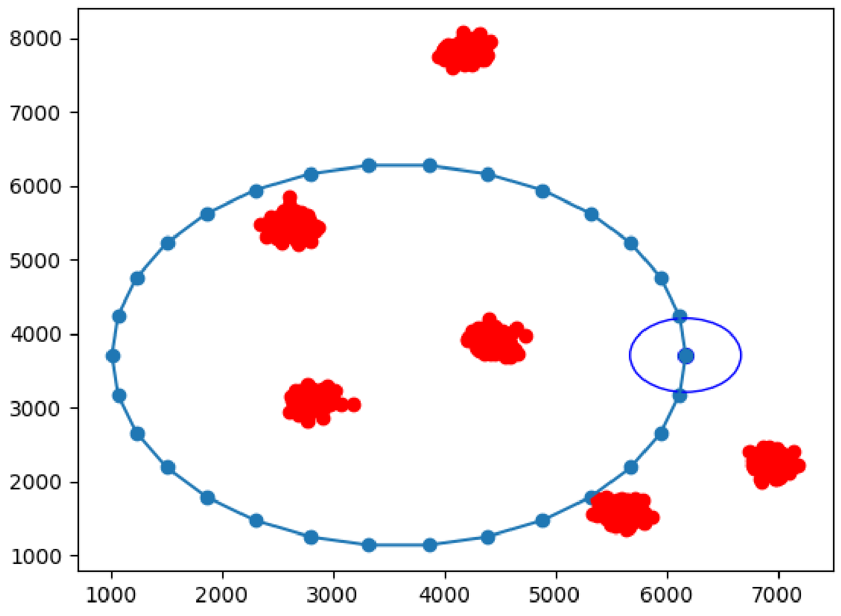


Figure 3.2: Example of the circular path (blue line) and clusters with nodes (red dots) location; the dark blue circle indicates the start of the trajectory.

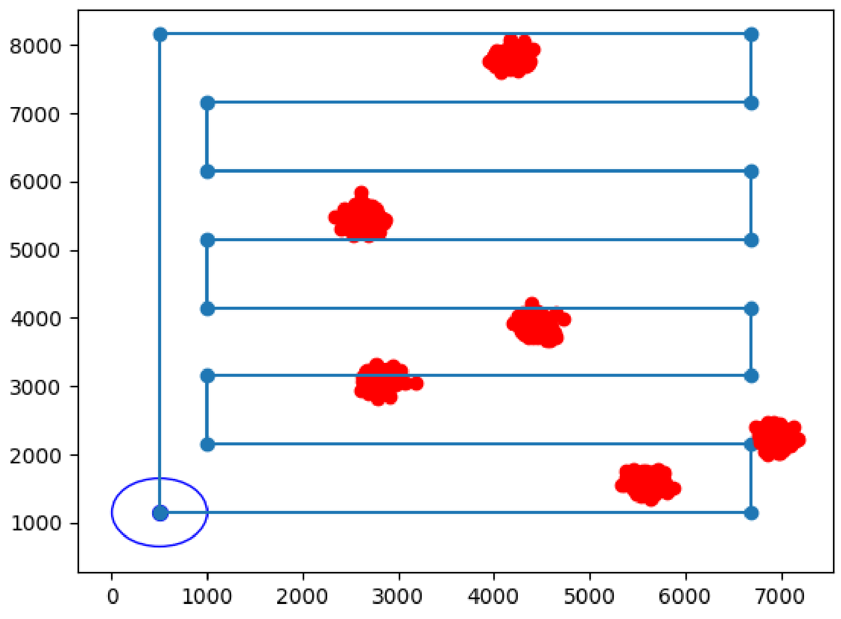


Figure 3.3: Example of Paparazzi path (blue line) and clusters with nodes (red dots) location; the dark blue circle indicates the start of the trajectory.

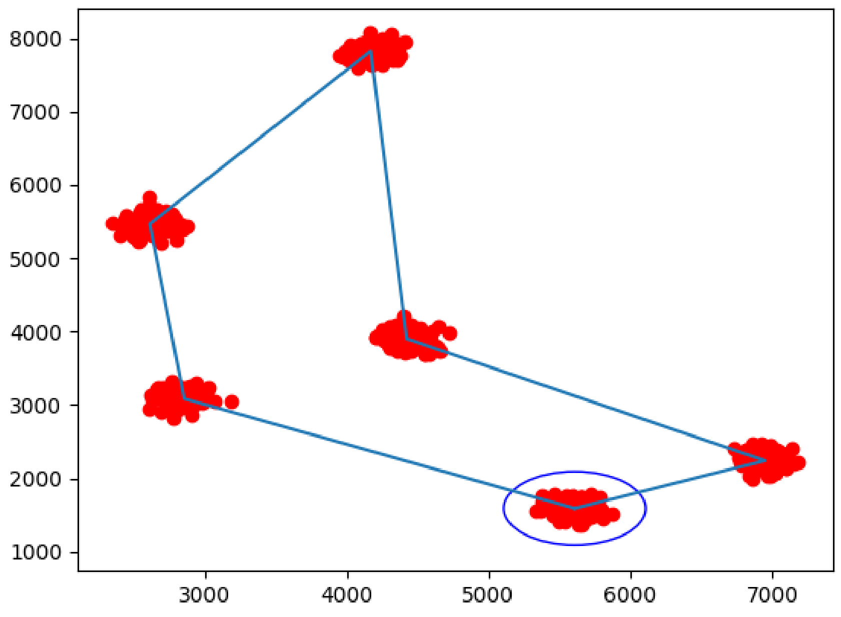


Figure 3.4: Example of TSP-driven path (blue line) and clusters with nodes (red dots) location; the dark blue circle indicates the start of the trajectory.

Table 3.2: Radio and Network Parameters

Parameter	Notation	Value	Parameter	Notation	Value
Area side [km]	L	10	Mean number of nodes per cluster	m	100
Intensity of parent points	λ_{p}	5	Locations' standard deviation	σ	100
Packet size [bits]	B	500	UL transmit power [dBm]	P_{tx}	14 dBm
Antennas loss [dBi]	A_{L}	2.5	Penetration loss [dB]	η	40
Environment constant	α	0.1581	Environment constant	β	9.6117
Noise power [W]	P_{N}	$30\ 10^{-17}$	Channel bandwidth [kHz]	B_c	180 KHz
Subcarrier spacing [kHz]	Δf	3.75	Available subcarriers	N_{SC}	48
Carrier frequency [MHz]	f_c	1747.5	RU duration [ms]	T_{RU}	32
MCS index	I_{MCS}	6			

As one can easily see, the performance of the considered network depends both on the UABS mobility pattern and the UABS NB-IoT cell configuration. In the following section, it is shown how the respective parameters (e.g., UABS height and speed, NB-IoT coverage class configuration and cluster dimension) affect the performance.

3.3 Numerical Results

In this section, NB-IoT network performance will be analyzed. As previously mentioned, a number of metrics have been jointly studied, being:

- the access rate, R_{acc};
- the percentage of nodes completely served, S_{suc};
- the mean latency perceived by nodes, $\Delta\tau_{\text{avg}}$;
- the network throughput, T_{net};
- the mean energy consumption of NB-IoT nodes, E.

These metrics are computed via the following formulas:

$$R_{\text{acc}} = \frac{n_{\text{suc}}}{n_{\text{att}}} \tag{3.6}$$

$$S_{\text{suc}} = \frac{n_{\text{suc}}}{n_{tot}} \cdot 100 \tag{3.7}$$

$$\Delta\tau_{\text{avg}} = \frac{\sum_{u=1}^{n_{\text{tot}}} (\tau_{u,\text{tx}} - \tau_{u,\text{start}})}{n_{\text{tot}}} \tag{3.8}$$

$$T_{\text{net}} = \sum_{u=1}^{n_{\text{tot}}} t_u \tag{3.9}$$

$$E = \frac{\sum_{u=1}^{n_{\text{tot}}} E_u}{n_{\text{tot}}} \tag{3.10}$$

The access rate R_{acc} in equation (3.6) denotes the fraction of the number of succeeded transmissions, n_{suc}, over the overall attempted by anyone of the IoT nodes, n_{att}. In this way, it can be analyzed how frequently a node u may get access to the channel. When the R_{acc} value gets close to zero, the number of access attempts has to increase before success. Equation (3.7) is used to identify how effective is the UABS service. In fact, S_{suc} counts the number of nodes successfully transmitting their traffic demand over the total number of present IoT nodes, n_{tot}, in percentage. Then, equation (3.8) computes the latency or delay for the node u spanning from the time in which the request is started, $\tau_{u,\text{start}}$, until when the transmission succeeded, $\tau_{u,\text{tx}}$. If a node is not able to transmit its packet, it is not considered in the average latency computation. Equation (3.9) computes the overall network throughput by summing the one of each node u, t_u. Finally, equation (3.10) averages the energy consumed by all nodes present in the service area, n_{tot}

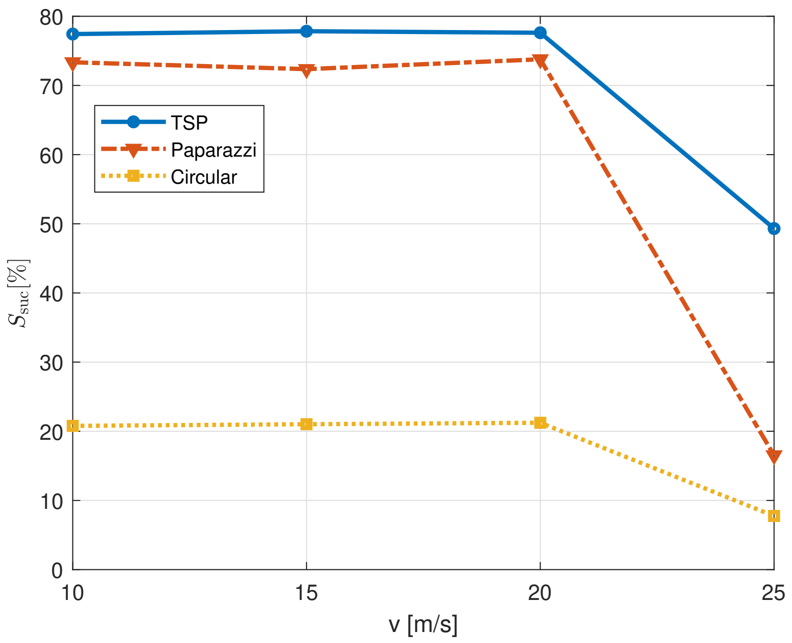


Figure 3.5: Percentage of served IoT nodes for different UABS speeds and trajectories with $h = 200$ m.

To start with, Figure 3.5 shows the percentage of IoT nodes served by the UABS, which means their traffic demand is completely fulfilled. Its value is represented while varying the UABS speed, v, and for the different trajectories. This picture has some relevant outcomes. First, the performance is not the same when varying speed; for each trajectory, it drops fairly below the 50% of served nodes. This is clearly related to the time interval in which the UABS is able to maintain a robust radio connection with each node. In fact, if the received power falls below the receiver sensitivity before the signalling is completed and the node is scheduled an uplink resource on the NPUSCH (for example, if the UABS has flown away too fast), it will not be able to successfully transmit its packet. This effect becomes relevant when discussing latency because the value of v creates a trade-off between low application delays and successful transmissions. At first glance, it is also evident that the TSP trajectory is able to serve a higher number of nodes for every UABS speed, making more robust what first proposed in [79]. In fact, this trajectory ensures the UABS gets in close vicinity with each cluster and each node, while minimizing the distance traveled. On the opposite, the circular trajectory decreases the percentage of successful transmissions by 50% with respect to TSP and Paparazzi ones, on average. The coverage extension and the time spent over each node do not allow in this case a sufficient service. The analysis of the other metrics would allow us to finally assess and discuss the final trajectory-dependent performance.

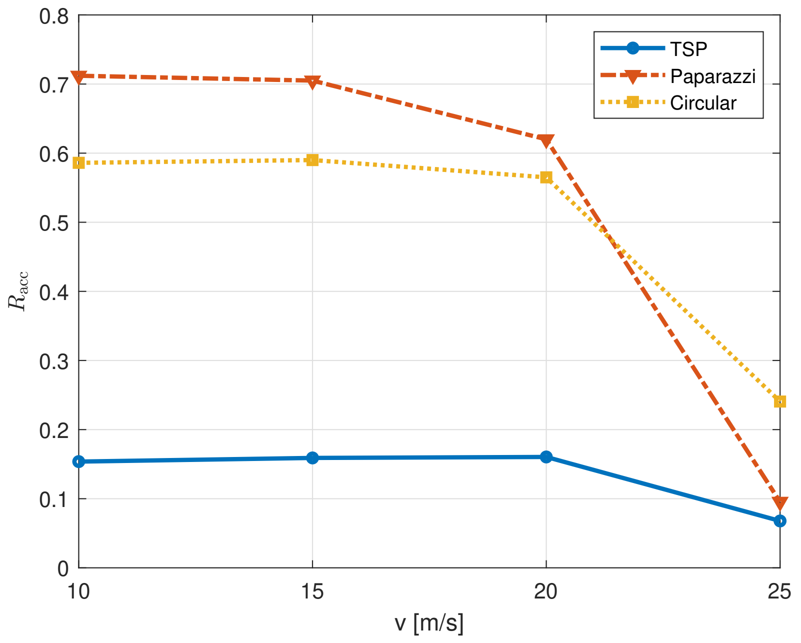


Figure 3.6: Access rate of IoT nodes for different UABS speeds and trajectories with $h = 200$ m.

Figure 3.6 represents the access rate, R_{acc}, again while varying the UABS speed for different trajectories. By looking at increasing speeds and for the same trajectory, the value of R_{acc} is dropping for each trajectory at 25 m/s, as it was for the service in Figure 3.5. With the UABS moving faster, the occasions to attempt channel access decrease. From equation (3.6), this metric evaluates together the number of overall attempts tried by nodes and, from these, the ones which are successful. This might explain why the Paparazzi and circular trajectories have higher access rates; these can be achieved if the total number of attempts is small (the denominator) with respect to other trajectories. If a lower number of nodes try to access at the same time to the channel (maybe for connectivity issues) the probability of successful transmission increases (see equation (3.6)).

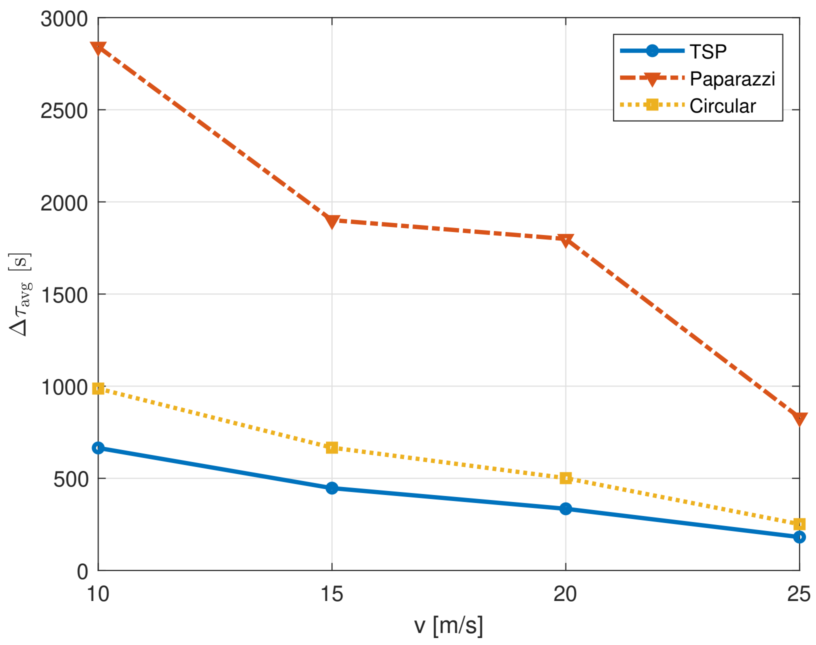


Figure 3.7: Perceived latency of IoT nodes for different UABS speeds and trajectories with $h = 200$ m.

Another figure of merit is the latency with which packets are transmitted (on average), $\Delta\tau_{\text{avg}}$, in Figure 3.7. As expected, the latency decreases with increasing speed for each trajectory; the NB-IoT nodes that can successfully transmit their packet are served faster because the UABS will reach them with a smaller delay. Furthermore, focusing on the latency of the different trajectories, it can be noticed that the TSP path performs better, followed by the circular one. In fact, the distances covered by these two trajectories are fairly smaller with respect to the other. On the contrary, for its characteristic of scanning the entire service area, the Paparazzi trajectory employs much larger delays than the other two.

The network throughput, T_{net}, that can be achieved has been also analyzed. As formulated in equation (3.9), these results will also depend on the ones of Figures 3.7 and 3.5 because of the dependence on service completion and latency in equation (3.4). Interestingly, three different trends with different UABS speeds, v, can be observed for the three different trajectories: i) the TSP shows a maximum in the curve, ii) the circular does not change significantly, while iii) the Paparazzi shows an improvement with increasing velocities. The maximum shown by the network throughput in the TSP trajectory corresponds exactly to the service-latency trade-off with UABS speed mentioned before. However, because of the low total service offered, the circular trajectory is hardly showing any maxima. Similarly, the maximum cannot be appreciated for the Paparazzi path, but the network throughput appears to be increasing with increasing speed. In fact, one can observe that at lower UABS speeds as 10 to 15 m/s, the average latency is so large (almost one or a

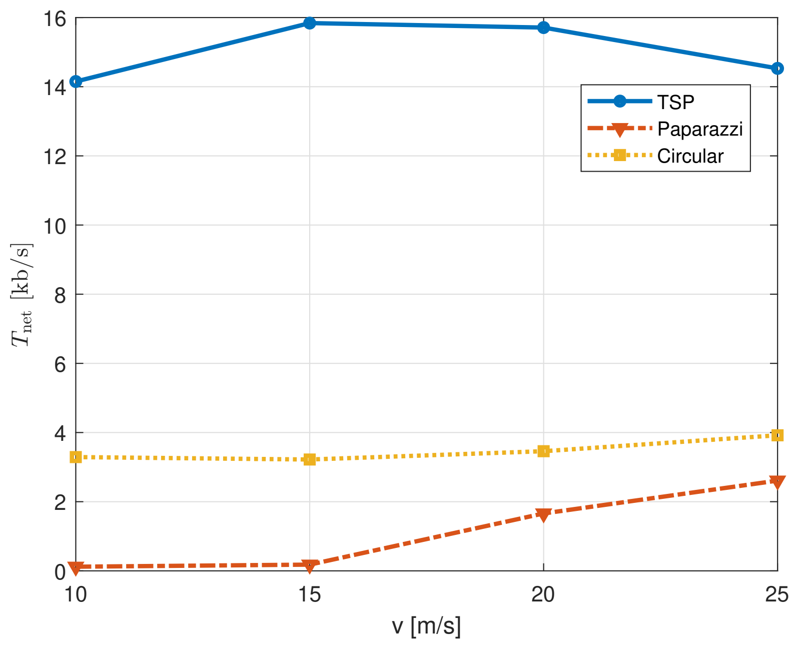


Figure 3.8: Network throughput for different UABS speeds and trajectories with $h = 200$ m.

half hour) that the maximum cannot occur for the chosen UABS speed range. As expected, the TSP trajectory shows a relevantly higher throughput with respect to the other two, up to 16 kb/s. Also, because of a larger average latency, the Paparazzi trajectory achieves an even lower throughput than the circular one (which had a worse percentage of served nodes). As last performance metric, the impact of trajectories on the average energy consumption, E, is reported in Figure 3.9.

To avoid cluttering, the impact of two different UABS heights, h, is shown for the same trajectory. Results, in this case, are provided considering the TSP path, being the trajectory showing better results in terms of almost all metrics.

In this case, the percentage of served NB-IoT nodes with varying UABS speed, v, is plotted in Figure 3.10. The behavior of the two curves with respect to speed is the same as already discussed in Figure 3.5. However, it can be seen a slight difference from before, that is a lesser sharp decrease in successful transmissions when the speed is 25 m/s. For lower speeds, the system performs better at lower UABS heights, h, while for higher speeds increasing heights improves the service. One might expect that since for higher altitudes the transmitter-receiver distance increases on average, the curve with the larger UABS height would perform always worse than the other (as it happens for values of v lower than or equal to 20 m/s). However, this does not consider the NB-IoT signalling messages procedure and timing. In fact, for increasing values of h, not only the transmitter-receiver distance gets higher, but also the coverage range of the UABS increases (given by the angle of incidence of the UABS with the ground, or elevation angle in [83, 84]). Consequently, it increases the time interval during which, on average, the NB-IoT node remains in the coverage range of the UABS. As shown in Figure 3.10, this effect can be more appreciated with increasing values of speed.

What has been discussed above is confirmed in Figure 3.11, showing the average energy consumption, E,

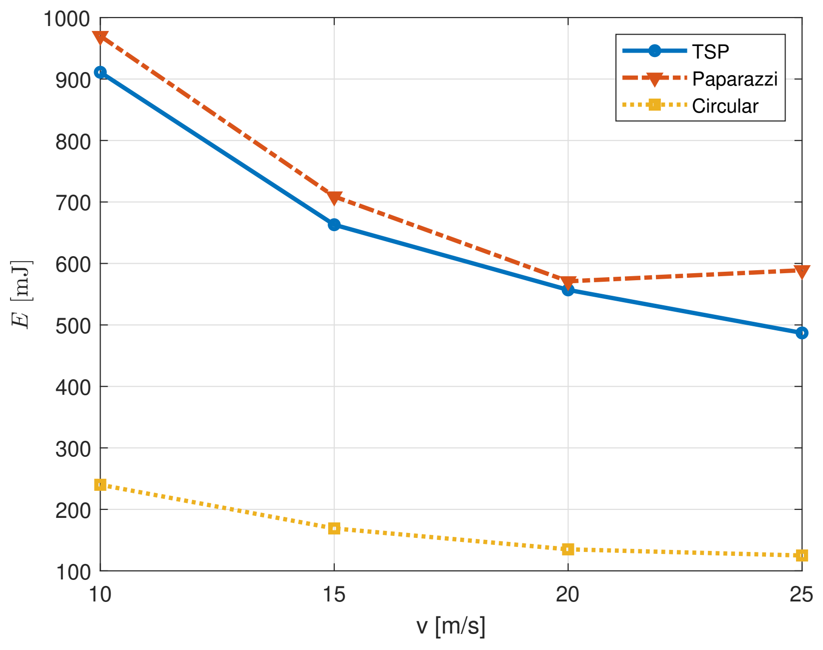


Figure 3.9: Average energy consumption of IoT nodes for different UABS speeds and trajectories with $h = 200$ m.

with varying UABS speeds and heights. Being an average, its value depends on the number of nodes able to reach the UABS connectivity and enter the RA (i.e. signalling) procedure. The curves' decreasing trend with speed is the same for the two h values and was already discussed for the TSP in Figure 3.9. A higher altitude lets a larger number of nodes be in connectivity with the UABS, therefore leading to the start of the RA procedure. This increases the overall energy consumption, regardless of successful transmission or not. With lower speeds, more nodes would be able to complete the signalling procedure, whether with higher speeds the energy consumed accounts only for the transmission of NB-IoT Msg1 and/or Msg2.

The analysis of the energy consumption can be validated by Figure 3.12. It represents the access rate, R_{acc}, for different UABS heights and speeds in the TSP trajectory. Here, the curve trend with respect the to speed, v, is the same, as for Figure 3.11, and the curve with the higher height, h, has a lower access rate. This means that a larger number of attempts does not correspond to the same number of successful transmissions. In fact, also due to the average larger transmitter-receiver distances with a higher-height UABS, it gets more difficult for nodes to complete the RA procedure.

Related to average latency, $\Delta\tau_{\text{avg}}$, one can observe a quite less significant impact. In Figure 3.13, the value of $\Delta\tau_{\text{avg}}$ is affected more by increasing UABS speeds rather than increasing UABS heights.

As evinced until now, the TSP trajectory performs better than the other in terms of successful transmission, latency and throughput, making it the most promising among the three for cellular IoT (e.g. NB-IoT) applications. Based on the provided results, one can extract the behavior of UAV networks with respect to speed, height and trajectory choice.

In the numerical evaluation, at first relevant factors, an operator must take into account when deploying

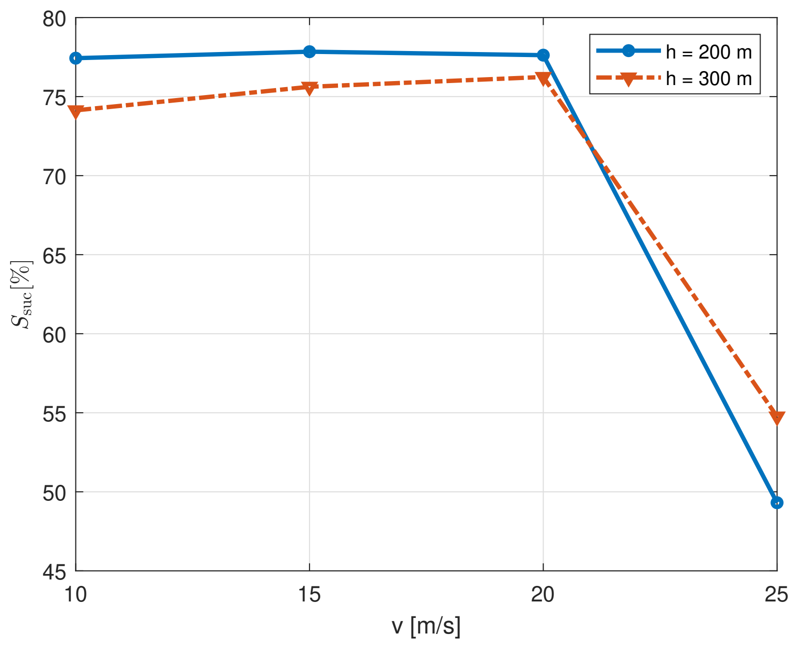


Figure 3.10: Percentage of served IoT nodes for different UABS speeds and heights in the TSP path.

this kind of system are identified. First, the speed undergoes a trade-off between average perceived latency and throughput from one side to the total number of successfully served nodes on the other. In this sense, the NB-IoT protocol plays a relevant role, since higher speeds do not allow the completion of the RA procedure and therefore the scheduling of uplink resources in the NPUSCH. Moreover, though a larger altitude would grant an increased connectivity range, it also increases the transmitter-receiver distance, which has again a negative impact on the successful completion of the RA procedure.

From the shown results, one can also infer conclusions on the trajectory selection. It can be stated that the circular trajectory, as would be for any path that travels the perimeter of a convex figure, is neither an effective trajectory in terms of latency nor a robust choice for the service of a massive number of IoT nodes. In fact, it is not able to follow the particular deployment of nodes for a given service area. On the other hand, a Paparazzi trajectory seems a good alternative, since it ensures to scan of the overall service area. This might be a fair solution if a mobile operator does not know a priori the location of nodes. However, because of the fact that this information is usually easy to retrieve and the Paparazzi trajectory becomes expensive in terms of energy consumption and latency, is probably not desirable. This confirms the expectations on the robustness of the TSP path.

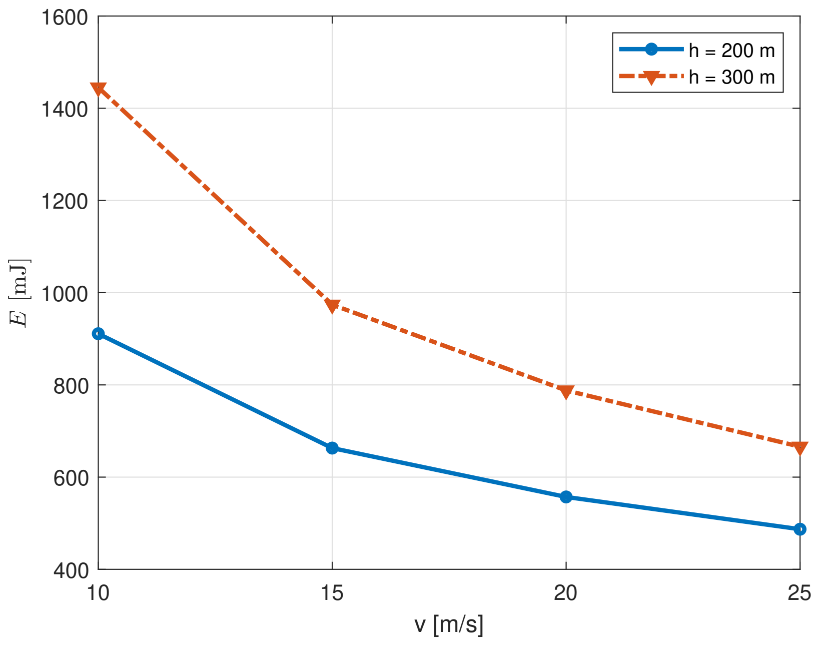


Figure 3.11: Average energy consumption of IoT nodes for different UABS speeds and heights in the TSP path.

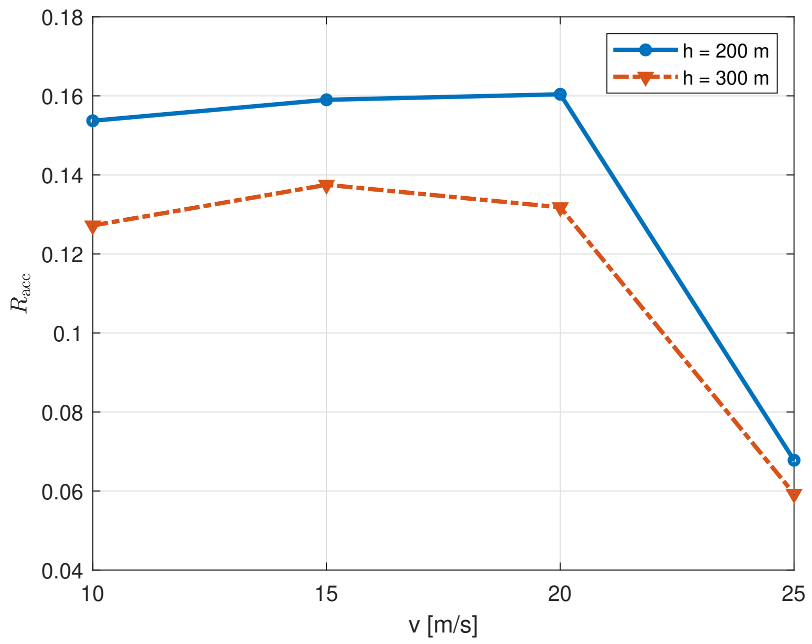


Figure 3.12: Access rate for different UABS speeds and heights in the TSP path.

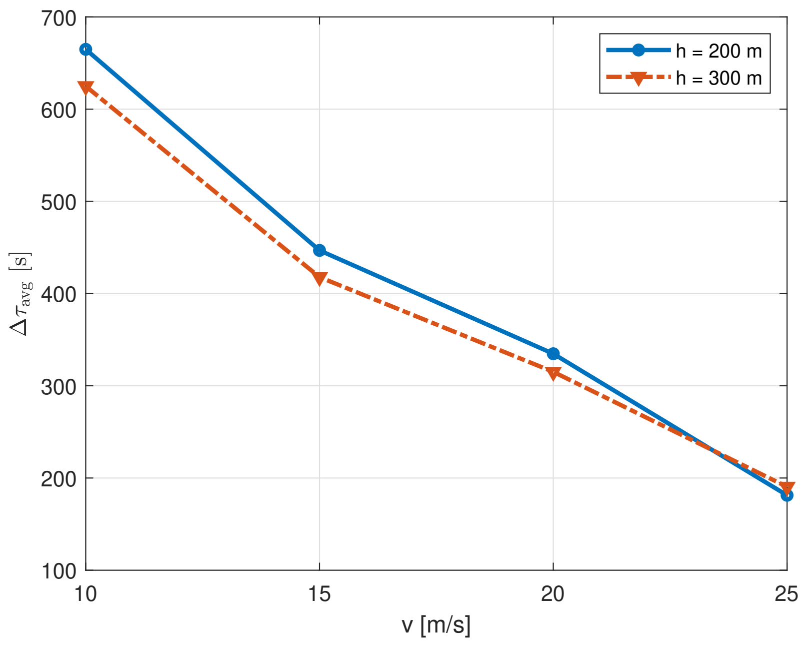


Figure 3.13: Perceived latency of IoT nodes for different UABS speeds and heights in the TSP path

3.4 Conclusions

This chapter proposed a thorough network performance evaluation of an UABS-aided NB-IoT network through detailed simulations. The different aspects of the NB-IoT protocol have been considered, including the signalling granting resources for uplink transmission and the different NB-IoT coverage classes. Then, the system performance have been evaluated jointly considering the service offered, access rate, average latency, network throughput and energy consumption metrics. UABS speed and height reveal a noticeable impact on the final performance, requiring a performance trade-off on the different metrics. Finally, it has been also observed the impact of different trajectory selections, with the trajectory given by the TSP solution being the most suitable for clustered environments.

Chapter 4

Trajectory Design for UAV Networks

4.1 Introduction

With the advent of 5G and beyond networks, novel advanced paradigms target network scalability and seamless communications. Therefore, as described in the previous chapter, the use of UABSs that may fly on-demand exactly when and where service is needed [101, 102], arises high interest and expectations. An important use case is offered by vehicular wireless networks, in which UABSs serve as relays between vehicular users and the network, enabling users to upload data collected by on-board sensors [103, 104, 105, 106, 107, 108]. Such user-generated data are collected by the network and then forwarded to other vehicles by means of BSs or Road Side Units (RSUs). Being able to offer stronger, possibly LoS, links to vehicles as compared to (static) ground BSs, UABSs can support high demanding V2X applications, such as advanced driving [10, 11] and extended sensing [12, 13], as specified by 3GPP [14].

In this context, the design of the trajectory assumes a fundamental role. Since the continuous variation in time of the environment constitutes a critical challenge, recent trends for UAV trajectory design show that the use of RL is particularly helpful [109]. Indeed, solving mathematical optimization models is not possible when a-priori input data is unavailable or requires too high complexity and computation time. To solve such problems, RL allows instead to learn in an environment with little prior information available. As it will be discussed later, RL balances the environment exploration done by an agent with the exploitation of acquired knowledge through time, aspects that allow learning the dynamics of a vehicular scenario.

The problem with RL-based solutions stems largely from the need to re-train an RL policy from scratch for any new environment, e.g., for a new traffic pattern of the ground users. Therefore, after exploring RL-based solutions, meta-learning [110] techniques will be addressed in this chapter in order to solve such a problem. Meta-learning is able to transfer information from previously experienced configurations to new conditions, reducing the time needed to optimize the UABS's policy. Standard meta-learning solutions for RL, also known as meta-RL, require the designer to have access to the simulators corresponding to all the previously encountered traffic conditions [111]. This may be practically impossible, or at least computationally prohibitive. Given these limitations of conventional meta-RL, the use of continual meta-RL via Continual Meta Policy Search (CoMPS) [112] will be addressed, which removes the need to revisit previous traffic conditions, and it operates online, acquiring new knowledge as new conditions are encountered.

The rest of the chapter is organized as follows. A review of the state of the art is provided in Section 4.2. The use of Deep Q-Network (DQN) and its extensions are explored in Section 4.3, whereas a comparison between discrete and continuous action spaces by means of DQN and Deep Deterministic Policy Gradient (DDPG) is reported in Section 4.4. The use of meta-learning is described in Section 4.5. Finally. Section 4.6 concludes the chapter.

4.2 Literature Overview

A key problem in UABS-based networks is designing algorithms that can efficiently optimize the trajectory of the UABS while ensuring target performance given some constraints [113, 114, 115]. As a means to find such trajectories, convex optimization approaches have been widely adopted under the assumption of fixed ground user locations [116]. In order to alleviate the impact of the simplifications required to apply convex optimization tools, RL-based solutions have been leveraged for the case of static ground users. As a matter of fact, most of the works consider fixed IoT nodes [117, 118]. Examples of DQN based algorithm are described in [119], where the goal of the algorithm developed by the authors is to get fresh data collection for time-critical IoT services. The use of UABSs for Search and Rescue (SAR) missions is considered in [120]. In this case, authors exploit a RSSI-based Q-learning in a GPS-denied indoor environment in order to detect users to be served. A fleet of UABSs is considered in [121], so a multi-agent RL approach is used in order to formulate the path planning problem taking into account multiple UABSs in the scenario. Such an issue is again addressed with a DQN algorithm by feeding the neural network within the model with global maps of the environment.

The problem reaches a new level of complexity when the target of UAV services is moving. Authors in [122] adopt Q-learning for the trajectory control part, and [123] employs Q-learning for 3D trajectories in a multi-UABS environment where users are roaming. [124] studies the optimal positions of UABSs in a sparse highway with a variation of the Deep Independent Q-learning for multiple agents. However, the communication range of drones is here considered fixed, therefore ignoring spatiotemporal channel fluctuations. Moving users are considered in [125], where authors propose a RL approach for multiple UABSs trajectory planning to serve vehicles considering the UABSs energy consumption. However, in this case, the trajectory planning turns out to be very simple since a highway scenario is considered. In [126], authors take into account mobility and they present a scenario where the path planning is based on the initial position as well as the prediction of the users' next location. To do so, they exploit Q-Learning for the trajectory design, which is fed with the prediction of users' movement generated by an echo state network algorithm that exploits GPS coordinates supposed to be available by mining data from social networks. An emergency scenario is considered in [127], where authors propose a Q-Learning-based algorithm to find the best positions of multiple UABSs in order to maximize the number of covered users moving in an area, rather than learning a trajectory to follow them in the environment. In [128], authors study the optimal positioning of a relaying UABS which collects data from one vehicle user while avoiding interfering with nearby V2V communications, while in [104] the optimization is performed taking into account the energy efficiency, the latency and the backhaul link capacity. Both works consider a deployment problem, without taking into account a complete flying trajectory. Works like [129, 130, 124] consider highway scenarios where the UABS movement is limited to one direction, while in [131, 132] authors deeply simplify the city layout, by discretizing it into road segments. Nonetheless, all these works provide as input to the system the exact vehicles' positions.

Besides standard RL-based techniques, conventional meta-learning was previously considered for UABS trajectory optimization in [133] by assuming that the ground users are static and have known locations. The same authors in [134] extended their previous work by considering multiple UABSs. Unlike these previous works, in the following, traffic conditions characterized by vehicular users with a priori unknown locations will be considered, moving beyond conventional meta-RL by accounting for the constraint that simulators for previous traffic configurations cannot be revisited.

4.3 Deep Q-Learning-based Trajectory Design

At first, the use of DQN and its extensions (DDQN and Double Dueling Deep Q Network (3DQN)) considering a discrete action space will be considered.

4.3.1 System Model

One UABS, u, and a set of vehicles denoted as GUEs, g, belonging to the set $\mathcal{G}$, are present in the scenario of interest. The UABS is equipped with a radio antenna system enabling beamforming for the mmWave frequency bands; its position is given by $[x_t, y_t, h]$, with constant altitude h, at time instant t. GUEs implement an *extended sensing* application, according to which, each vehicle wants to exchange data gathered through local sensors or video with other vehicles nearby [14]. To do this each GUE sends V2X messages (e.g., Cooperative Awareness Message (CAM) and Collective Perception Message (CPM)), to the network, every t_{msg} seconds, with $t_{msg} \in [0.1, 1]$ seconds. In the following, it is assumed $t_{msg} = 1$ s. Cars are moving in an urban environment considering a map based on an area of the city of Bologna (Italy). This city map is characterized by many possible paths each vehicle can take; their generation is based on Simulation of Urban Mobility (SUMO), an open source traffic simulator [135] (an example is provided in Figure 4.1). In order to improve the system performance, the UABS is expected to identify the most suitable trajectory to maximize network service.

Since the RL algorithm used for this work is based on a discrete set of actions, as it will be discussed in Section 4.3.4, the UABS can move only in predetermined directions. To model this behavior, the map (1460 x 760 m^2) is divided into a squared grid world with vertices corresponding to the possible positions. The number of positions available depends on the UABS speed, so the slower the UABS, the more granular the grid will be (in simulations, there are 2628, 1176, and 666 possible positions for the slowest, the medium and the fastest settings, respectively).

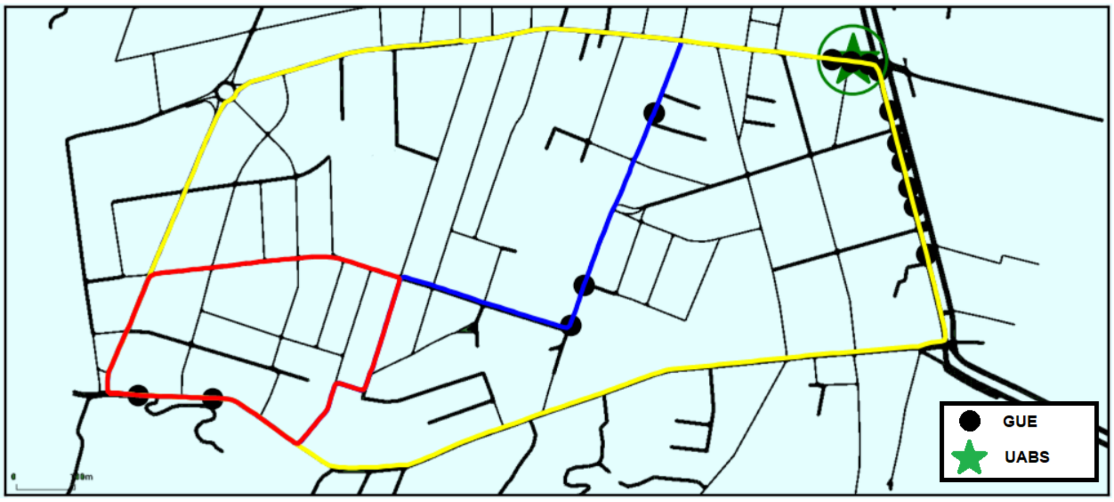


Figure 4.1: Example: three possible paths (yellow, blue, red) taken by GUEs.

4.3.2 Channel Model

A mmWave communication system working at a carrier frequency $f_c = 30$ GHz is assumed. Accordingly, the channel model adopted is provided by the 3GPP in [136], considering, specifically, the Urban Macro (UMa) scenario (more details in Chapter 7.4 of [136]). The model introduces probability for a link to be in LoS conditions (see Table 7.4.2-1 of [136]), p_{L}, that depends on the projected 2D terminal distance from the UABS, d_{2D} and user terminal height, h_{UT}. Therefore, the channel losses, l_{L} and l_{N} for LoS and NLoS, respectively, depend also on other parameters characterizing the propagation link, such as the terminal height, h_{UT}, the UABS height, h, and the 3D distance between the UABS and the terminal. Slow fluctuations due to shadowing are considered with parameters $\sigma_{\mathrm{L}} = 4$ and $\sigma_{\mathrm{N}} = 6$ (dB), respectively. Consequently, each link in the scenario has a channel loss (in dB scale) $PL = l_{\mathrm{L}} + \sigma_{\mathrm{L}}^*$ with probability p_{L} or $PL = l_{\mathrm{N}} + \sigma_{\mathrm{N}}^*$ with a probability $1 - p_{\mathrm{L}}$, where σ_{L}^* and σ_{N}^* are the shadowing samples taken from the Gaussian distributions $\sigma_{\mathrm{L}}^* \sim \mathcal{N}(0, \sigma_{\mathrm{L}}^2)$ and $\sigma_{\mathrm{N}}^* \sim \mathcal{N}(0, \sigma_{\mathrm{N}}^2)$, respectively.

Finally, the SNR in dB is defined as:

$$\mathrm{SNR} = [P_{\mathrm{tx}} + G_{\mathrm{tx}} + G_{\mathrm{rx}} - PL] - P_{\mathrm{noise}} \tag{4.1}$$

where P_{tx} is the transmitted power in dBm, G_{tx} and G_{rx} represent the gain in transmission and reception, respectively, in dB and P_{noise} is the noise power.

4.3.3 Beamforming

Tackling a 6G network scenario, it is assumed transmissions take advantage of beamforming techniques. In particular, the UABS is using a fixed Grid of Beams, operating on a set of $N_{\mathrm{beam}} = 9$ resulting on the ground as circular spots and arranged in a 3x3 grid. $\mathbf{\Phi}$ denotes the solid angle deriving from the radiation pattern and $\alpha_{beam} \approx \mathbf{\Phi}/N_{\mathrm{beam}}$ as the solid angle of a beam. Consequently, the maximum gain G can be expressed as [137]:

$$G = \frac{41000}{(\alpha_{beam}\frac{360}{2\pi})^2} \tag{4.2}$$

For the sake of simplicity, the radiation pattern is assumed ideal, with gain G inside the angle α_{beam}, and zero outside. ϕ denotes the 2D angle of the UABS's vertical plane, from which it is inferred the field of view of the UABS projected on the ground.

4.3.4 Problem Definition

Time t is discretized into steps of duration t_{msg}. According to standard RL problems, an agent interacts with an environment during an episode, lasting T steps. At each step, starting from state s_t, belonging to the state space, $\mathcal{S}$, the agent chooses an action, a_t, from a set of possible actions, $\mathcal{A}$, according to its policy π. After selecting such action, the agent moves to state $s_{t+1} \in \mathcal{S}$ and receives a scalar reward $r_t = r(s_t, a_t)$ based on a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

To this end, some definitions are introduced:

- an *agent* as the UABS, u, whose target is to design the trajectory which maximizes a reward function;
- a *state* in the state space that consists of the agent location and number of vehicles under each beam. The agent position is expressed in (x, y) coordinates at the relative time instant t. The UABS is able to get information related to the number of GUEs under its field of view, therefore a vector $\mathbf{b}_t$, in which each element $b_{t,i}$ is equal to the number of GUEs under the i-th beam at time instant t, is defined. Therefore, a state is defined as $s_t = \{x_t, y_t, \mathbf{b}_t\}$;
- an *action* of the agent as a movement on the map in one of 8 possible directions or hovering. Therefore, the Action Space is defined as $\mathcal{A} = \{0, \leftarrow, \uparrow, \rightarrow, \downarrow, \nwarrow, \nearrow, \searrow, \swarrow\}$, where 0 represents the decision to stay still;
- the *reward* measures the benefit of selecting a specific action a while being in a state s. Since the agent aims at maximizing the number of served GUEs, it is computed as the sum throughput of the vehicles seen by the UABS at time instant t. Therefore, it is defined as

  $$r_t = \sum_{g \in \mathcal{G}} r_t^{(g)} := \sum_{g \in \mathcal{G}} B_{\text{ch}} \log(1 + \text{SNR}_t^{(g)}), \tag{4.3}$$

  where $r_t^{(g)}$ indicates the capacity of the $g - th$ GUE at time t for the UABS in state s_t and action a_t, and where B_{ch} is the channel bandwidth.

4.3.5 Q-Learning

Q-Learning is a model-free RL algorithm that consists of an agent interacting with an environment in order to learn and optimize its behavior. In particular, it aims at iteratively improving the state-action value function, or Q-function, which represents an expectation of the discounted cumulative future reward R_t from the current state s_t up to the last step T:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}[R_t | s_t = s, a_t = a, \pi] \tag{4.4}$$

where $\mathbb{E}_{\pi}$ is the expected value under policy π and with R_t given by:

$$R_t = \sum_{i=t}^{T} \gamma^{i-t} r(s_t, a_t) \tag{4.5}$$

where $\gamma \in [0,1]$ represents the discount factor, which balances the importance of the immediate and the future reward.

Given a transition $< s_t, a_t, r_t, s_{t+1} >$, $Q_{(s,a)}$ can be expressed by the Bellman equation in terms of Q-value of the next state s_{t+1}:

$$Q(s_t, a_t) = r_t + \gamma \max_a Q(s_{t+1}, a) \tag{4.6}$$

In other words, it is possible to express the Q-value as the sum of the immediate reward and the discounted future reward of the state that follows, without the need of calculating each value as the sum of the expected cumulative reward.

In its simplest form, Q-Learning exploits the Q-Table, a look-up table in which the Q-value for each state-action pair is stored and regularly updated. The agent chooses an action based on an epsilon greedy policy [138], thus the chosen action may be random with probability ϵ or it can be the action with the highest Q-Value with probability 1-ϵ. Each time an action is chosen, the updated Q-Value is computed as:

$$Q^{new}(s_t, a_t) = Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{(t+1)}, a) - Q(s_t, a_t)) \tag{4.7}$$

where α represents the learning rate, which determines how new information overrides old information.

4.3.6 Deep Q-Learning

The main drawback of Q-Learning is that in case of high dimensional state or action spaces, the Q-Table would require too much time to be created, as well as too much space to be stored. For these reasons, DQN has been introduced in order to represent the policy π or other learned functions as a deep neural network which, taking the state as input, estimates the Q-values for all the different actions an agent may take.

Q-values can be any real values, which makes the problem a regression task, thus optimized with a function of the error loss between the predicted and the true values, estimated using equation 4.6. Both true and predicted target values are used to calculate the loss and consequently to update the neural network weights, leading to a huge correlation between target values and network weights. To avoid convergence issues and make the training more stable, two different networks are used:

– the *action network*, with parameters θ, represents the predicted value and it is used to select the action the agent takes and it is updated every n steps;

– the *target network*, with parameters θ^{-}, is a clone of the *action network* used only to compute the true value and it is updated every $m >> n$ steps by copying the action network's weights.

Therefore, the loss is computed as:

$$L = f(r_t + \gamma \max_a Q(s_{t+1}, a, \theta^{-}) - Q(s_t, a_t, \theta)) \tag{4.8}$$

As f, in the following the Huber Loss [139] has been chosen.

4.3.7 Double Deep Q-Learning

In [140], authors claim that the standard Q-Learning algorithm is known to overestimate action values under certain conditions. In order to prevent such overestimation, a DDQN algorithm is presented. The idea behind this algorithm is to decouple the selection from the evaluation, by exploiting the *action network* to select the action, whereas the corresponding Q-Value is estimated using the *target network*.

Therefore, the loss function changes as:

$$L = f(r_t + \gamma Q(s_{t+1}, \arg\max_a Q(s_{t+1}, a, \theta), \theta^{-}) - Q(s_t, a_t, \theta)) \tag{4.9}$$

4.3.8 Dueling Deep Q-Learning

Given the policy π, it is possible to define the state-value function $V^{\pi}_{(s)} = \mathbb{E}_{a\sim\pi(s)}[Q^{\pi}_{(s,a)}]$, which represents the expected total reward following policy π starting from state s.

Consequently, it is possible to define the Advantage function as

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s) \tag{4.10}$$

which represents how advantageous would be an action with respect to the others at the given state following policy π.

Dueling DQN [141] introduction is motivated by the fact that it is unnecessary to know the outcome of each action at each time step, so, exploiting two separate streams (which are recombined in order to generate the Q-values) for the estimation of the State values and the Advantages, the dueling architecture can learn separately which states are important from which action is better to select. This also provides the possibility to integrate it in the classical DQN schemes, whose functioning remains the same, since no changes to inputs and outputs are made. In the following, the term 3DQN refers to a Dueling Double DQN scheme.

4.3.9 Numerical Results

Parameters set in simulations are reported in Table 4.1. During training, the Buffer Size is set to N $= 500000$ and the Batch Size K $= 128$.

Results are presented in terms of total reward, which is the sum of the reward obtained by the agent at each step, that is $R = \sum_{t=0}^{T} r_t$, where T is the length of one episode and r_t has been defined in equation (4.3).

Table 4.1: DQN Simulation Parameters

Parameter	**Notation**	**Value**
Number of GUEs	N_g	15
Average GUE speed [m/s]	v_g	10
GUE transmit power [dBm]	$P_{\text{tx},g}$	20
Carrier frequency [GHz]	f_c	30
Channel bandwidth [MHz]	B_{ch}	1.44
Effective noise power [dBm]	P_{noise}	-106.4
Episode Length [s]	T	380
Learning Rate	α	0.001
Discount Factor	γ	0.8
Buffer Size	N	500k
Batch Size	K	128
Action Network Update	n	1
Target Network Update	m	500

Figure 4.2 shows the total reward as a function of the number of episodes for different algorithms presented in Section 4.3.4, when setting v_u=20 m/s and $\phi = 140°$. Note that, by changing ϕ, the antenna gain at the UABS will change according to equation (4.2). It can be clearly seen that the dueling architecture offers more advantages since the dueling DDQN and the dueling DQN perform better with respect to the other architectures. The reason behind this may be explained by the fact that the dueling architecture is able to learn which states are important regardless of the action to take, which means that the agent is able to detect which are the most important locations (i.e., states) to reach first and then optimize how to reach them. Surprisingly, the double architecture alone does not provide good results, even worse than the standard DQN. This may happen due to the fact that the Q-value, in the double architecture, is estimated exploiting the *target network*, which is updated slower w.r.t. the *action network*, as explained in Section 4.3.6. This penalizes the knowledge the agent may have acquired and which is not used until the next *target network* update. In addition, the figure shows how the introduction of the beam information $\mathbf{b}_t$ into the state definition gives a huge boost in terms of training time, allowing the algorithms to converge after 1000 episodes. On the other hand, the removal of such information, which changes the state definition to $s_t = \{x_t, y_t\}$, increases the time needed for the algorithms to converge up to 10000 episodes, since the agent has fewer data at its disposal. Finally, the horizontal line shows the performance of a UABS taking decisions without using any RL-based solution, but only selecting each action by going towards the direction of the beam with the highest number of vehicles inside at each time step. This allows us to compare the RL architectures with a benchmark, where the UABS is exploiting the information available but is not actually learning how to use it properly. As it can be clearly seen, the benchmark performance is very poor w.r.t. those achieved when RL is used.

On overall, the 3DQN is proved to be the best solution, since it converges faster w.r.t. the standard dueling

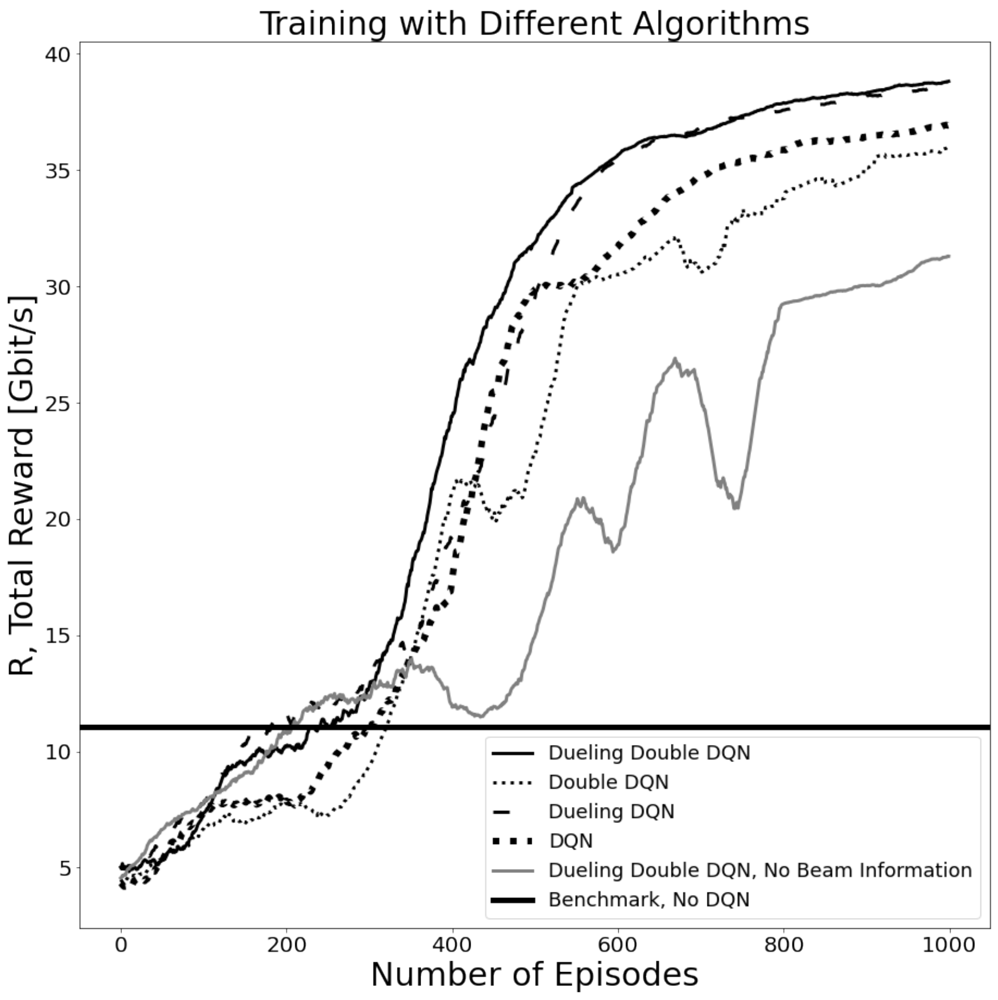


Figure 4.2: Comparison between different algorithms for v_u=20 m/s and $\phi = 140°$.

architecture. Therefore, the next results will be presented considering only such an algorithm.

Figure 4.3 shows the total reward as a function of the number of episodes for different UABS speed, v_u, having set $\phi = 140°$. Results are comparable in terms of total reward obtained at the end of the training, even though higher speeds offer faster convergence mainly because the agent is able to explore the entire map faster. In addition, it can be noticed that choosing such speeds does not worsen the performance, even if they are higher w.r.t. the average GUEs speed.

Finally, Figure 4.4 depicts the total reward as a function of the number of episodes for three different overall fields of view of the UABS, ϕ, when setting v_u=20 m/s. As can be seen, a larger field of view, and therefore a larger angle of aperture per beam, provides better results mainly because more GUEs are seen by the UABS. In addition, the beam angle of the aperture helps also in the detection of the GUE movement direction, resulting in faster learning.

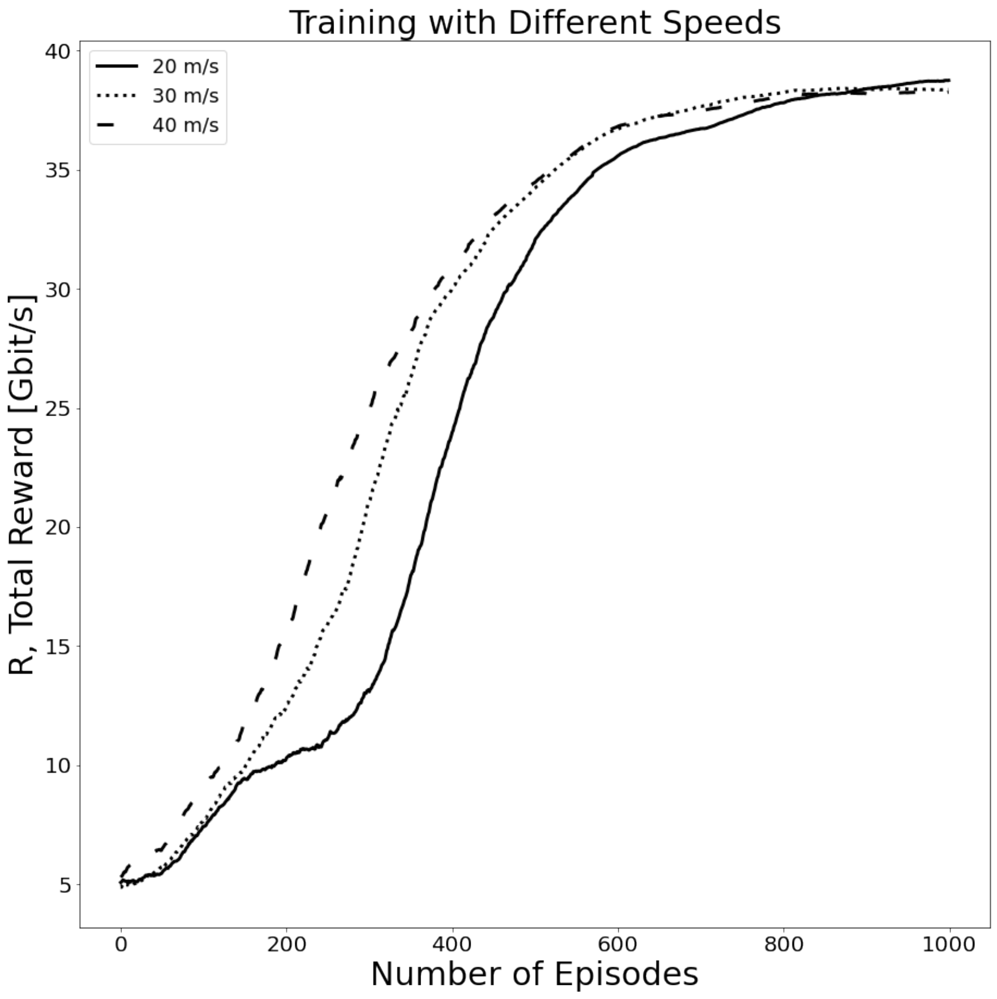


Figure 4.3: Comparison between different UABS speeds with Dueling DDQN for $\phi = 140°$.

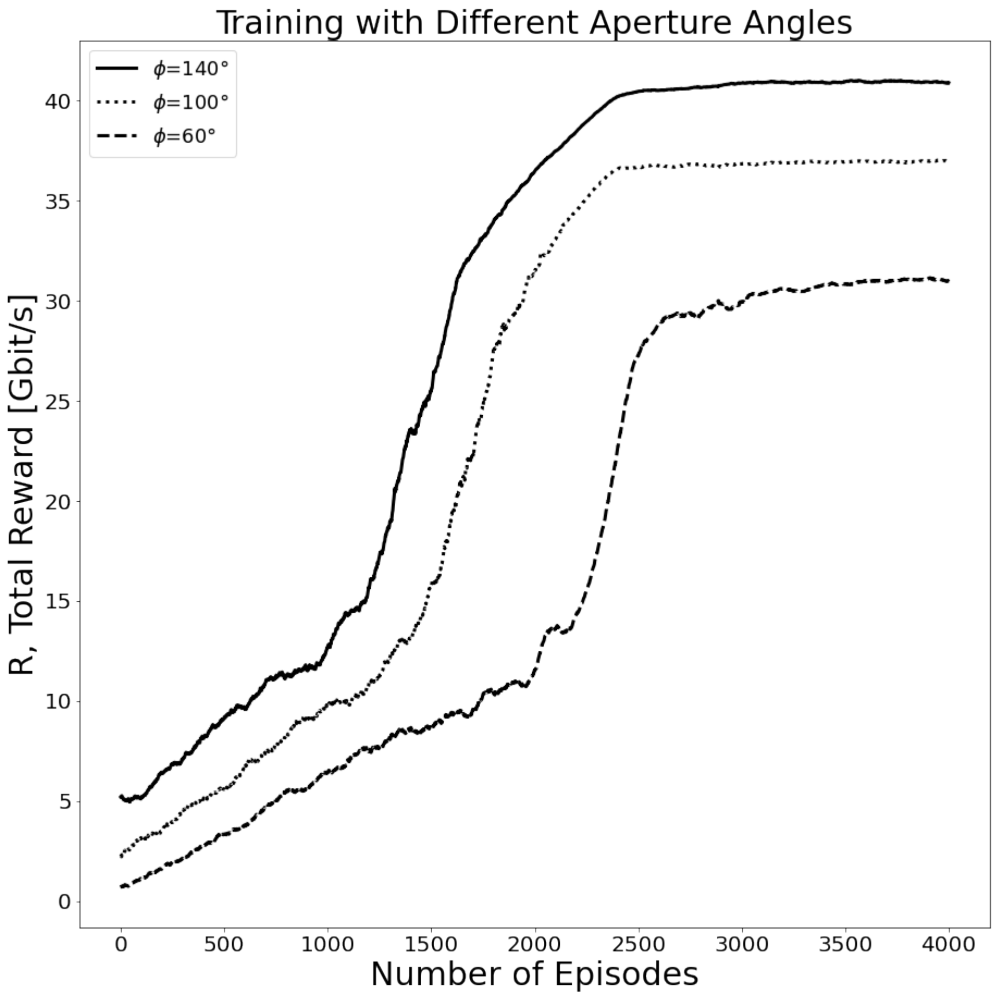


Figure 4.4: Comparison between different UABS aperture angles with Dueling DDQN for v_u=20 m/s.

4.4 Comparing Dueling Double Deep Q-Network and Deep Deterministic Policy Gradient

After describing a 3DQN-based solution, a comparison with the DDPG algorithm will be described in the following in order to understand the advantages and/or limitations of the use of such algorithm as well as the exploitation of a continuous action space.

4.4.1 System Model

The scenario considered is the same one reported in Section 4.3.1. In the same way, the channel model and the beamforming approach are the ones presented in Section 4.3.2 and Section 4.3.3, respectively.

One major difference is the introduction of a priority-based mechanism in the service provided to GUEs. As a matter of fact, given the stringent requirements of vehicular communications [14], it is important also to consider the case when the UABS offer continuous service that guarantees radio access for multiple contiguous time instants during a time interval with an application dependent duration. For these reasons, a GUE g is assumed to be *served* at time t if a target SNR, SNR_{th}, is met:

$$y_t^{(g)} = \begin{cases} 1 & \text{if } \mathrm{SNR}_t^{(g)} > SNR_{th} \\ 0 & \text{otherwise} \end{cases} \tag{4.11}$$

The concept of the user *satisfaction* then is introduced to define a target Quality of Experience (QoE). Suppose that GUE $g \in G$ has a service time window W of duration T_w, composed of N intervals of duration Δt=1s. Being N_s the number of intervals in W in which g has been *served*, the GUE is *satisfied* in that service window if $N_s \geq \hat{N}_s$, where $\hat{N}_s$ is a threshold that depends on the target application requirements. To keep track of the service history at each time instant t and for each GUE g, a priority $p_t^{(g)}$ is introduced and assigned to each GUE. It corresponds to the number of times the vehicle g has been served during the previous N time intervals.

4.4.2 Problem Definition

Again, the RL problem definition is the same provided in Section 4.3.4. However, to address the priority mechanism previously described and to consider a continuous action space in the case of DDPG, the following changes have been introduced:

- $\mathcal{S}$ is represented by $(x_t, y_t, \mathbf{b}_t)$, where (x_t, y_t) is the position of the UABS on the xy plane and $\mathbf{b}_t$ is now representing a vector of N_{beam} elements where each element $b_{i,t}$ corresponds to the sum of the priorities $p_t^{(g)}$ of the GUEs $g \in G$ inside the i-th beam at time instant t;
- $\mathcal{A}$ refers to the possible movement directions of the UABS, measured as the angle of rotation with respect to a global reference system, and its current speed of flight in m/s. The discrete set used in 3DQN is

limited to actions $a \in \mathcal{A}$, where $\mathcal{A} = \mathcal{M}_D \times \mathcal{V}_D$; with $\mathcal{M}_D = \{-135°, -90°, -45°, 0°, 45°, 90°, 135°, 180°\}$, the limited directions set, and $\mathcal{V}_D = \{10, 20, 30, 40\}$ m/s the set of possible speeds. In the case of DDPG, the agent may choose actions $a \in \mathcal{A}$ with $\mathcal{A} = \mathcal{M}_C \times \mathcal{V}_C$, where $\mathcal{M}_C$ and $\mathcal{V}_C$ are continuously defined in $[-180°, +180°]$ and $[10, 40]$ m/s, respectively;

– the reward R is represented by the instantaneous weighted normalized sum rate considering some penalties factors: $r_t = R_t - (\lambda_b + \lambda_n)$,

$$r_t = R_t - (\lambda_b + \lambda_n), \tag{4.12}$$

$$R_t = \sum_{g \in G} r_t^{(g)} p_t^{(g)} = \sum_{g \in G} \frac{B_{\text{ch}} log_2(1 + SNR_t^{(g)})}{R_{\text{best}}} p_t^{(g)}. \tag{4.13}$$

Where $SNR_t^{(g)}$ is the signal-to-noise ratio of the g-th link measured at the UABS, $p_t^{(g)}$ is the current priority of the g-th GUE, B_{ch} is the channel bandwidth and R_{best} is a normalization factor, calculated as the highest rate achieved by a counterfeit placed GUE exactly underneath the UABS at closest distance $h^{(u)}$, thus assuming it suffers from the lowest possible path loss according to the channel model implemented. Normalizing the rate helps to define penalties that work on a known range of values and improve the training performance. The penalty λ_b is assigned in case the UABS tries to go outside the borders of the area considered, while the penalty λ_n is assigned if $R_t = 0$.

For the implementation of the UABS's trajectory design based on a discrete action space, the 3DQN algorithm was used, while for the continuous action space, the DDPG algorithm has been exploited.

3DQN

To solve the trajectory design problem using a discrete set of actions, the algorithm 3DQN was used, as presented in Section 4.3.

DDPG

DDPG was specifically designed to operate with a continuous action space [142]. It inherits from DQN the use of a replay buffer D to store experiences and the exploitation of delayed target networks to stabilize the training. Differently from the previous algorithm, DDPG uses two different Deep Neural Networks (DNNs): the actor network, with parameters μ, and the critic network, with parameters ψ. The critic network estimates the Q-values for all states and actions, while the objective of the actor network is to parameterize a deterministic policy $\pi_\mu(s|\mu) := \arg\max_a Q(s, a|\mu)$, which chooses the best agent's action while in state s. The idea of the algorithm is to use Q-values estimates, provided by the critic, as a score for the performance of the deterministic policy π_μ. By maximizing such a score the policy will approach the optimal one.

4.4.3 Numerical Results

In this section, results will be presented in terms of total reward, which is obtained by summing the rewards obtained by the agent, that is $R = \sum_{t=0}^{T} r_t$, where T is the length of one episode and r_t has been defined in equation (4.12). As a second performance metric, the percentage of satisfied users, P_g, is obtained as the ratio between the number of satisfied service windows for each GUE and the total number of service windows requested by the same GUE to the UABS, averaged among all GUEs. Parameters used during the simulation are listed in Table 4.2.

Parameter	**Notation**	**Value**
UABS height [m]	$h^{(u)}$	100
UABS speed [m/s]	$v^{(u)}$	[10-40]
Number of GUEs	$\|\|G\|\|$	15
GUEs speed [m/s]	$v_m^{(g)}$	12
UABS aperture angle	$\phi^{(u)}$	100°
Service time window duration [s]	T_w	10
Interval duration [s]	δt_{packet}	1
Transmit power [dBm]	P_{tx}	14
Noise power [dBm]	P_{noise}	-106.4
Transmitting antenna gain [dB]	G_{tx}	0
Receiving antenna gain [db]	G_{rx}	24
Carrier frequency [GHz]	f_c	30
Channel bandwidth [MHz]	B_{ch}	1.44 MHz
Reward normalization factor	R_{best}	20.00
Episode duration [s]	T	380
Reward penalties	λ_b, λ_n	0.01
Number of episodes	N_{ep}	5000
Discount factor	γ	0.9

Table 4.2: DQN/DDPG Simulation Parameters

Figure 4.5 shows the comparison among different DNN input layer architectures, Raw and One Hot Encoding (OHE). Raw encodes for each state variable its current value onto a neuron. OHE, instead, uses a set of neurons for each state variable so that its current value is encoded as a single neuron in the set that is "on", with value 1, while the others are "off", with value 0. The plot shows the total reward as a function of the number of episodes, using a moving average with window 20. To avoid the problem of overfitting, during each episode, vehicles randomly choose roads to travel following a given traffic distribution generated by SUMO. During training, the UABS may choose random directions to explore the environment and gain new experiences or it can take the estimated best action (i.e., the action providing the highest reward according to its current knowledge) following the current policy. The plot is shown in terms of evaluation episodes,

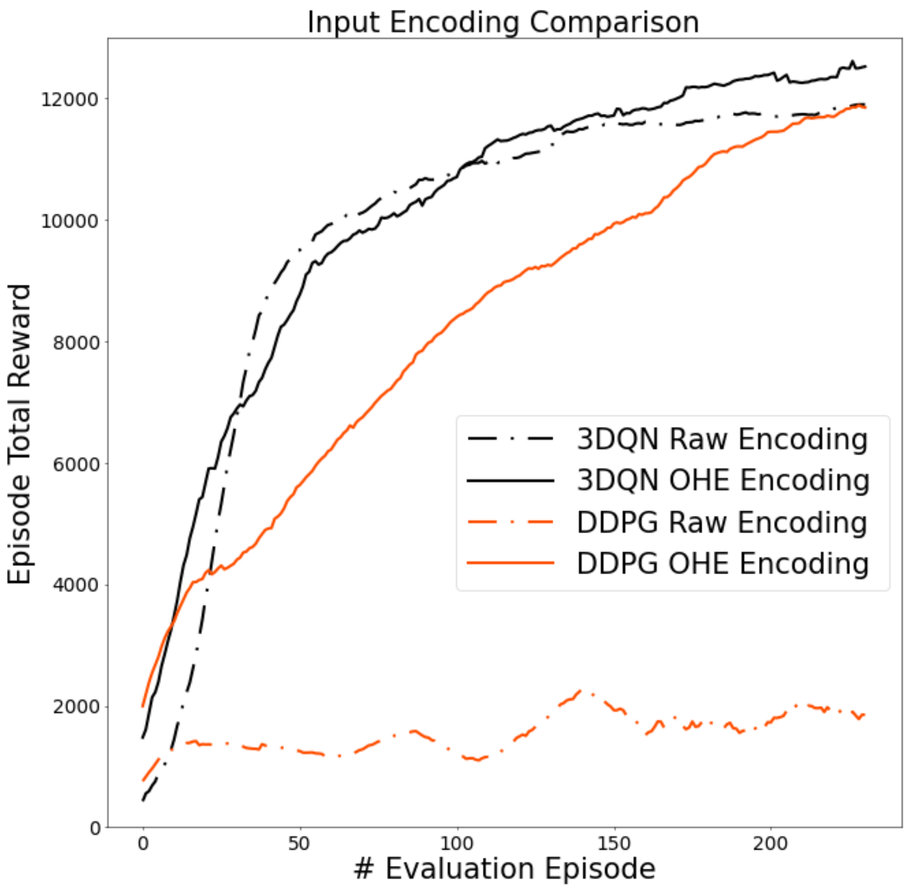


Figure 4.5: Total reward R, as a function of the number of episodes, N_{ep}, during evaluation

so, every 20 training episodes, an evaluation of the agent is performed by following the best policy (i.e., no random action can be taken), while vehicles follow a pre-defined path (shown in Figure 4.1).

One can notice that, in the case of 3DQN, OHE converges to a better result since the obtained reward is higher. Instead, OHE is the only possible choice for the DDPG algorithm, since the Raw architecture fails during training resulting in a bad trajectory for the UABS.

Indeed, OHE optimizes separately the estimates for each element of the state, making the training more difficult but also providing better results. For sake of brevity, the results presented in the following are obtained by an UABS that uses OHE as the input layer and 3DQN algorithm for training, but they can be easily extended for the DDPG-OHE case.

Figure 4.6 shows the P_g for different satisfaction thresholds $\hat{N}_s$. It can be seen that the priority $p_t^{(g)}$, introduced in the reward function, is fundamental to let the UABS learn a trajectory that maximizes the QoE. Indeed, by removing it from the reward function (i.e., considering the reward as the instantaneous rate only), the UABS is not able to provide services with high user satisfaction threshold requirements. Also, it is important to give the UABS the possibility to choose its speed. Reducing the Action space to the direction selection only and fixing the speed, for example to 20 m/s, the QoE is worsened.

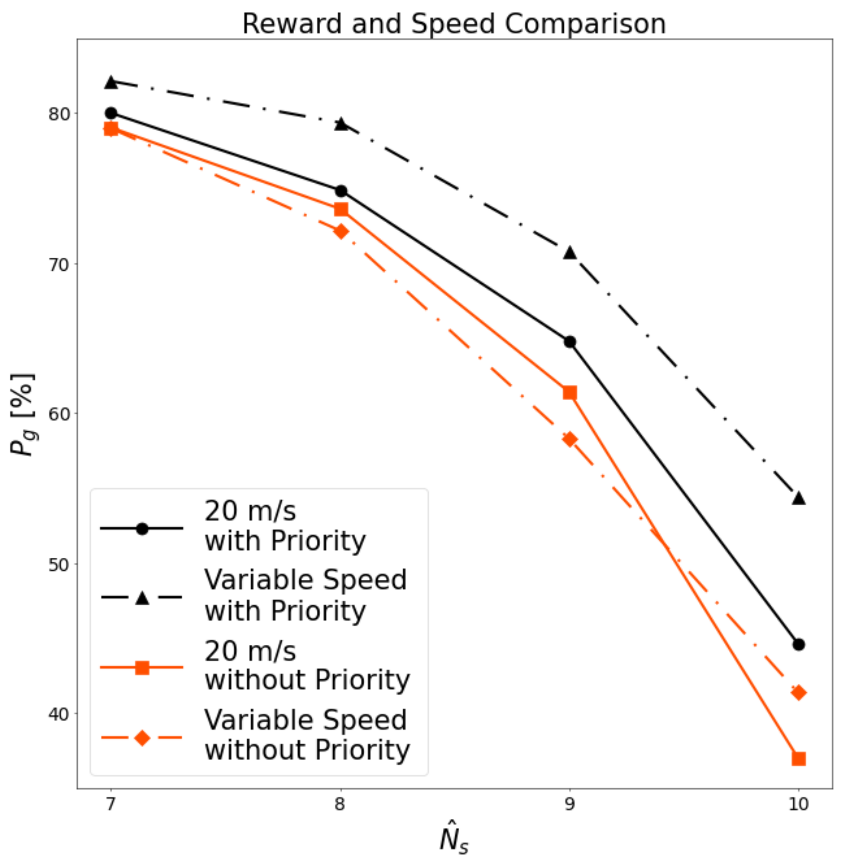


Figure 4.6: Percentage of satisfied users, P_g, for different service thresholds, $\hat{N}_s$, considering 3DQN and OHE encoding. The priority curve shows the performance when considering the reward function defined in equation (4.12) and an UABS trained using a reward function that do not use the priority factor $p_g(t)$ ("without Priority"), that is $R_t = \sum_{g \in G} r_t^{(g)}$.

4.5 Meta-Reinforcement Learning-based Trajectory Design

As stated before, meta-learning can deeply improve conventional RL performance by reducing the number of episodes that need to be simulated in order to optimize the policy that controls the UABS's trajectory when facing a new task. In the following, a meta-RL solution based on CoMPS [112] will be described.

4.5.1 System Model and Problem Definition

As illustrated in Figure 4.7, a learning task consists of an initial position $p_u[0] = [x_u[0], y_u[0]]$ of the UABS on the plane and of a traffic pattern. Time is discretized as $t = 0, 1, \ldots, T$, where T is the maximum duration of an episode. The traffic pattern is defined by the number G of GUEs, by the path P_g, speed v_g and (discrete) starting time instant $t_g \in \{1, \ldots, T\}$ for each GUE $g \in \{1, \ldots, G\}$, as well as by the probability p_{msg} that a GUE generates a packet at each time step. A path P_g is a piece-wise linear curve connecting successive points on the plane.

Given the input parameters $\tau = (G, \{v_g, P_g, t_g\}_{g=1}^{G}, p_{msg})$ defining a traffic pattern, a traffic simulator

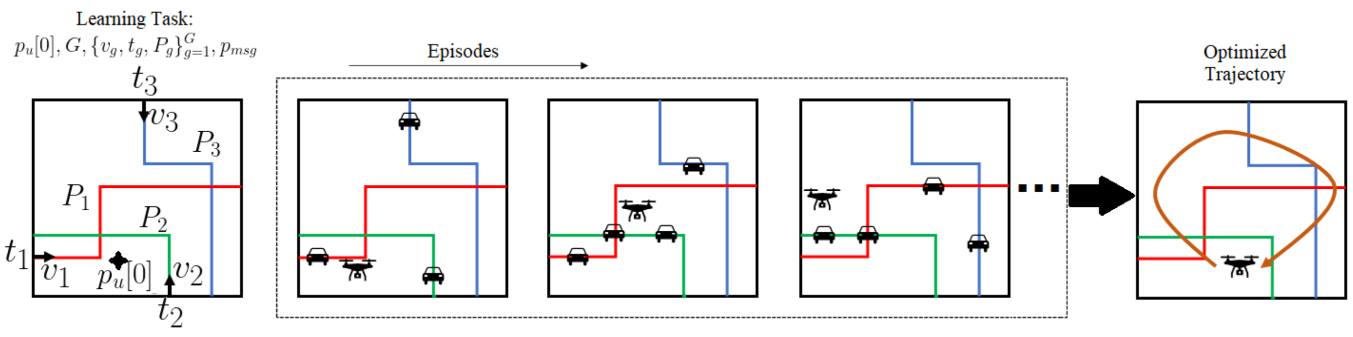


Figure 4.7: A learning task is defined by an initial UABS's position $p_u[0]$ and by a traffic pattern determined by the number of GUEs, G, the GUEs' speeds, $\{v_g\}_{g=1}^G$, the GUEs' discrete starting time instants, $\{t_g\}_{g=1}^G$, paths, $\{P_g\}_{g=1}^G$, and packet generation probability, p_{msg}. The UABS interacts with the learning task through a simulator over a number of episodes in order to optimize its trajectory.

produces the positions $p_g[t] = [x_g[t], y_g[t]]$ for each GUE $g = 1, \dots, G$ at discrete time instants $t = t_g, t_g + 1, \dots, T_g$, where T_g is the smaller value between the total duration of an episode, T, and the time at which the end point of a path is reached by the GUE g. Specifically, the simulator implements a Markov model $p[t] \sim \mathrm{P}_\tau(p[t]|p[t-1])$ to generate the GUEs' positions $p[t] = [\mathrm{p}_1[t], \dots, \mathrm{p}_G[t]]$ at time instant t as a function of the previous positions $p[t-1]$ as well as of the traffic pattern τ. The conditional distribution $\mathrm{P}_\tau(p[t]|p[t-1])$ can account for interactions among GUEs and for random events that may affect the GUEs' trajectories.

Assuming constant altitude, the UABS's position during the T discrete time instants of an episode is described by the sequence $p_u[t] = [x_u[t], y_u[t]]$ for $t \in [0, 1, \dots, T]$. At each time instant t, the UABS can hover, or it can move in one of the eight possible directions $\mathcal{A}_D = \{\leftarrow, \uparrow, \rightarrow, \downarrow, \nwarrow, \nearrow, \searrow, \swarrow\}$. Therefore, the action space is defined as $\mathcal{A} = \{\emptyset, \mathcal{A}_D\}$, with $\emptyset$ indicating the hovering decision.

While on route, at each time instant $t \in \{t_g, t_g+1, \dots, T_g\}$, a GUE can produce a message with probability p_{msg}. This measurement is stored only for the current time and discarded if not delivered to the UABS. Denoting as $\mathrm{SNR}_g[t]$ the SNR level of GUE g towards the UABS at time instant t, GUE g is assumed to be *covered* at time t if the inequality

$$\mathrm{SNR}_g[t] \geq \mathrm{SNR}_{\mathrm{th}} \tag{4.14}$$

holds, given a fixed threshold SNR_{th}. When condition (4.14) is satisfied, the GUE can successfully communicate a message to the UABS at time instant t. The UABS can receive at most $C_{\max}$ packets at the same time t. If more than $C_{\max}$ GUEs satisfy condition (4.14) and have a packet to transmit, the UABS randomly selects a subset of $C_{\max}$ GUEs from which to receive a packet.

The goal is to optimize the stochastic policy $\pi(a|s)$ for the UABS that selects *action* $a \in \mathcal{A}$ as a function of the current *state* s of the system, i.e., $a[t] \sim \pi(\cdot|s[t])$. The state is defined as the collection of all positions of UABS and GUEs, $s[t] = (p_u[t], p[t]) \in \mathcal{S}$. After selecting an action $a[t]$, the UABS and all the GUEs move to state $s[t+1]$ with transition probability $\mathrm{P}_\tau(s[t+1]|a[t], s[t])$ given as

$$\mathrm{P}_\tau(s[t+1]|a[t], s[t]) = \mathrm{P}_\tau(p[t+1]|p[t]) \cdot \mathbb{1}(p_u[t+1] = f(p[t], a[t])),$$

where the conditional distribution $\mathrm{P}_\tau(p[t+1]|p[t])$ is implemented by the traffic simulator; $f(p_u[t], a[t])$ is a function that updates the position of the UABS given action $a[t]$; and $\mathbb{1}(\cdot)$ is the indicator function. Given state s and action a, the UABS obtains a scalar random reward $r[t] \sim \mathrm{P}_\tau(r|s)$ equal to the sum of packets collected by the UABS, i.e.,

$$r = \min\left(C_{\max}, \sum_{g=1}^{G} r_g\right). \tag{4.15}$$

In (4.15), the random variable r_g equals one if GUE g has a packet to transmit and satisfies the coverage condition (4.14). Note that the random variable r_g is a function of the current state s, and that its stochasticity arises from the random packet generation process.

Given an initial UABS position $p_u[0]$ and the traffic pattern τ, the design problem for the policy $\pi(a|s)$ is formulated as the optimization of the discounted average return

$$\max_{\pi}\left\{J_{\tau_0}(\pi) = \sum_{t=1}^{T} \gamma^t \mathbb{E}_{\pi(a[t]|s[t])}\left[r[t]\right]\right\}, \tag{4.16}$$

with discount factor $\gamma \in (0,1]$ [143]. In (4.16), the problem configuration is identified as $\tau_0 = [p_u[0], \tau]$, by explicitly indicating the dependence of the expectation on the policy $\pi(a[t]|s[t])$. The average also accounts for the transition probability (4.15) and for the random reward (4.15).

As for the channel model considered, it has been described in 3.2.

4.5.2 Meta-Reinforcement Learning Algorithm

In this section, at first, a standard policy gradient-based solution will be explored. This approach addresses problem (4.16) from scratch for a fixed configuration τ_0 given by initial UABS position $p_u[0]$ and traffic pattern τ. Then, by exploiting continual meta-learning, it is possible to transfer knowledge across different configurations, to avoid a large number of training episodes.

Conventional Reinforcement Learning

To address problem (4.16) for a given configuration τ_0, a parameterized policy $\pi_\theta(a|s)$ is introduced and a standard policy gradient method [144, 143] is adopted. Accordingly, the gradient of the reward function $J_{\tau_0}(\pi_\theta)$ in (4.16) is estimated as

$$\widehat{\nabla}_\theta J_{\tau_0}(\pi_\theta) = \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a[t]|s[t]) G[t], \tag{4.17}$$

with return $G[t] = \sum_{t'=t}^{T} \gamma^{t'-t} r[t']$. The gradient (4.17) is computed at the end of each episode of T time steps based on the *experience* $e := [s[0], a[0], r[0], \dots, s[T], a[T], r[T]]$. The gradient (4.17) is used to update the policy parameters vector θ as

$$\theta \leftarrow \theta + \eta \widehat{\nabla}_\theta J_{\tau_0}(\pi_\theta) \tag{4.18}$$

with learning rate $\eta > 0$ [143].

Meta-Reinforcement Learning

In continual meta-RL, the UABS explores configurations τ_0^i sequentially over a discrete index $i = 0, 1, \ldots$ The goal is to transfer knowledge from previously observed tasks so as to prepare to solve problem (4.16) for future configurations using fewer episodes. A key challenge in this process is posed by the assumption that the UABS cannot run additional simulations for previously encountered configurations. This problem can be addressed by storing information about experiences from previous configurations.

Following [112], information is assumed to be transferred from previous tasks in the form of an initialized model parameter vector θ^0 for the policy gradient update (4.18). As illustrated in Figure 4.8, continual meta-RL consists of two main steps applied for each new configuration τ_0^i:

- Conventional policy gradient-based RL is applied over N episodes to maximize the expected reward $J_i(\theta) = J_{\tau_0^i}(\pi_\theta)$ with initialization θ_i^0, producing the optimized parameter vector $\theta_i^*(\theta_i^0)$ as a function of θ_i^0;
- A meta-update of the initialization θ_i^0 is applied with the goal of maximizing the sum of the expected rewards for the configurations encountered so far for the problem

$$\theta_{i+1}^0 \leftarrow \arg\max_{\theta^0} \sum_{i'=0}^{i} \tilde{J}_i(\tilde{\theta_i^*}(\theta^0)). \tag{4.19}$$

In (4.19), the notations $\tilde{J}_i(\theta)$ and $\tilde{\theta_i^*}$ indicate that the UABS cannot run new episodes for previous and current tasks, and hence it can only estimate the average return $J_i(\theta)$ and the optimized model parameter vector $\theta_i^*(\theta^0)$ for configurations $i' = 0, \ldots, i$. These are explained next.

In order to estimate $J_i(\theta)$ along with the policy parameter $\theta_i^*(\theta^0)$ without reusing the simulator, for configuration τ_0^i, Continual Meta Policy Search (CoMPS) [112] stores a *full experience set* $\mathcal{E}_i = \{[e_{i,n}, \pi_{i,n}]\}_{n=1}^N$ including all the experiences

$$e_{i,n} = [s_{i,n}[0], a_{i,n}[0], r_{i,n}[0], \ldots, s_{i,n}[T], a_{i,n}[T], r_{i,n}[T]] \tag{4.20}$$

for configuration τ_0^i, as well as the probabilities to choose the corresponding actions in $e_{i,n}$

$$\pi_{i,n} = [\pi_{\theta_{i,n}}(a_{i,n}[0]|s_{i,n}[0]), \ldots, \pi_{\theta_{i,n}}(a_{i,n}[T]|s_{i,n}[T])]. \tag{4.21}$$

In (4.20) and (4.21), the notations $s_{i,n}[t], a_{i,n}[t], r_{i,n}[t], \theta_{i,n}$ stand for state, action, reward, and policy parameter at time t for episode n in configuration τ_0^i. In addition, the *best* episode n^* is chosen as the episode that achieves the highest total reward without discounting factor γ [112], i.e., $n^* = \arg\max_n \sum_{t=0}^T r_{i,n}[t]$, and the corresponding experience e_{i,n^*} is saved in the *skilled experience set* $\mathcal{E}_i^*$.

Using the full experience sets $\{\mathcal{E}_{i'}\}_{i'=1}^i$ and the skilled experience sets $\{\mathcal{E}_{i'}^*\}_{i'=1}^i$, CoMPS addresses problem (4.19) as follows. First, *off-policy local updates* are used to obtain the optimized policy parameter vector $\tilde{\theta_i^*}(\theta^0)$ as

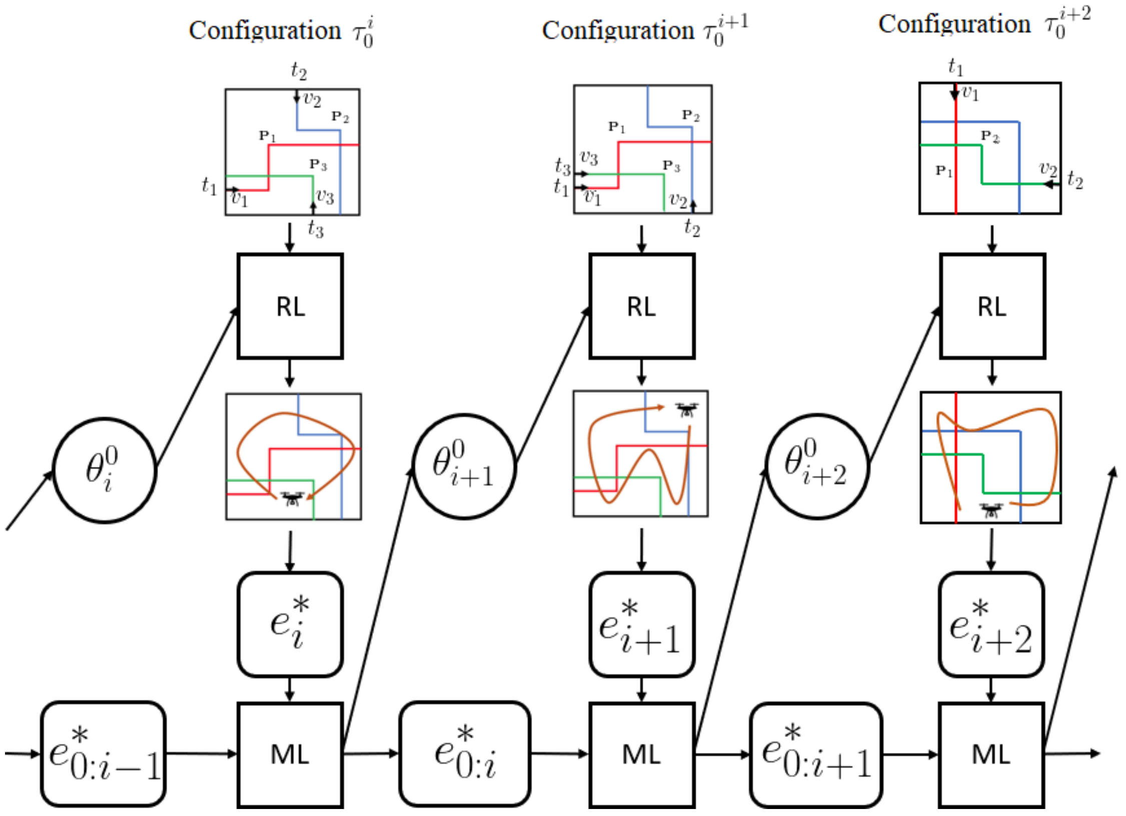


Figure 4.8: Continual meta-reinforcement learning: For each new configuration τ_0^i comprising UABS's initial position and traffic pattern, the UABS implements RL to optimize its trajectory starting from the current initialization of the policy parameter vector θ_i^0 inherited from the previous configurations. After completing optimization on the current configuration, experiences are saved in separate sets, and a meta-learning step (ML) is carried out using offline RL.

$$\tilde{\theta}_i^*(\theta^0) = \theta^0 + \eta \sum_{t=0}^{T} \frac{\pi_{\theta^0}(a_{i,n}[t]|s_{i,n}[t])}{\pi_{\theta_{i,n}}(a_{i,n}[t]|s_{i,n}[t])} \cdot \nabla_{\theta^0} \log \pi_{\theta^0}(a_{i,n}[t]|s_{i,n}[t]) G_{i,n}[t]$$

with learning rate $\eta > 0$ and corresponding discounted return $G_{i,n}[t] = \sum_{t'=t}^{T} \gamma^{t'-t} R_{i,n}[t']$ as defined in (4.17). In (4.22), the episode n is selected at random from the N episodes in set $\mathcal{E}_i$. Furthermore, the *importance sampling* ratio $\pi_{\theta^0}(a_{i,n}[t]|s_{i,n}[t])/\pi_{\theta_{i,n}}(a_{i,n}[t]|s_{i,n}[t])$ is included in (4.22) in order to compensate for the generally different probability assigned to action $a_{i,n}[t]$ given state $s_{i,n}[t]$ by the policies $\pi_{\theta^0}(a|s)$ and $\pi_{\theta_{i,n}}(a|s)$. This can partly mitigate the performance degradation caused by the adoption of off-policy optimization [145, 146].

The objective $\tilde{J}_i(\theta)$ is evaluated using the skilled experience $\mathcal{E}_i^*$ via behavioral cloning [147]. The behavioral cloning loss measures how well the policy π_θ can reproduce the near-optimal, skilled trajectory $e_{i,n^*} \in \mathcal{E}_i^*$. It is accordingly defined as

$$\tilde{J}_i(\theta) = -\sum_{t=0}^{T} \log \pi_\theta(a_{i,n*}[t]|s_{i,n*}[t]). \qquad (4.22)$$

Finally, CoMPS applies gradient-based optimization to problem (4.19) as

$$\theta^0 \leftarrow \theta^0 - \frac{\kappa}{i+1} \sum_{i'=0}^{i} \nabla_{\theta^0} \tilde{J}_i(\tilde{\theta}_i^*(\theta^0)), \tag{4.23}$$

with learning rate $\kappa > 0$.

In order to reduce computational complexity as i grows in (4.23), B tasks are sampled among the available $i+1$ tasks to compute the gradient in (4.23). This way, evaluating the meta-update (4.18) requires order $O(4I_{\text{meta}}BTC)$ operations, assuming I_{meta} iterations for the meta-update (4.23), where C represents the computational complexity of applying policy $\pi_\theta(a|s)$ from the state s. In contrast, conventional RL (4.17) requires order $O(2I_{\text{conven}}TC)$ operations, where the number of iterations I_{conven} is typically very large [117]. Therefore, by transferring knowledge from previous environments, meta-RL can significantly reduce the computational complexity.

4.5.3 Numerical Results

In this section, insights and experimental pieces of evidence on the benefits of meta-learning via CoMPS as compared to conventional RL are provided. Since meta-learning aims at transferring useful knowledge across different configurations encountered over time index i, as a benchmark, a basic *transfer RL* solution is considered, which uses the policy parameter vector θ_i^* optimized based on the ith configuration as the initialization of conventional RL (Section 4.5.2) for the $(i+1)$th configuration. If not stated otherwise, parameters used during the simulations are listed in Table 4.3.

Toy Example

At first, a simple setup consisting of a small 40 m $\times$ 40 m grid world with two possible tasks is considered. The configurations for the two tasks differ only in the path P_g traveled by the three GUEs ($G = 3$), whereas other parameters are fixed: The initial position of the UABS is set as the bottom-right corner of the square area, i.e., $p_u[0] = [20, 0]$; the speed for the GUEs are given as $v_1 = v_2 = v_3 = 1$ m per time step $t = 1$ s, the message generation probability is $p_{msg} = 1$, and the starting time instants of the GUEs are assumed to be $t_1 = 1, t_2 = 2, t_3 = 3$. The duration of an episode is set to $T = 60$ s. In the path P_g for task τ_0^1, all the GUEs start from the bottom right corner of the square area to move in clockwise direction along the perimeter of the area, while for task τ_0^2 the movement of GUEs is taken in counterclockwise. Lastly, tasks are assumed to be presented alternatively for every discrete time index i.

Figure 4.9 plots the average number of packets collected per episode, assuming $N = 50$ episodes, over time index i. The error regions are obtained by evaluating the standard deviation over 10 independent experiments. Conventional RL cannot take advantage of the data from i configurations, while the performance of transfer RL is affected by a negative transfer of information from the previous configurations. In contrast, meta-RL via CoMPS can effectively transfer information from the i previous configurations. This is illustrated

Table 4.3: Meta-RL Simulation Parameters

Parameter	Notation	Toy Example	Urban Scenario
Number of tasks explored	K	50	50
Number of tasks	N	50	50
Learning rate (Eq. 4.18)	η	0.001	0.001
Learning rate (Eq. 4.23)	κ	0.0001	0.0001
Discount factor	γ	0.8	0.8
Time step [s]	t	1	1
Maximum number of packets received by the UABS	$C_{\max}$	10	10
UABS speed [m/s]	v_u	1	20
GUE speed [m/s]	v_g	1	10
Transmit power [dBm]	P_{tx}	0	20
Noise power [dBm]	P_{noise}	-100	-100
Transmitting antenna gain [dB]	G_{tx}	0	0
Receiving antenna gain [dB]	G_{rx}	0	0
Packet generation probability	p_{msg}	1	1
SNR Threshold [dB]	SNR_{th}	50	-10
Carrier frequency [GHz]	f_c	30	30

by the initial trajectory optimized by meta-RL, which is shown in the top part of Figure 4.9 for increasing values of i. The figure demonstrates how meta-RL gradually identifies a useful initial trajectory from which fast adaptation can be carried out for both tasks.

Urban Scenario

In order to evaluate the effectiveness of meta-learning over a more realistic setting, simulated traffic patterns using the SUMO software for an area in the city of Bologna, Italy, is considered, with a dimension of 1500 m $\times$ 900 m [135]. In this scenario, $K = 50$ different task configurations, characterized by different numbers of GUEs (randomly chosen between 15 and 30) moving with different random speeds along different paths, are explored sequentially over time index $i = 0, \ldots, 49$. The duration of an episode is set to $T = 300$ s.

Figure 4.10 shows the average number of packets collected per episode across $N = 50$ total episodes as a function of time index i. Again, the error regions are obtained by considering the standard deviation over 10 independent experiments. In a manner that reflects well the results reported for the toy example, meta-RL outperforms both conventional and transfer RL by successfully transferring knowledge from previously encountered configurations.

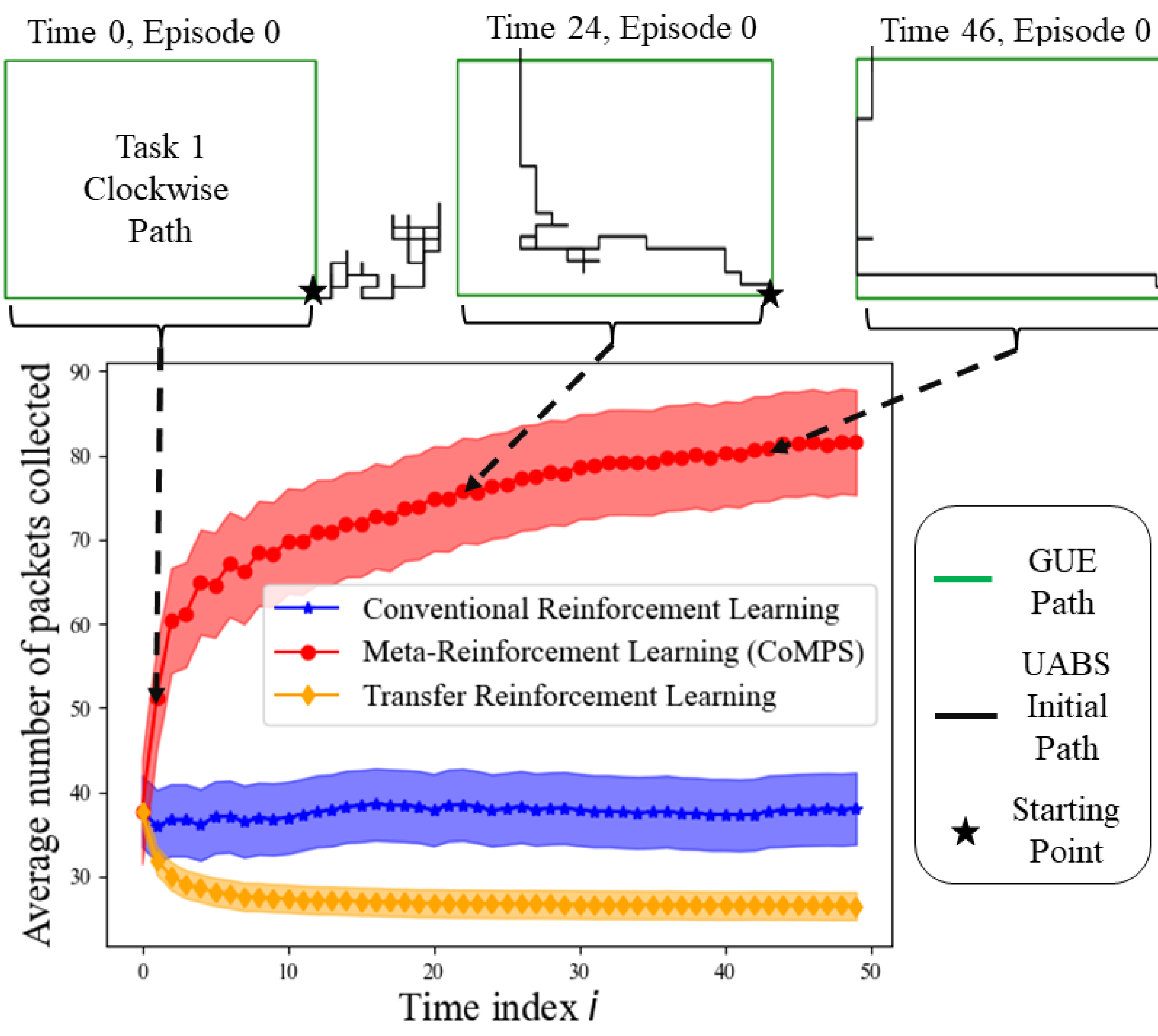


Figure 4.9: (Bottom) Average number of packets collected by the UABS across $N = 50$ episodes as a function of time index i; (Top) Initial trajectory of UABS obtained from the meta-learned initialization θ_i^0 (4.19) (visualized as a black line). For this toy example, two tasks are deployed alternately for each time i while the only difference between the two tasks is the path P_g: even i takes a clockwise path while odd i has a counterclockwise path.

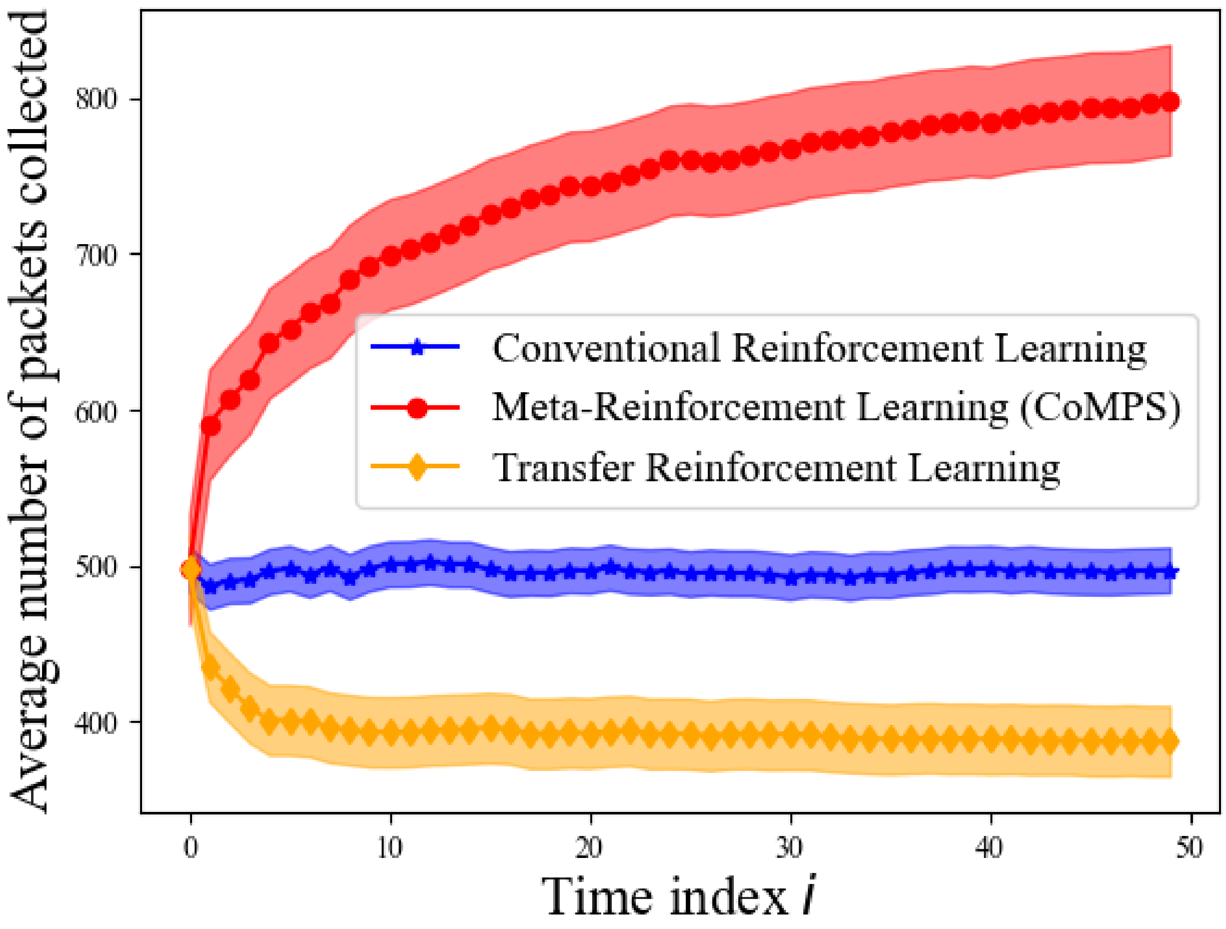


Figure 4.10: Average number of packets collected by the UABS across $N = 50$ episodes as a function of time index i. GUEs' paths are generated using the SUMO software [135].

4.6 Conclusions

In this chapter different strategies to solve the trajectory design problem in UABS network providing services to vehicular applications have been described.

At first, architectures based on Q-Learning and its extensions have been considered. It has been proved that such algorithms are able to generate a good trajectory by letting the agent explore the environment, allowing the UABS to track vehicles' movement, following them during their path. In particular, the dueling architecture turns out to be suitable for the target tracking problem and the inclusion of beamforming information in the state definition can help solve such a problem. In addition, an analysis on different UABS related parameters was performed, showing that there exists an optimal configuration that leads to better results in terms of network throughput.

Secondly, a comparison between 3DQN and DDPG has been carried out, by addressing the advantages and the limitations of the use of a discrete action space against a continuous one. Also, the advantages of using OHE encoding for the input state during training have been proved and a priority-based mechanism has been introduced in the reward function to offer continuous service for high demanding V2X applications. Even if the pool of actions is more limited, due to its intrinsic limitation to use a discrete set, 3DQN outperforms DDPG, showing that a more complex algorithm and action space is not useful in the considered problem.

Finally, in order to reduce the data requirements for RL-based training, this chapter covers the idea of extracting useful information from previously encountered traffic configurations to adapt quickly to new environments via meta-RL. Even without the ability to actively revisit previous traffic conditions, meta-RL can optimize the initial policy parameter vector so as to reduce the number of exploration steps during training.

Chapter 5

Radio Resource Management Techniques for UAV Networks

5.1 Introduction

In this chapter, RRM techniques for UABSs-aided vehicular networks are investigated. Although 5G and beyond networks aim to guarantee service availability for an ever-increasing number of users even in urban areas, the type of service and the related requirements in terms of latency, quality, and data rate are very stringent [148]. A scenario at mmWave frequencies where terrestrial macro BSs (MBSs) provide access to vehicles together with UABSs, which are connected to macro Base Stations (MBSs) for the command and control signalling, is considered. For the UABS, this becomes a backhaul link through which vehicular traffic is forwarded. In this context, an Integer Linear Program (ILP) that jointly addresses vehicular applications' requirements, beam selection and resource allocation optimization is proposed, where both terrestrial and aerial BSs are considered.

There already exist some works in the literature studying RRM techniques considering UABSs for vehicular applications. Among these, the minimum number of UAVs to be deployed to offer communication coverage to all vehicles in the area is studied in [149, 150] and, in particular, the issue of improving resource allocation for UAV-enabled vehicular communications is addressed in [151, 122, 149]; the energy consumption performance is analyzed in [151], while [122] focuses on network throughput and [149] on the percentage of resources allocated to vehicles. However, these resource allocation problems do not consider nor optimize the backhaul connections - compulsory for UAV-aided architectures - or joint terrestrial-aerial coordination. Instead, in this chapter, a joint optimization of the backhaul and access connections considering aerial and terrestrial links at the same time is introduced. On the other hand, both radio resource assignment and backhaul are considered, for example, in [152, 153]; however, the challenges of beamforming and user movement are not taken into account. In this chapter, the RRM includes an optimized beam selection strategy together with time-frequency resource allocation and the algorithm is re-computed dynamically to accommodate the vehicular movement. Furthermore, a novel performance metric measuring the QoE of vehicular users is considered. To the best of my knowledge, no other work in the literature designs a joint RRM strategy between terrestrial and aerial infrastructure optimizing a QoE metric and accounting for backhaul capacity limits, access-backhaul resource splitting and beamforming.

The rest of the chapter is organized as follows. Section 5.2 describes the considered model and its implications, whereas Section 5.3 introduces the proposed ILP for RRM. Section 5.4 analyzes the obtained results and Section 5.5 concludes the chapter.

5.2 System Model

5.2.1 Reference Scenario and Application Requirements

An urban scenario with the elements depicted in Figure 5.1 is considered. A set $\mathcal{A}$ of UAVs is flying above the area at a constant altitude, h, from the ground. Each UAV is assumed to be dedicated to a predefined mission, but at the same time has radio-frequency equipment on board (i.e., acts as UABS). Then, fixed on the ground there are terrestrial BSs, hereafter referred to as MBSs, being part of a set $\mathcal{M}$. Through the city roads are driving vehicular users, hereafter denoted as GUEs, g, belonging to the set $\mathcal{G}$.

The channel model considered is the same described in Section 4.3.2. MBSs have wired connections with the network core, therefore, this link is assumed to be robust and has the capacity sufficient to forward signalling and content. On the opposite, UABSs have to maintain a wireless communication link with MBSs to receive the necessary command and control signalling. This backhaul link is subject to RRM optimization.

As for the application requirements, the extended sensing scenario is analyzed, according to which, each vehicle wants to exchange data gathered through local sensors or video with other vehicles nearby. By exchanging V2X messages [14], vehicles can enhance the perception of their surroundings beyond what their own sensors can detect. As the network is collecting sensor data, the traffic requirements are particularly challenging for the uplink communication [103]. The traffic demand D_g for the GUE $g \in \mathcal{G}$ depends on the degree of automation considered, with a data rate ranging between 1 and 1000 Mbps [103]. The mobility of

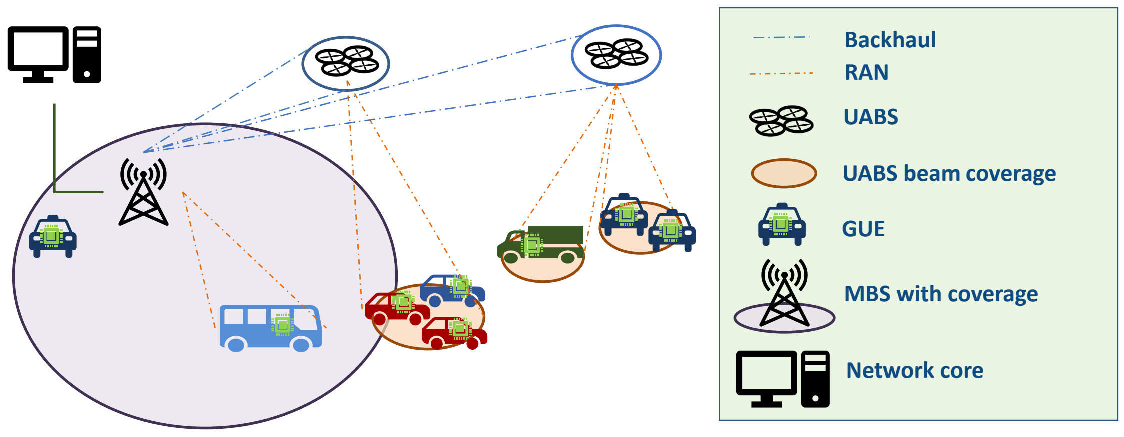


Figure 5.1: Network components and architecture.

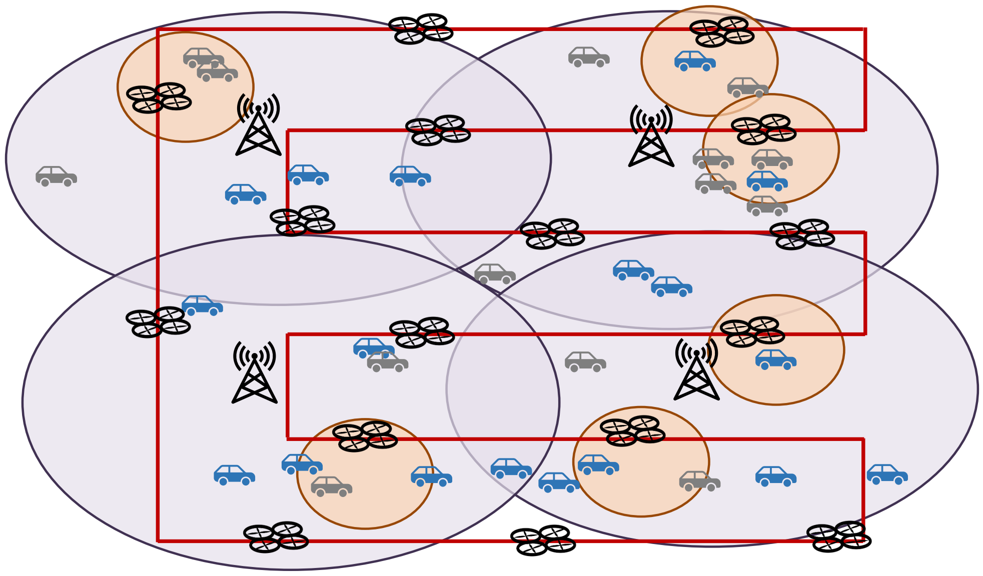


Figure 5.2: Scenario example where UAVs follow a Paparazzi-scan trajectory to accomplish a predetermined task

GUEs and UABSs leads to a dynamic scenario, which is considered by updating the traffic demand of GUEs with a time granularity of $\Delta t = 100$ ms[1]. One GUE, g, is denoted as served in the given interval, Δt, if it is able to upload its demand D_g. To account for an appropriate QoE metric, each GUE, g, is assumed to be *satisfied* if it is served for at least $\hat{N}_\mathrm{s}$ time intervals within a given time window, $T_w = N_\mathrm{w}\,\Delta t$ (being $\hat{N}_\mathrm{s}$ a given percentage of N_w). The QoE requirement, $\hat{N}_s$, is determined by the time the vehicle may take to execute a maneuver (e.g., turn at crossroads, enter/exit roundabouts, stops, and so on). Example values are reported in [148], considering the average values of vehicles' speed and communication range in specific use cases. To increase the probability of satisfying a vehicle, the network prioritizes the traffic of GUEs who have been served in the previous intervals (see Section 5.3.2).

5.2.2 UAVs Trajectory

As previously mentioned, the UAVs in the scenario have predefined missions (other than communication) constraining them from flying on an already determined trajectory. Possible UAVs' tasks might be video-monitoring an area or collecting sensor data scattered all over the scenario. A suitable trajectory for such missions is a Paparazzi-scan, which shapes a serpentine that scans the entire area [156]. A scenario example is pictured in Figure 5.2. To model the UAV movement, the scan serpentine is completely defined by the area side and a parameter introduced on purpose, named as Paparazzi-scan sensing radius, r_p. The value of $2r_p$ defines the serpentine width, while the height and length are respectively defined by Q and Q'. The r_p choice depends on the UAVs potential coverage; its definition allows UABSs to eventually cover the entire service area for a number of N_uav. Each new UAV enters the scan trajectory every t_p. Please note that the model

[1]Note that 100 ms is the packet generation interval for many types of V2X including CAM and CPM [154, 155]

presented hereafter works independently from the UAV trajectory, and it can be re-applied for any given UAV path.

5.3 Problem Definition

5.3.1 Approaching Radio Resource Management

A radio RU may span over three dimensions: time, frequency, and space. For what concerns time-frequency radio resources definition, the numerology provided by the 3GPP standard for 5G [157] is considered. In general, time-frequency radio resources refer to a set of Resource Blocks (RBs), each composed of 12 consecutive subcarriers carrying an OFDM signal. Subcarriers span over the frequency axis with a subcarrier spacing Δf. Then, every RB is defined in time by time slots of duration T_{slot}, which are strictly related on the choice of Δf for carrying 14 OFDM symbols. Then, the chosen bandwidth value, B, determines the total number of RBs at disposal, W. If $B_{\text{ch}} = \frac{B}{12\Delta f}$, W can be formalized as:

$$W = \frac{B}{12\Delta f} \cdot \frac{\Delta t}{T_{\text{slot}}}. \tag{5.1}$$

Furthermore, the space dimension is defined by a number of *covering beams*, resulting from beamforming. Supposing fully digital or hybrid beamforming capabilities, the available K beams are determined by the UAV's antenna system configuration. In particular, while each MBS can activate all the beams available on the ground footprint, UABSs can activate at the same time only N_{beam} beams of the K available, due to the need to limit payload and energy consumption of UAVs. This leads us to the optimization of the most appropriate beam selection depending on the current GUE demand.

With the aim of applying resource reuse without creating interference, let us now observe the different links in the network: i) UABS-MBS, ii) GUE-MBS, iii) GUE-UABS.

UABS-MBS

The wireless backhaul forwards all traffic handled by UABSs to the MBSs. Since it is important to guarantee here high capacity and avoid interference, backhaul links are assigned dedicated resources from the overall pool, not shared with (and constraining) access links. Clearly, because of the scenario dynamicity, the set of RUs given to backhaul might change over time.

GUE-MBS

For this access link, a safe reuse of RUs in the space dimension cannot be considered. Given the wide area to cover, MBSs' beams might in fact create overlapping footprints on the ground. Reuse of the same RBs in overlapping beams would result in non-negligible interference.

GUE-UABS

On the contrary, UABSs fly at constant heights highly above the ones of MBSs (hundreds of meters against 25-30 meters), and the resulting coverage spot on the 2D ground plane of each beam is smaller and better distinguishable from others of the same UABS. For this reason, safe reuse of RBs in beams of the same UABS is assumed. To further explain how to access RUs are shared, the GUE-UABS links can be seen as vertical links, while the GUE-MBS ones as horizontal links. While horizontal links are -potentially- interfering with others (and reuse is not recommended), vertical ones leverage on highly directive and narrow beams, such that RB reuse and share will not severely interfere with others. These concepts are later applied to avoid interference.

The number of available RBs is not the same at MBSs and UABSs. Each MBS m may need to save RUs for other services, and therefore has $W^*_m = W/2$ RBs. On the contrary, to be safe from interference of overlapping beams from other UABSs, $W^*_a = W/|\mathcal{A}|$ are the RBs available at the single beam. Relevant to note, backhauling requires dedicated RBs from the same initial pool as access, and therefore a proper tradeoff of access-backhaul RBs needs to be found. In the following, an ILP algorithm providing an optimal solution and maximizing network performance is proposed.

5.3.2 RRM Strategy and Problem Formulation

Given the previous definition of a *served* GUE within an interval Δt, the proposed objective function maximizes the number of successfully served GUEs by MBS and UABS joint operation. From the above discussion, one knows how RUs can be assigned and when reuse is possible. However, there are some degrees of freedom left to be considered: i) the BS (MBS or UABS) from which a GUE $g \in \mathcal{G}$ has assigned RUs, ii) how many RUs should be allocated given the demand D_g, iii) the UABSs' beams that should be activated, iv) and the number of RUs needed by backhaul links given its capacity or data rate. These aspects are then considered in the proposed ILP, which results in a beam selection and joint resource allocation optimization considering the backhaul capacity.

To introduce the ILP, the binary variables are first defined:

$$y_g = \begin{cases} 1 & \text{if user } g \in \mathcal{G} \text{ is served} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{g,m} = \begin{cases} 1 & \text{if } g \in \mathcal{G} \text{ is assigned RUs by MBS } m \in \mathcal{M} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{g,a} = \begin{cases} 1 & \text{if } g \in \mathcal{G} \text{ is assigned RUs by UABS } a \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}$$

$$e_{j_a} = \begin{cases} 1 & \text{if beam } j_a \in \mathcal{K}_a \text{ is active on UABS } a \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}$$

The objective function aims at maximizing the number of served users, through y_g, trying to serve them continuously. In order to allow a continuous service, each user $g \in \mathcal{G}$ is weighted with a priority value that varies over time, p_g. A higher value of p_g increases the probability that g will be served in the following time instants. In particular, in each time window (i.e., $t = 1, .., T_w$) p_g varies as follows:

$$p_g(t) = \begin{cases} 1 & \text{for } t = 1 \\ p_g(t-1) + 1 & \text{if } y_g(t) = 1 \\ p_g(t-1) & \text{if } y_g(t) = 0 \end{cases} \tag{5.2}$$

The following integer variables are also subject to optimization and, together with binary variables, are the output of the RRM procedure. They specify the number of RUs given to a specific communication link:

- $w_{g,m}$ and $w_{g,a}$ represent the number of resources assigned to user $g \in \mathcal{G}$ by the MBS $m \in \mathcal{M}$ or UABS $a \in \mathcal{A}$, respectively;
- $w_{a,m}$ is the number of resources assigned by the MBS $m \in \mathcal{M}$ to the backhaul with UABS $a \in \mathcal{A}$.

To summarize, the purpose of the following RRM ILP is: i) define an optimal joint operation between MBSs and UABSs with the variables $x_{g,m}$, $w_{g,m}$, $x_{g,a}$, $w_{g,a}$ while maximizing y_g; ii) optimize the selection of UABSs' beams with variables e_{j_a} to reach the maximum number of GUEs, iii) considering the RB assignment to backhaul with variables $w_{a,m}$, for all $g \in \mathcal{G},\ a \in \mathcal{A},\ m \in \mathcal{M}$.

The RRM procedure runs every Δt at each MBS $m \in \mathcal{M}$. The problem is thus dependent on m:

$$\mathcal{P}(m): \quad \max \left(\sum_{g \in \mathcal{G}_m} p_g y_g \right) \tag{5.3a}$$

$$w_{g,m} r_{g,m} \Delta t + \\ + \sum_{a \in \mathcal{A}_m} \sum_{j_a \in \mathcal{K}_a} k_{g,j_a} w_{g,a} r_{g,a} \Delta t \geq y_g D_g, \quad \forall g \in \mathcal{G}_m \tag{5.3b}$$

$$\sum_{g \in \mathcal{G}_m} w_{g,m} + \sum_{a \in \mathcal{A}_m} w_{a,m} \leq W_m^* \tag{5.3c}$$

$$\sum_{g \in \mathcal{G}_m} k_{g,j_a} w_{g,a} + w_{a,m} \leq W_a^*, \quad \forall a \in \mathcal{A}_m, \forall j_a \in \mathcal{K}_a \tag{5.3d}$$

$$\sum_{g \in \mathcal{G}_m} \sum_{j_a \in \mathcal{K}_a} w_{g,a} k_{g,j_a} r_{g,a} \leq \frac{r_{a,m}}{F_B} w_{a,m}, \quad \forall a \in \mathcal{A}_m \tag{5.3e}$$

$$\sum_{j_a \in \mathcal{K}_a} e_{j_a} \leq N_{\text{beam}}, \quad \forall a \in \mathcal{A}_m \tag{5.3f}$$

$$\sum_{g \in \mathcal{G}_m} w_{g,a} k_{g,j_a} \leq e_{j_a} W_a^*, \quad \forall a \in \mathcal{A}_m, \forall j_a \in \mathcal{K}_a \tag{5.3g}$$

$$x_{g,m} + \sum_{a \in \mathcal{A}_m} x_{g,a} \leq 1, \quad \forall g \in \mathcal{G}_m \tag{5.3h}$$

$$w_{g,m} \leq x_{g,m} W_m^*, \quad \forall g \in \mathcal{G}_m \tag{5.3i}$$

$$w_{g,a} \leq x_{g,a} W_a^*, \quad \forall a \in \mathcal{A}_m, \forall g \in \mathcal{G}_m \tag{5.3j}$$

$$x_{g,a},\ e_{j_a} \in \{0,1\}, \quad \forall a \in \mathcal{A}_m, \forall g \in \mathcal{G}_m, \forall j_a \in \mathcal{K}_a \tag{5.3k}$$

$$x_{g,m},\ y_g \in \{0,1\}, \quad \forall g \in \mathcal{G}_m \tag{5.3l}$$

$$w_{g,a} \in \{0, ..., W_a^*\}, \quad \forall a \in \mathcal{A}_m, \forall g \in \mathcal{G}_m \tag{5.3m}$$

$$w_{g,m} \in \{0, ..., W_m^*\}, \quad \forall g \in \mathcal{G}_m \tag{5.3n}$$

$$w_{a,m} \in \{0, ..., \min[W_m^*, W_a^*]\}, \quad \forall a \in \mathcal{A}_m \tag{5.3o}$$

where k_{g,j_a} is an input to the problem and indicates with value 1 whether vehicle g is covered by beam j_a of UABS $a \in \mathcal{A}_m$ and 0 otherwise. Sets $\mathcal{A}_m \subseteq \mathcal{A}$ and $\mathcal{G}_m \subseteq \mathcal{G}$ depend on m and are computed later in Algorithm 4. F_B is denoted as the backhaul factor and ranges within [0,1]; it is used to enhance the backhaul

Algorithm 4: RRM Model implementation.

```
Input: $\mathcal{M}$, $\mathcal{A}$, $\mathcal{G}$, $D_g$, $\forall g \in \mathcal{G}$, $N_{\text{beam}}$, $W_a^*$, $W_m^*$, $T_{\text{sim}}$,
    Channel parameters in Table 5.1
Output: $y_g,\ w_{g,m},\ w_{g,a},\ w_{a,m}\ \forall g \in \mathcal{G},\ \forall a \in \mathcal{A},\ \forall m \in \mathcal{M}$
for $t = 0$ to $T_{\text{sim}}$ do
    Update positions $\forall g \in \mathcal{G},\ \forall a \in \mathcal{A},\ \forall m \in \mathcal{M}$
    for each MBS $m \in \mathcal{M}$ do
        for each candidate UABS $a \in \mathcal{A}$ do
            compute $r_{a,m}$;
            for each GUE $g \in \mathcal{G}$ do
                compute $r_{g,m},\ r_{g,a}$;
    for each MBS $m \in \mathcal{M}$ do
        compute subsets $\mathcal{G}_m$ and $\mathcal{A}_m$;
        solve $\mathcal{P}(m)$;
    for each GUE $g \in \mathcal{G}$ do
        if $(t\ mod\ N_{\text{w}}) = 0$ then
            $p_g = 1$
        else if $y_g = 1$ and $1 \leq (t\ mod\ N_{\text{w}}) < N_{\text{w}}$ then
            $p_g + = 1$
```

capacity, by reducing the RUs needed to forward traffic: lower F_B values correspond to greater backhaul capacity. In more detail, each constraint of $\mathcal{P}(m)$ has a specific role. First, constraint (5.3b) ensures each vehicle g transmits a demand of D_g bits given the rate of the unitary RU and the number of RUs assigned by a specific BS. Then, constraints (5.3c) and (5.3d) guarantee that the number of RUs assigned does not exceed the maximum available for MBSs and UABSs bases, respectively. Clearly, RUs allocated for the backhaul are accounted for in both. Also, constraint (5.3e) ensures the backhaul capacity is enough to forward the UABS vehicular traffic to the network. As motivated previously, RUs are reused for different beams only by UABSs. Then, constraints (5.3f) and (5.3g) limit the number of beams that can be simultaneously activated at each UABS $a \in \mathcal{A}_m$ to N_{beam}. Finally, constraints (5.3h) to (5.3j) specify that each vehicle is served by one BS at a time. Expressions (5.3k)-(5.3o) show the validity interval of each variable in $\mathcal{P}(m)$.

5.3.3 Solving the ILP

To finally assess the model performance, a dynamic environment is simulated by following the steps in Algorithm 4. It shows a realistic implementation over time, where RRM decisions might change depending on traffic, movement and channel variations. In fact, at each time t, the positions of vehicles and UABSs are updated, and respective data rates re-computed. Also, traffic priorities are updated with equation (5.2).

Table 5.1: Network and Channel Parameters.

Parameter	**Notation**	**Value**
Area sides [m^2]	QxQ'	1800x1600
Number of MBSs	$\|\mathcal{M}\|$	6
Number of UAVs	$\|\mathcal{A}\| = N_{\mathrm{uav}}$	18
SNR threshold [dB]	γ_{th}	-13.7
UABS transmit power [dBm]	$P_{\mathrm{tx,A}}$	23
GUE transmit power [dBm]	$P_{\mathrm{tx,G}}$	20
UABS transmission gain [dB]	$G_{\mathrm{tx,A}}$	17.72
UABS receiver gain [dB]	$G_{\mathrm{rx,A}}$	17.72
GUE transmission gain [dB]	$G_{\mathrm{tx,G}}$	0
Paparazzi-scan sensing radius [m]	r_{p}	200
Paparazzi-scan time shift [s]	t_{p}	25
UABS speed [m/s]	v_A	20
UABS altitude [m]	h	100
UABS aperture angle	α_{A}	140°
Number of available beams at UABSs	K	9
Max nr. of UABS active beams per Δt	N_{beam}	4
GUE traffic demand per Δt [kbit]	D_g	100
Effective noise power [dBm]	P_{noise}	-106.4
Overall bandwidth [MHz]	B	400
Channel bandwidth [MHz]	B_{ch}	1.44
Subcarrier spacing [kHz]	Δf	120
Slot duration [ms]	T_{slot}	0.125
Number of time windows for QoE	N_{w}	60

5.4 Numerical Evaluation

5.4.1 Performance Metrics

The first considered performance metric is the *service time*, T_{s}, that is the amount of time a vehicle is served, computed in each time window T_{w}. By denoting as N_{s} the number of intervals of duration Δt in T_{w} during which the vehicle is served, it holds: $T_{\mathrm{s}} = N_{\mathrm{s}}\,\Delta t$. In the next section, the statistics of this service time are shown in terms of CCDF (Complementary Cumulative Distribution Function), $\overline{F}_{\mathrm{T_s}} = Prob\{T_{\mathrm{s}} \geq t_{\mathrm{s}}\}$. As a second performance metric, the percentage of satisfied users, $P_g^{(sat)}$, is computed, that is the ratio between the number of users for which $N_{\mathrm{s}} \geq \hat{N}_{\mathrm{s}}$ w.r.t. the total number of vehicles.

5.4.2 Numerical Results

The achieved results are presented in this section. Simulations of the proposed model and scenario run in a Python environment, whereas the Gurobi solver provides the output from the ILP. Each simulation lasts $T_{\text{sim}} = 60$ seconds and parameter settings are listed in Table 5.1. To properly account for vehicular movement through time, the vehicular traces are obtained from SUMO, as it has been done in the previous chapter.

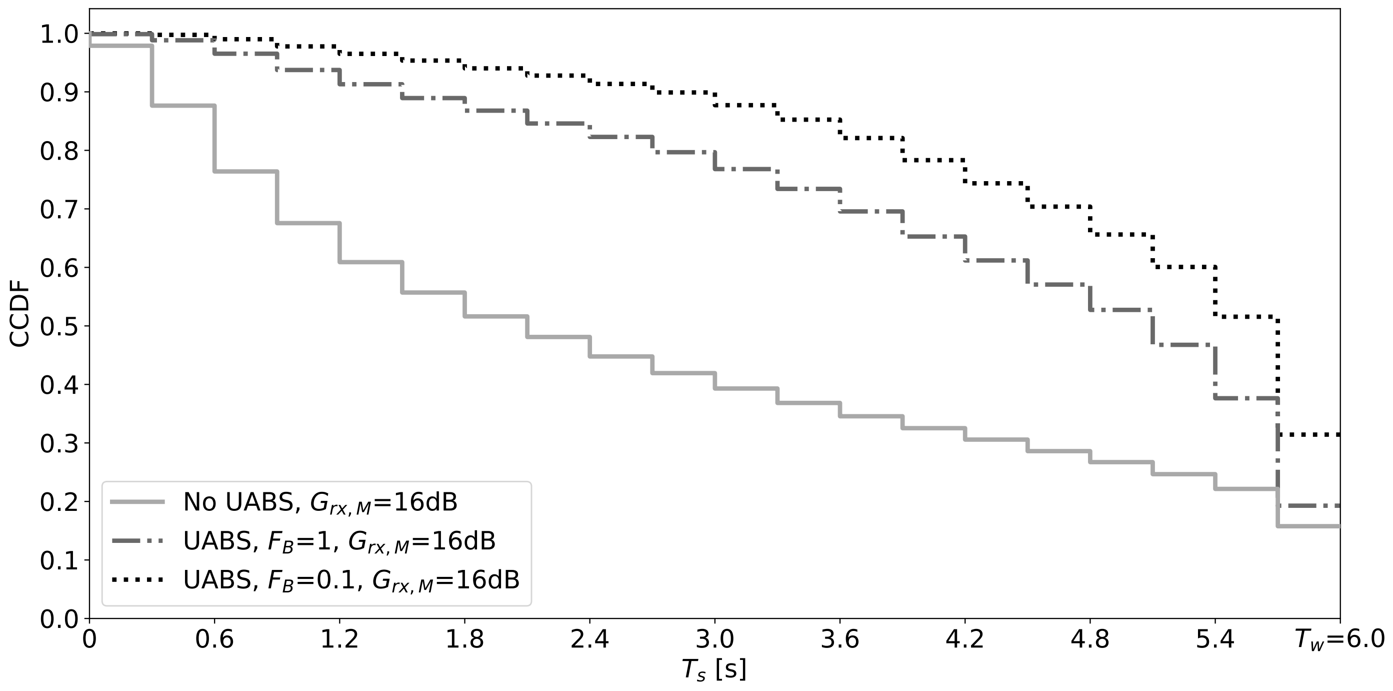


Figure 5.3: CCDF functions over the service time, T_s, with receiving gain $G_{\text{rx,M}} = 16$ dB

In Figure 5.3, one can observe the CCDF of the service time, T_s, when setting $G_{\text{rx,M}} = 16$ dB. Three curves highlight: i) benchmark case, where UAVs are not active as UABSs (obtained by setting $N_{\text{beam}} = 0$); ii) UABSs active and $F_B = 1$; iii) UABSs active and $F_B = 0.1$. Focusing on the first case, the benchmark shows an expected decrease in service time. Indeed, with increasing T_s, the probability $\overline{F}_{T_s}$ for the single GUE is lower. This applies also to the other cases. However, curves with settings ii) and iii) show a notable performance improvement with respect to i). This proves the presence of UABSs helps in maintaining stable and durable links. A further improvement can be appreciated with $F_B = 0.1$. In fact, one can deduce from constraint (5.3e) that the lower is F_B, the higher becomes the backhaul capacity for each RU, allowing the network to use a reduced number of RUs for this link and an increased one for access links.

As in Figure 5.3, Figure 5.4 shows the CCDF with service time for the same settings, except for the MBS receiving gain, $G_{\text{rx,M}} = 25$ dB. As expected, the same behavior as before is observed, but with a substantial improvement in all cases of the GUE's $\overline{F}_{T_s}$ (i.e. GUEs almost double the probability of achieving T_s service time). Indeed, a higher receiving gain of MBSs not only enhances the capacity of backhaul and access links but allows a larger coverage area.

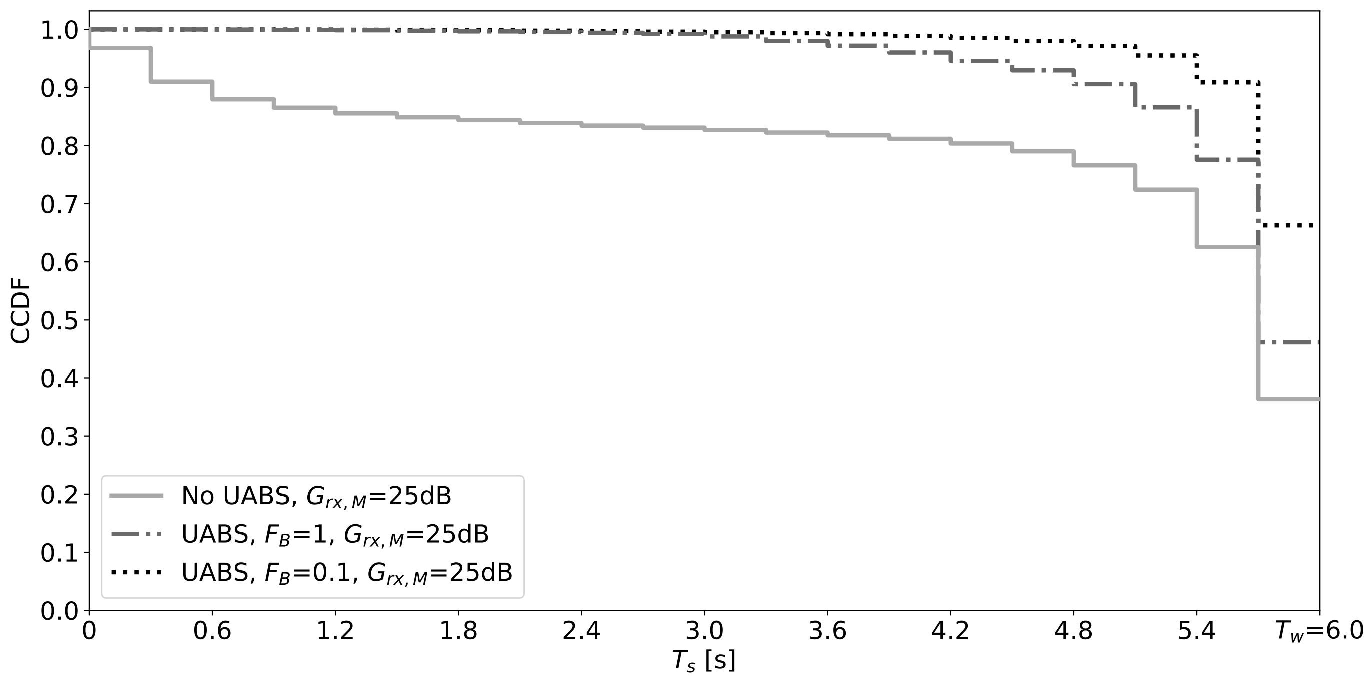


Figure 5.4: CCDF functions over the service time, T_s, with receiving gain $G_{\text{rx,M}} = 25$ dB.

Figure 5.5 shows the percentage of satisfied GUEs, $P_g^{(sat)}$, depending on the minimum number of intervals of duration Δt given by the application requirements, $\hat{N}_s$, chosen over the time window T_w. Results are shown for the same parameter settings as Figure 5.3 and over a number of QoE thresholds: 85%, 90%, 95%, 97%, and 99%. As expected from the results of Figure 5.3, the benchmark shows a lower percentage of GUEs satisfied w.r.t. the cases in which UABSs are present. Furthermore, the curve has a decreasing trend with $\hat{N}_s$. In fact, this repeats the behavior of the CCDF above. Then, as before, from the curves having $F_B = 1$ or $F_B = 0.1$, it can be evinced that a lower F_B is always able to substantially increase the GUE satisfaction. In particular, Figure 5.5 shows that a backhaul capacity enhanced 10 times may improve network performance by more than 20%.

Similarly to Figure 5.4 and 5.5, Figure 5.6 presents values of $P_g^{(sat)}$ while varying the QoE threshold for $G_{\text{rx,M}}$ increased from 16 to 25 dB. The general behavior of the previous curves applies also here. As before, when increasing the beamforming gain at the reception of MBSs, each curve shows a notable increase in GUE satisfaction. This confirms the important role of the usage of beamforming at mmWaves. In fact, with respect to the previous figure, $P_g^{(sat)}$ increases up to 50%.

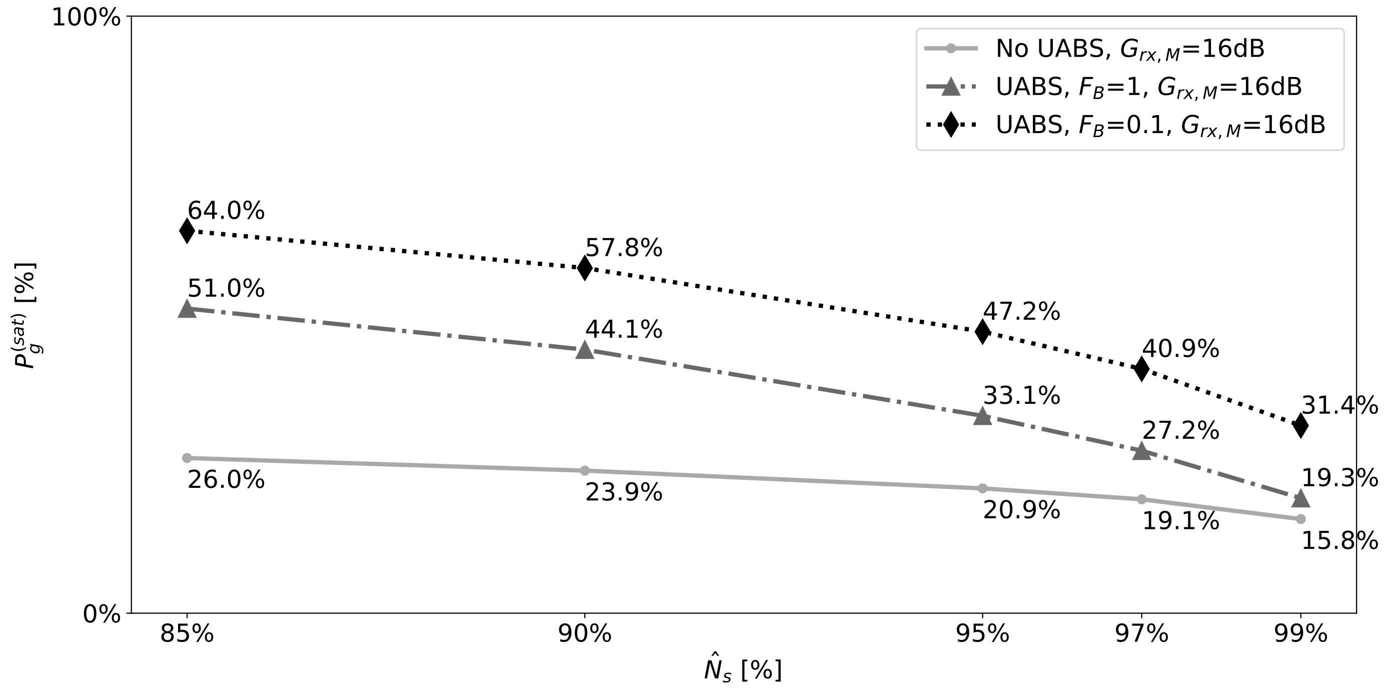


Figure 5.5: Percentage of satisfied GUEs while varying the QoE threshold, $\hat{N}_{\mathrm{s}}$, for different parameter settings and $G_{\mathrm{rx,M}} = 16$ dB.

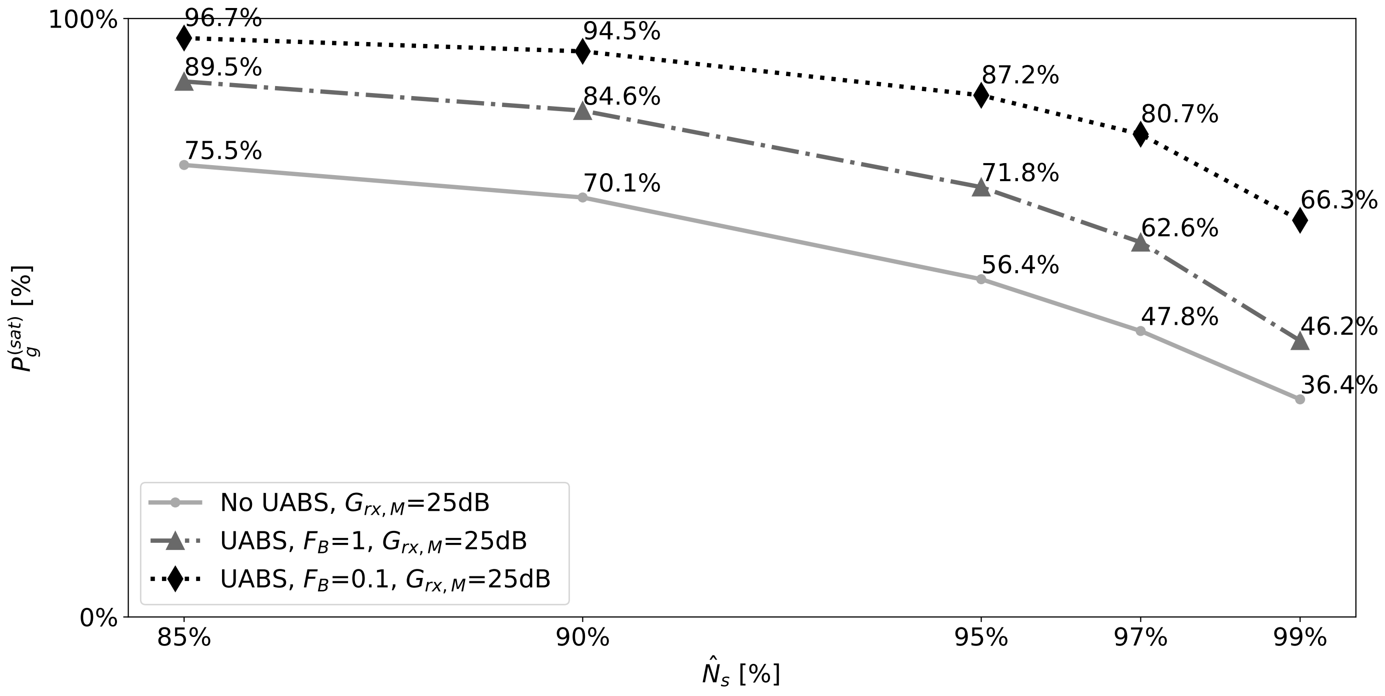


Figure 5.6: Percentage of satisfied GUEs while varying the QoE threshold, $\hat{N}_{\mathrm{s}}$, for different parameter settings, $G_{\mathrm{rx,M}} = 25$ dB

5.5 Conclusions

This chapter covers the fundamental aspects of RRM in a 3D Network (3DN) for vehicular service, analyzing beamforming, resource allocation, and a new QoE metric at mmWaves frequencies. An ILP is proposed to optimize the beam selection and RBs assignment between backhaul and access links, considering a joint aerial-terrestrial operation. Results highlight that the presence of UABSs and the beamforming gain at reception of MBSs increases significantly the QoE for vehicular applications. Moreover, a relevant aspect that emerges is the role of the backhaul capacity, which may limit the traffic handled by a UABS.

Conclusions

The first part of the thesis (Chapters 1 and 2) deals with the study of terrestrial IoT network employing the two most widespread LPWAN technologies, LoRaWAN and NB-IoT, whereas, in the second part, the study is focused on UAV-aided where UABSs are used to provide service to IoT and V2X applications.

At first, this thesis presented a new MATLAB-based simulator tool, LoRaWANSim, which includes physical and upper layers of the LoRa/LoRaWAN protocol stack and which has been openly delivered to the community. As demonstrated, such a simulator has a number of unique features (including, but not limited to (i) support of multiple gateways both for uplink and downlink, (ii) uplink-uplink, uplink-downlink, downlink-uplink, and downlink-downlink interference accounting, (iii) receive window prioritization, (iv) modeling of energy consumption) whilst being decently simple to use and easy to configure, being based on MATLAB.

Besides, some results have been obtained using such simulators, showing not always intuitive trade-offs. A study on different coding rates has been carried out in interference- and noise-limited scenarios. First, it has been demonstrated that some benefit on the packet delivery rate, although marginal, can be achieved only in the latter scenario. Second, as the number of EDs increases significantly, thus making the network operate in heavily interference-limited conditions, the adoption of more powerful coding rates has been proved to be counterproductive. In addition, the impact of downlink transmissions (e.g., acknowledgments) on the average ED energy consumption has been assessed, showing that increasing the number of GWs affects not only the packet delivery rates in uplink and downlink but also the devices' battery life. All these, as well as many other aspects of LoRaWAN, require further investigations and I believe that the simulator presented in this thesis can be a great help.

Secondly, a new Adaptive Data Rate algorithm for LoRaWAN networks, called Collision-Aware ADR, has been proposed, which takes into account collision probability at the MAC layer to assign data rates to end devices. The new algorithm has been compared with the standard as well as other solutions already presented in the literature, considering different performance metrics via simulation and experimental approaches. Results show that CA-ADR outperforms such solutions for networks that are not strongly limited by connectivity issues. This is because CA-ADR exploits the orthogonality of signals emitted with different data rates, a fact that allows for drastically reduce collisions among transmissions.

In addition, cloud- and fog-based architectures have been studied and compared in terms of performance indicators, such as network throughput, latency, and processing capabilities. The fog architecture has been proven to be feasible since even if the NS is deployed on a common Raspberry Pi, it is still able to manage a sufficient amount of traffic for many real-life IoT applications. In addition, the fog architecture allows for reducing the end-to-end latency as expected, while maintaining very good performance in terms of network throughput when compared to the cloud-based scenario. Finally, it has been underlined that the lower latency achievable with the fog architecture can help also in reacting to dynamic environmental changes.

After dealing with LoRaWAN, a detailed comparison with NB-IoT has been proposed, accounting for technical aspects, both at PHY and Link layers, and regulatory issues. In conclusion, the two technologies differ in many aspects and both have strengths and weaknesses. Depending on the specific application, the best solution can be identified based on the above-reported discussion and numerical results. Even though NB-IoT implements a more robust modulation and coding scheme, together with a highly reliable Link layer, it is heavily limited by larger energy consumption, not always suitable for specific IoT applications. Therefore, NB-IoT is more adequate for applications that are demanding in terms of reliability and network throughput. In addition, being not limited by any regulation in terms of duty cycle, devices can transmit more frequently or bigger data volumes. On the other hand, LoRaWAN is convenient for applications for battery-constrained use cases, where the reliability requirements can be relaxed.

At the very same time, results show that the network configurations (e.g., ADR support for LoRaWAN, or the CE level support and RACH configurations for NB-IoT) affect the performance of the considered technologies quite significantly. Notably, as shown in this thesis, either of the considered technology may outperform its counterpart according to the scenario considered. This is very important when considering the communication technology to be used for the particular use case scenario and it motivates more in-depth studies on the effects the different network parameters have on the technology performance as well as the development of relevant optimization mechanisms.

After considering static IoT scenarios with fixed terrestrial BSs, the thesis explored the role of UAVs in the network architecture.

Chapter 3 has been dedicated to UAV-aided NB-IoT. This chapter proposed a network performance evaluation accounting for different aspects of the NB-IoT protocol by considering different performance indicators, such as the access rate, latency, network throughput, and energy consumption. UABS speed and height have been proven to have a noticeable impact on the final performance, thus leading to important trade-offs on the different metrics. In addition, the choice of the trajectory, as one can expect, assumes a huge role, with the trajectory given by the TSP solution being the most suitable for clustered environments as the one presented in this chapter. However, this motivates the study of better mechanisms to design the trajectory to be used by the UABS, as it has been presented in the following chapters.

For the reasons just discussed, chapter 4 proposed different strategies to solve the trajectory design problem in UABS network, in particular when dealing with mobile users and V2X applications which have many differences with respect to the static IoT scenario considered before. At first, it has been proved that Q-Learning-based algorithms are able to generate good trajectories allowing the UABS to track and follow vehicles during their path. Specifically, the dueling architecture turns out to be the best solution for the target tracking problem, with a boost given by the inclusion of beamforming information. Secondly, a comparison between the use of a discrete or a continuous action space by means of 3DQN and DDPG algorithms has been carried out, by addressing their advantages and limitations. For this comparison, a priority-based mechanism has been introduced in the reward function to offer continuous service for specific high demanding V2X applications. Even if the pool of actions is more limited, due to its intrinsic limitation to use a discrete set, 3DQN outperforms DDPG, showing that a more complex algorithm and action space is not useful in the considered problem. In addition, in order to reduce the data requirements for RL-based training, this chapter proposed the use of a meta-RL solution to extract useful information from previously encountered traffic configurations, so that the agent can adapt faster to new tasks it has to perform. As a matter of fact, meta-RL can optimize the initial policy parameter vector so as to reduce the number of exploration steps during training.

Finally, chapter 5 proposed a RRM scheme in a UAV-aided vehicular network by analyzing beamforming and resource allocation at mmWaves frequencies. An ILP is proposed to solve such a problem with the goal of optimizing the RBs assignment between backhaul and access links as well as the beam pattern selection, considering jointly aerial-terrestrial operations. It proved that the presence of UABSs increases significantly the QoE for vehicular applications which require specific continuous services. In addition, the role of the backhaul capacity emerges as a limiting factor for the traffic handled by a UABS.

These last chapters presented a careful analysis of the use of UABSs proving how their introduction in the network architecture may be extremely interesting and worth exploring, especially for very demanding applications, such as V2X communications.

Appendix A

Appendix - A Sensing without Sensors System for Soil Moisture Estimation

A.1 Introduction

One of the application domains that benefited most from the growth of the IoT is smart agriculture. Indeed, according to the research community, agriculture products will have a very high demand by 2050 [158]; hence it is important to develop innovative approaches which can face such a trend. For this purpose, IoT and related technologies represent a potential solution to solve such problems. Furthermore, in smart agriculture, many applications can be developed to help farmers by automating and optimizing agricultural productivity[15, 159, 160, 161]. Among these, soil monitoring applications play a vital role. As a matter of fact, soil parameters, such as moisture and temperature, are essential for the life and the growth of plants, and, at the same time, their monitoring is helpful to avoid waste of resources, i.e., irrigation water [162].

In this appendix, an application for estimating soil moisture based on the propagation of electromagnetic waves in underground wireless communications is proposed. It is known, indeed, that sub-GHz radio waves can propagate underground with a loss rate that depends on the soil type and moisture content. Therefore, two radio transceivers buried at a fixed distance can exchange signals and make it possible to provide indirect estimates of soil moisture content without the need for ad-hoc sensors. For this purpose, the developed system, starting from the RSSI, can evaluate the soil moisture through properly tuned Machine Learning (ML) algorithms, realizing the Sensing-without-Sensors concept. More specifically, such a system can estimate the soil moisture of an entire volume instead of a single point, as most other commercially available humidity sensors do.

In this context, a good choice is a communication protocol based on LoRa working in the EU863-870 MHz band because it is expected that its waves can travel underground up to about 5-10 m. As explained before, LoRa is considered one of the most promising LPWAN technologies, which are technologies specifically designed to allow long-range communication while being very energy efficient. LoRa does not provide a high data rate; hence, it is helpful for applications that require sending a small amount of data periodically. Because

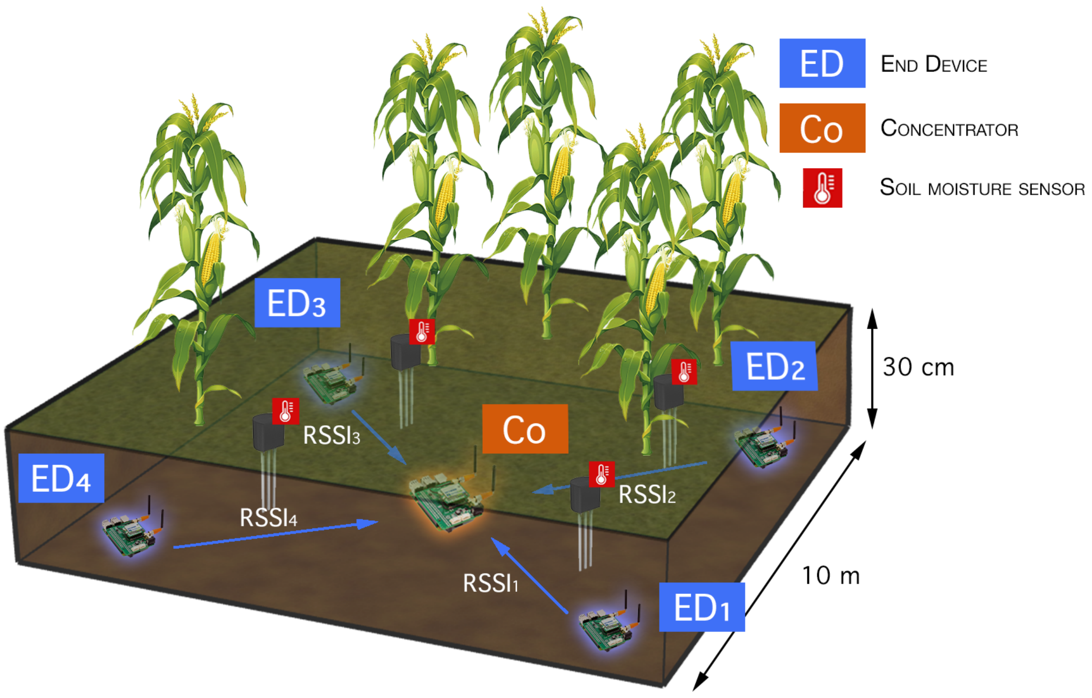


Figure A.1: Overview of the LoRa wireless network architecture used in the test-bed.

of all these reasons, it is deemed one of the best technologies to use in the smart agriculture domain.

The remainder of the chapter is organized as follows. In Section A.2, a review of the state of the art is presented. Section A.3 provides a detailed analysis of the system, whose preliminary results are presented in Section A.4. Finally, Section A.5 concludes the section.

A.2 Literature Overview

Different methods have been developed in order to design soil moisture prediction systems. Many works in the literature are based on Peplinski's principle [163], which is a semi-empirical model which takes into account the soil composure, the soil moisture, and the frequency to determine the complex permittivity of the water-soil mixture. Starting from this, authors in [164] and [165] derived path loss models for underground communication. In [166], authors carried out an experimental analysis of the underground communication for different sub-GHz bands, and they derived a model which characterizes the behavior of the channel by estimating the RSSI for different soil moisture levels. However, the definition of the soil moisture level from these models is not straightforward, and, in most cases, they cannot be applied to all the ranges of values of soil moisture. For these reasons, a ML-based approach is the most suitable since it turns out to be particularly useful when it is not possible to derive an analytical model.

Although the interest for ML techniques for precision agriculture has gained much attention during the last years [167], there are not so many systems specifically designed for precise soil moisture prediction. In [168], authors address different algorithms to predict the evolution of soil moisture for 1, 2, and 7 days ahead, starting from measurements collected in different fields. In the same way, in [169], deep Neural Network (NN) models are used to make predictions for the following days according to selected meteorological parameters. Another approach based on deep NNs is presented in [170], where authors exploit precipitation, air temperature, net radiation, and ground temperature data. A data mining system is presented in [171]. Such a system can gather weather predictions from several weather stations and use such data to predict the soil moisture for the next day. A sensing system named SoMoS is described in [172]. It is shown that by measuring the variation of the RSSI, it is possible to detect precipitation events that produced a significant variation in the soil moisture. Such a system has been evaluated in a laboratory environment [173]. Another approach based on RFID passive sensors is proposed in [174], where it is explained how variations in soil moisture modify the soil permittivity, which consequently affects the impedance match/mismatch between the tag chip and the sensor probe.

In contrast to the above-cited works, the goal in the following is to predict the soil moisture without the need for soil moisture sensors (if not for the training phase) or other types of data but based on the RSSI only, thus reducing the cost in comparison to other solutions implementing sensors over radio devices. Furthermore, the proposed design allows the estimation of the soil moisture of a volume and not of a single point. Thanks to this, there is no need to deploy many devices on the field, making the overall monitoring solution even cheaper. Finally, being the system based on LoRa, it is possible to monitor huge areas without the need for complex infrastructures, but with just a single GW, thanks to the coverage range such technology can provide.

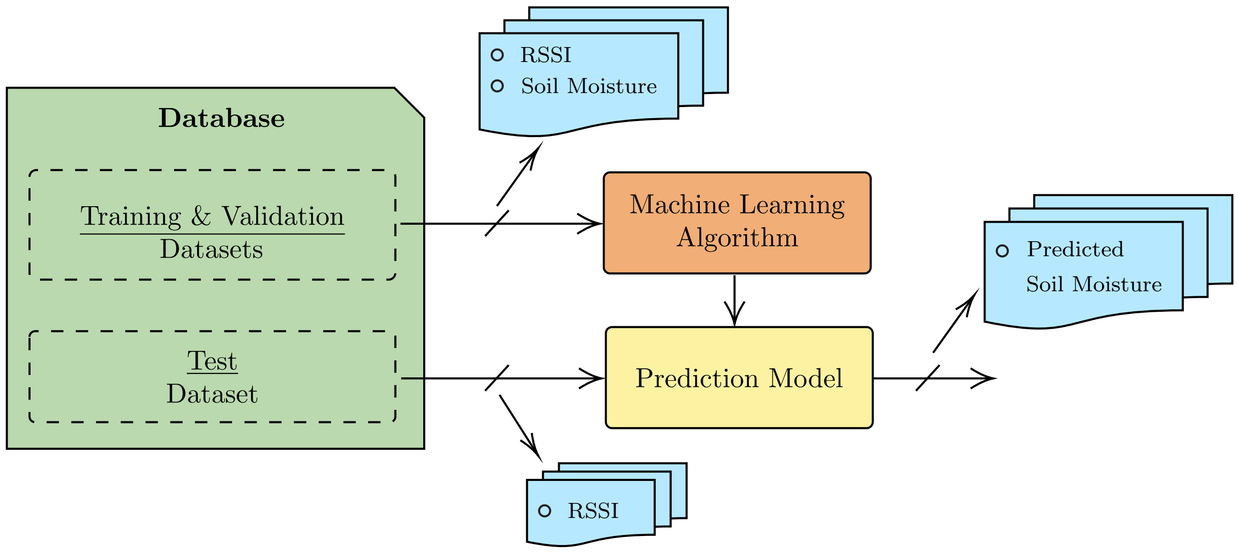


Figure A.2: Block scheme of the complete methodology.

A.3 Methodology

A.3.1 Sensing without Sensors System

The main element of the proposed system is called *tile*. Each tile consists of four EDs, located at the angles of a square area ($10\,\mathrm{m} \times 10\,\mathrm{m}$ in the experimental setup) and buried at $30\,\mathrm{cm}$ underground. EDs transmit data packets to a Concentrator, located in the center of the tile, which is equipped with one LoRa transceiver buried at $30\,\mathrm{cm}$, for receiving data from EDs, and one LoRa transceiver on the terrain surface to forward the measured RSSI to a GW located in a building nearby the experimental field. Such GW is connected to a NS where data are collected and processed. The devices are based on the Semtech SX1272 module, and they use a fixed frequency of $868.1\,\mathrm{MHz}$ for their transmissions. Data collected include the RSSI of the four links between every single ED and the Concentrator. The four EDs send dummy packets to the Concentrator via LoRa, which measures the RSSI of each transmission and periodically sends a message containing such information to the GW via LoRaWAN.

The ground truth soil moisture levels are measured inside the volume, in the four middle points which fall between the Coordinator and each ED, as reported in Fig A.1. To get such measurements, commercial sensors (Sentek Drill&Drop TriScan) have been used.

All the observed data are collected in a database where each entry is composed of:

- the average value of RSSI measured by the four EDs located in the squared tile;
- the average value of the soil moisture measured by the commercial probes;
- the corresponding timestamp of the instant when the measurements, both in terms of RSSI and soil moisture were taken.

The collected data are used to train, validate and test the ML algorithms which are briefly summarized in Section A.3.2. The approach is summarized in Figure A.2.

A.3.2 Machine Learning Algorithms

The ML task of predicting the value of one or more continuous target variables t given the value of an input vector x is widely known as regression. Given a training data set comprising N observations $\{\mathbf{x}_n\}$, where $n = 1, ..., N$ and $\mathbf{x}_n$ is a D-dimensional vector, together with corresponding target values t_n, the goal is to predict the value of t for a new observation $\mathbf{x}$. In the simplest approach, this can be done by directly constructing an appropriate function $y(\mathbf{x})$ whose values for new inputs $\mathbf{x}$ constitute the predictions, $\hat{t}$, for the corresponding values of t. Considering that the Concentrator measures the RSSI of each packet received by the 4 EDs, the n-th observation, x_n, can be expressed as the average RSSI, while t_n is the average soil moisture level of the considered four points. In this work, the regression problem is solved using two well-known ML algorithms:

– Decision Tree: it is a tree structure predictive model that goes from observations, represented in the branches of the tree, to conclusions about the target value, represented in the leaves. The algorithm is named regression tree if the predicted outcome can take continuous values (i.e., a real number). These algorithms are simple to understand and interpret, but they are usually not robust against small changes in the training dataset, which can lead to completely different tree structures. Despite this, when the provided training set is large enough, the algorithm shows remarkably high accuracy and low computational complexity.

– Neural Network: considering the case study of this work, the sought dependence between RSSI and soil moisture is not linear; therefore, a shallow NN has been chosen as a suitable regression algorithm [175]. During the training phase, the network tracks and approximates the function described by the inputs. Once the approximation is completed, it is possible to predict the soil moisture values according to the new RSSI observations.

A.4 Numerical Results

The trial was carried out in a 3-year-old commercial apple orchard at the experimental farm of the University of Bologna (Cadriano (BO), Italy, 44°33'03"N, 11°24'36"E, 33 m a.s.l.), of the variety Fuji grafted on M9 rootstock. Trees were grown on a silty clay loam soil (Haplic Calcisol soil, $20\,\mathrm{g}\ 100\,\mathrm{g}^{-1}$ sand, $42\,\mathrm{g}\ 100\,\mathrm{g}^{-1}$ silt, and $38\,\mathrm{g}\ 100\,\mathrm{g}^{-1}$ clay; pH = 7.6) at a density of 3030 tree ha^{-1} and managed as a spindle training system. The orchard was drip irrigated, and the floor management included herbicide strips along the row and grassed alleys regularly. It is worth noticing that the set of soil moisture values is not characterized by huge variations since the humidity of the field is kept under control for other activities carried out on it.

According to the procedure described in Section A.3.1, $N = 1152$ observations of average RSSI and corresponding soil moisture levels have been collected in the database. The set is made of data collected between April and July 2021.

To properly train and characterize the ML algorithms, the acquisitions have been randomly split in *training*, *validation* and *test* sets as $60\,\%$, $20\,\%$ and $20\,\%$ respectively, as depicted in Fig A.2. A single-hidden-layer feed-forward NN with 10 neurons and linear activation function in the hidden layer has been used, and the well-known *k-fold cross-validation* method has been chosen to avoid overfitting. The logical structure of the network is depicted in Figure A.3.

The metric adopted to measure the performance of the ML algorithms is the Root Mean Squared Error (RMSE), defined as

$$\mathsf{RMSE} = \sqrt{\frac{\sum_{i=1}^{N_\mathrm{t}} (\hat{t}_i - t_i)^2}{N_\mathrm{t}}}$$

where N_t is the number of points of the test dataset.

Figure A.4 and Figure A.5 show the results of the regression of the Decision Tree and the NN, respectively. The green points represent the real observations, t, while the red squares are the estimated soil moisture

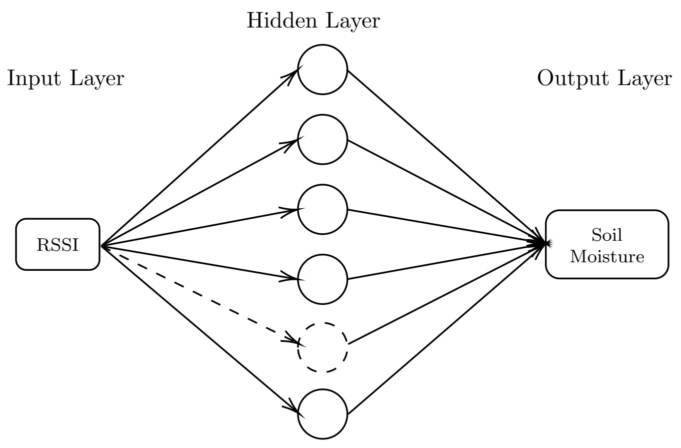


Figure A.3: Single-hidden-layer neural network considered for soil moisture prediction.

levels, $\hat{t}$. As depicted in the figures, both the tested algorithms successfully find a model to fit the relation between the RSSI and the respective soil moisture. In fact, both the algorithms show good performance with $\mathsf{RMSE} \approx 0.30\,\%$.

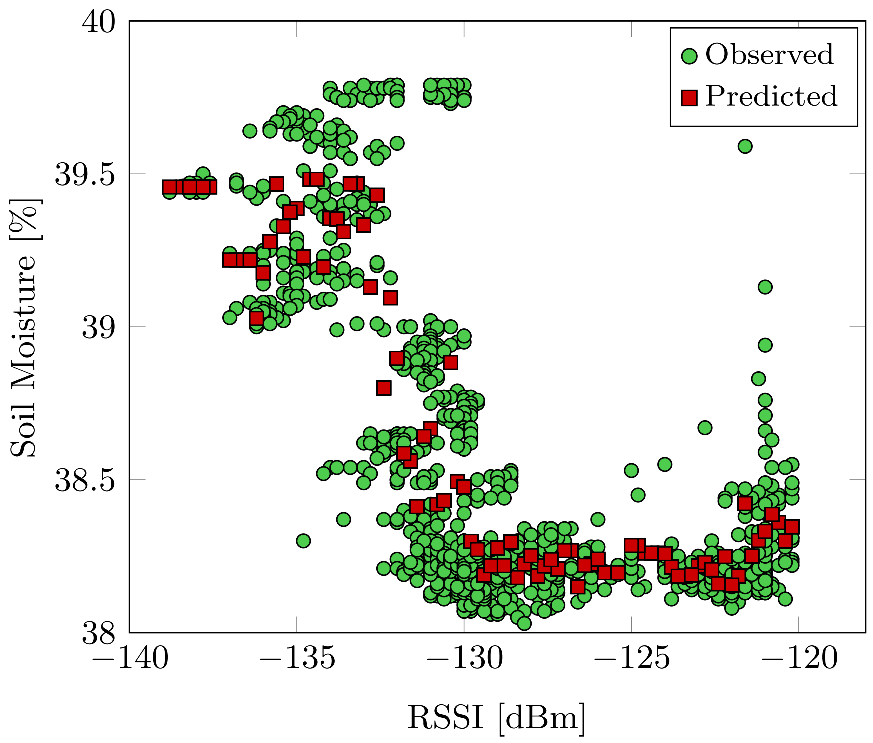


Figure A.4: Performance of the Decision Tree algorithm. In green the real observations; in red the estimated soil moisture levels.

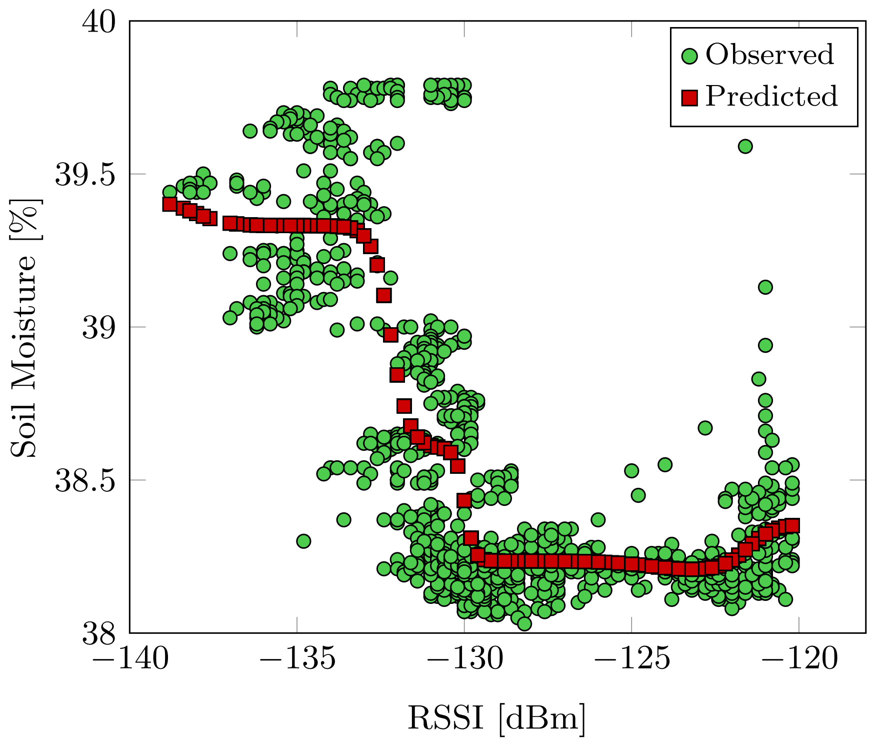


Figure A.5: Performance of the NN. In green the real observations; in red the estimated soil moisture levels.

A.5 Conclusion

In this appendix, an innovative system able to estimate the soil moisture in a field using the RSSI measurements to model the underground propagation via a machine learning approach has been presented. The system's novelty lies in the concept of Sensing-without-Sensors approach, which makes the estimation possible without the need for actual sensors. Furthermore, the proposed solution is able to estimate the soil moisture of an entire volume instead of a pointwise measurement, as commercial systems usually do.

List of Publications

The work presented in this thesis has led to the following publications in international conferences, journals and books:

- [176] Chiara Buratti, Konstantin Mikhaylov, Riccardo Marini and Roberto Verdone, ”Low-power Wide-Area Networks: A Comparative Analysis Between LoRaWAN and NB-IoT”, *CNIT Technical Report-05: Internet of Things: Technologies, Challenges and Impact*, 2020
- [177] Riccardo Marini, Walter Cerroni and Chiara Buratti, ”A Novel Collision-Aware Adaptive Data Rate Algorithm for LoRaWAN Networks”, *IEEE Internet of Things Journal*, 2021, vol. 8, num. 4, pp. 2670-2680. doi:10.1109/JIOT.2020.3020189
- [178] Riccardo Marini, Konstantin Mikhaylov, Gianni Pasolini and Chiara Buratti, ”LoRaWANSim: A Flexible Simulator for LoRaWAN Networks”, *Sensors*, 2021, vol. 21, num. 3, doi:10.3390/s21030695
- [179] Silvia Mignardi, Riccardo Marini, Roberto Verdone and Chiara Buratti, ”On the Performance of a UAV-Aided Wireless Network Based on NB-IoT”, *Drones*, 2021, vol. 5, num. 3, doi:10.3390/drones5030094
- [180] Riccardo Marini, Enrico Testi, Chiara Buratti, Andrea Giorgetti and Roberto Verdone, ”A Sensing-without-Sensors System for Soil Moisture Estimation”, in *Proc. 2021 IEEE MetroAgriFor*, 2021, doi:10.1109/MetroAgriFor52389.2021.9628635
- [181] Riccardo Marini, Konstantin Mikhaylov, Gianni Pasolini and Chiara Buratti, ”Low-Power Wide-Area Networks: Comparison of LoRaWAN and NB-IoT Performance”, *IEEE Internet of Things Journal*, 2022, vol. 9, no. 21, pp. 21051-21063, doi: 10.1109/JIOT.2022.3176394.
- [182] Silvia Mignardi, Danila Ferretti, Riccardo Marini, Francesca Conserva, Stefania Bartoletti, Roberto Verdone and Chiara Buratti, ”Optimizing Beam Selection and Resource Allocation in UAV-Aided Vehicular Networks”, in *Proc. 2022 Joint European Conference on Networks and Communications/6G Summit (EuCNC/6G Summit)*, 2022, doi:10.1109/EuCNC/6GSummit54941.2022.9815631
- [183] Riccardo Marini, Leonardo Spampinato, Silvia Mignardi, Roberto Verdone and Chiara Buratti, ”Reinforcement Learning-Based Trajectory Planning For UAV-aided Vehicular Communications”, in *Proc. 2022 IEEE 30th European Signal Processing Conference (EUSIPCO)*, 2022

- [184] Riccardo Marini, Sangwoo Park, Osvaldo Simeone and Chiara Buratti, ”Continual Meta-Reinforcement Learning for UAV-Aided Vehicular Wireless Networks”, *accepted at ICC 2023*
- Leonardo Spampinato, Alessia Tarozzi, Chiara Buratti and Riccardo Marini, ”Deep RL path planning models for UAV- Aided networks offering V2X services: Comparing discrete to continuous action spaces”, *submitted to ICASSP 2023*

Bibliography

[1] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications”. In: *IEEE Communications Surveys Tutorials* 17.4 (2015), pp. 2347–2376.

[2] P. Datta and B. Sharma. “A survey on IoT architectures, protocols, security and smart city based applications”. In: *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. 2017, pp. 1–5.

[3] M. Dholu and K. A. Ghodinde. “Internet of Things (IoT) for Precision Agriculture Application”. In: *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*. 2018, pp. 339–342.

[4] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen. “Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems”. In: *Proc. IEEE* 104.5 (2016), pp. 1013–1024.

[5] Carles Gomez, Joaquim Oller, and Josep Paradells. “Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology”. In: *Sensors* 12.9 (2012), pp. 11734–11753. URL: https://www.mdpi.com/1424-8220/12/9/11734.

[6] Matti Siekkinen, Markus Hiienkari, Jukka K. Nurminen, and Johanna Nieminen. “How low energy is bluetooth low energy? Comparative measurements with ZigBee/802.15.4”. In: *Proc. IEEE Wireless Communications and Networking Conference Workshops* (2012), pp. 232–237.

[7] Johnathan Robertson. “The need for low cost, high reach, wide area connectivity for the internet of things: A mobile network operator’s perspective”. In: *Machina Research* (2014).

[8] Statista. *Number of LPWAN connections by technology worldwide from 2017 to 2023*. [Online] Available: https://www.statista.com/statistics/880822/lpwan-ic-market-share-by-technology/. 2021.

[9] LoRa Alliance. *LoRaWAN Specification (V1.0.2)*. 2016.

[10] 5GAA. “A visionary roadmap for advanced driving use cases, connectivity technologies, and radio spectrum needs”. In: *White Paper* (Sept. 2020).

[11] Gorka Velez, Angel Martin, Giancarlo Pastor, and Edward Mutafungwa. “5G Beyond 3GPP Release 15 for Connected Automated Mobility in Cross-Border Contexts”. In: *Sensors* 20.22 (2020).

[12] Junil Choi, Vutha Va, Nuria Gonzalez-Prelcic, Robert Daniels, Chandra R. Bhat, and Robert W. Heath. “Millimeter-Wave Vehicular Communication to Support Massive Automotive Sensing”. In: *IEEE Communications Magazine* 54.12 (2016), pp. 160–167.

[13] Barbara M. Masini, Alessandro Bazzi, and Alberto Zanella. “A Survey on the Roadmap to Mandate on Board Connectivity and Enable V2V-Based Vehicular Sensor Networks”. In: *Sensors* 18.7 (2018).

[14] ETSI. “5G; Service requirements for enhanced V2X scenarios”. In: *ETSI TS 22.186 version 16.2.0* (Nov. 2020).

[15] L. Feltrin, G. Tsoukaneri, M. Condoluci, C. Buratti, T. Mahmoodi, M. Dohler, and R. Verdone. “Narrowband IoT: A Survey on Downlink and Uplink Perspectives”. In: *IEEE Wireless Communications* 26.1 (2019), pp. 78–86.

[16] L. Krupka, L. Vojtech, and M. Neruda. “The issue of LPWAN technology coexistence in IoT environment”. In: *2016 17th International Conference on Mechatronics - Mechatronika (ME)* (2016), pp. 1–8.

[17] J. de Carvalho Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic, and A. L. L. Aquino. “LoRaWAN — A low power WAN protocol for Internet of Things: A review and opportunities”. In: *2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech)* (2017), pp. 1–6.

[18] Rashmi Sharan Sinha, Yiqiao Wei, and Seung-Hoon Hwang. “A survey on LPWA technology: LoRa and NB-IoT”. In: *ICT Express* 3.1 (2017), pp. 14–21. URL: https://www.sciencedirect.com/science/article/pii/S2405959517300061.

[19] LoRa Alliance. *LoRaWAN 1.1 Regional Parameters*. 2017.

[20] Thomas Janssen, Noori BniLam, Michiel Aernouts, Rafael Berkvens, and Maarten Weyn. “LoRa 2.4 GHz Communication Link and Range”. In: *Sensors* 20.16 (2020). URL: https://www.mdpi.com/1424-8220/20/16/4366.

[21] L. Feltrin, C. Buratti, E. Vinciarelli, R. De Bonis, and R. Verdone. “LoRaWAN: Evaluation of Link- and System-Level Performance”. In: *IEEE Internet of Things Journal* 5.3 (2018), pp. 2249–2258.

[22] M. Slabicki, G. Premsankar, and M. Di Francesco. “Adaptive configuration of LoRa networks for dense IoT deployments”. In: 2018, pp. 1–9.

[23] V. Hauser and T. Hégr. “Proposal of Adaptive Data Rate Algorithm for LoRaWAN-Based Infrastructure”. In: *2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)* (2017), pp. 85–90.

[24] Martin Stusek, Dmitri Moltchanov, Pavel Masek, Konstantin Mikhaylov, Otto Zeman, Martin Roubicek, Yevgeni Koucheryavy, and Jiri Hosek. “Accuracy Assessment and Cross-Validation of LPWAN Propagation Models in Urban Scenarios”. In: *IEEE Access* 8 (2020), pp. 154625–154636.

[25] Semtech Corporation. *LoRa Modulation Basics*. [Online] Available: http://wiki.lahoud.fr/lib/exe/fetch.php?media=an1200.22.pdf. 2015.

[26] Semtech Corporation. *SX1272/3/6/7/8: LoRa Modem Design Guide*. 2014.

[27] Semtech Corporation. *LoRaWAN – simple rate adaptation recommended algorithm*. 2016.

[28] Gianni Pasolini, Chiara Buratti, Luca Feltrin, Flavio Zabini, Cristina De Castro, Roberto Verdone, and Oreste Andrisano. “Smart City Pilot Projects Using LoRa and IEEE802.15.4 Technologies”. In: *Sensors* 18.4 (2018). URL: https://www.mdpi.com/1424-8220/18/4/1118.

[29] J. M. Marais, A. M. Abu-Mahfouz, and G. P. Hancke. “A Review of LoRaWAN Simulators: Design Requirements and Limitations”. In: *2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC)* (2019), pp. 1–6.

[30] Martin Bor, Utz Roedig, Thiemo Voigt, and Juan Alonso. “Do LoRa Low-Power Wide-Area Networks Scale?” In: 2016, pp. 59–67.

[31] Alexandru-Ioan Pop, Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. “Does Bidirectional Traffic Do More Harm Than Good in LoRaWAN Based LPWA Networks?” In: *CoRR* abs/1704.04174 (2017). URL: http://arxiv.org/abs/1704.04174.

[32] NS-3 Consortium. *ns-3*. [Online] Available: https://www.nsnam.org/. 2020.

[33] D. Magrin, M. Centenaro, and L. Vangelista. “Performance evaluation of LoRa networks in a smart city scenario”. In: 2017, pp. 1–7.

[34] M. Capuzzo, D. Magrin, and A. Zanella. “Confirmed traffic in LoRaWAN: Pitfalls and countermeasures”. In: 2018, pp. 1–7.

[35] K. Q. Abdelfadeel, D. Zorbas, V. Cionca, and D. Pesch. “$FREE$ —Fine-Grained Scheduling for Reliable and Energy-Efficient Data Collection in LoRaWAN”. In: *IEEE Internet of Things Journal* 7.1 (2020), pp. 669–683.

[36] G. Callebaut, G. Ottoy, and L. van der Perre. “Cross-Layer Framework and Optimization for Efficient Use of the Energy Budget of IoT Nodes”. In: *2019 IEEE Wireless Communications and Networking Conference (WCNC)* (2019), pp. 1–6.

[37] 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.

[38] A. Waret, M. Kaneko, A. Guitton, and N. El Rachkidy. “LoRa Throughput Analysis With Imperfect Spreading Factor Orthogonality”. In: *IEEE Wireless Communications Letters* 8.2 (2019), pp. 408–411.

[39] S. Kim, Y. Yoo. “Contention-Aware Adaptive Data Rate for Throughput Optimization in LoRaWAN”. In: *Sensors* 18.6 (2018).

[40] F. Cuomo, M. Campo, A. Caponi, G. Bianchi, G. Rossini, and P. Pisani. “EXPLoRa: Extending the performance of LoRa by suitable spreading factor allocations”. In: *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2017, pp. 1–8.

[41] V. K. Sarker, J. P. Queralta, T. N. Gia, H. Tenhunen, and T. Westerlund. “A Survey on LoRa for IoT: Integrating Edge Computing”. In: *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*. Rome, Italy, 2019, pp. 295–300.

[42] H. Truong. “Enabling Edge Analytics of IoT Data: the Case of LoRaWAN”. In: *2018 Global Internet of Things Summit (GIoTS)*. Bilbao, Spain, 2018.

[43] Tuan Nguyen Gia, Jorge Peña Queralta, and Tomi Westerlund. “Exploiting LoRa, edge, and fog computing for traffic monitoring in smart cities”. In: *LPWAN Technologies for IoT and M2M Applications*. Ed. by Bharat S. Chaudhari and Marco Zennaro. Academic Press, 2020, pp. 347–371.

[44] Paula Fraga-Lamas, Mikel Celaya-Echarri, Peio Lopez-Iturri, Luis Castedo, Leyre Azpilicueta, Erik Aguirre, Manuel Suárez-Albela, Francisco Falcone, and Tiago M. Fernández-Caramés. “Design and Experimental Validation of a LoRaWAN Fog Computing Based Architecture for IoT Enabled Smart Campus Applications”. In: *Sensors* 19.15 (2019), p. 3287. URL: http://dx.doi.org/10.3390/s19153287.

[45] Gianluca Davoli, Walter Cerroni, Slavica Tomovic, Chiara Buratti, Chiara Contoli, and Franco Callegati. “Intent-based service management for heterogeneous software-defined infrastructure domains”. In: *International Journal of Network Management* 29.1 (2019), e2051.

[46] J. Pan and J. McElhannon. “Future Edge Cloud and Edge Computing for Internet of Things Applications”. In: *IEEE Internet of Things Journal* 5.1 (2018), pp. 439–449.

[47] M. Hata. “Empirical formula for propagation loss in land mobile radio services”. In: *IEEE Transactions on Vehicular Technology* 29.3 (1980), pp. 317–325.

[48] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, and I. Tinnirello. “Impact of LoRa Imperfect Orthogonality: Analysis of Link-Level Performance”. In: *IEEE Communications Letters* 22.4 (2018), pp. 796–799.

[49] Lluís Casals, Bernat Mir, Rafael Vidal, and Carles Gomez. “Modeling the Energy Performance of LoRaWAN”. In: *Sensors* 17.10 (2017), p. 2364. URL: http://dx.doi.org/10.3390/s17102364.

[50] J. Petäjäjärvi, K. Mikhaylov, M. Hämäläinen, and J. Iinatti. “Evaluation of LoRa LPWAN technology for remote health and wellbeing monitoring”. In: *2016 10th International Symposium on Medical Information and Communication Technology (ISMICT)* (2016), pp. 1–5.

[51] Microchip Technology Inc. *RN2483 Low-Power Long Range LoRa Technology Transceiver Module*. http://ww1.microchip.com/downloads/en/DeviceDoc/RN2483-Low-Power-Long-Range-LoRa-Technology-Transceiver-Module-Data-Sheet-DS50002346D.pdf. 2015.

[52] O. Afisiadis, M. Cotting, A. Burg, and A. Balatsoukas-Stimming. “On the Error Rate of the LoRa Modulation With Interference”. In: *IEEE Transactions on Wireless Communications* 19.2 (2020), pp. 1292–1304.

[53] G. Ferré and A. Giremus. “LoRa Physical Layer Principle and Performance Analysis”. In: *Proc. IEEE International Conference on Electronics, Circuits and Systems* (2018), pp. 65–68.

[54] Norman Abramson. *The Aloha System: Another Alternative for Computer Communications*. Houston, Texas, 1970. URL: https://doi.org/10.1145/1478462.1478502.

[55] M. Aydinlik and M. Salehi. “16-QAM turbo coded modulation for the Gilbert-Elliot channel model”. In: *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*. Vol. 3. 2002, 1393–1397 vol.3.

[56] M. Chiani, E. Milani, and R. Verdone. “A semi-analytical approach for performance evaluation of TCP-IP based mobile radio links”. In: *Globecom ’00 - IEEE. Global Telecommunications Conference. Conference Record (Cat. No.00CH37137)*. Vol. 2. 2000, 937–942 vol.2.

[57] D. Bankov, E. Khorov, and A. Lyakhov. “Mathematical model of LoRaWAN channel access with capture effect”. In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. 2017, pp. 1–5.

[58] O. Liberg, M. Sundberg, E. W. Johan, B. J. Sachs. *Cellular Internet of Things: Technologies, Standards, and Performance*. Elsevier, 2017.

[59] Mads Lauridsen, Huan Nguyen, Benny Vejlgaard, Istvan Z. Kovacs, Preben Mogensen, and Mads Sorensen. “Coverage Comparison of GPRS, NB-IoT, LoRa, and SigFox in a 7800 km² Area”. In: *Proc. VTC Spring*. 2017, pp. 1–5.

[60] Benny Vejlgaard, Mads Lauridsen, Huan Nguyen, Istvan Z. Kovacs, Preben Mogensen, and Mads Sorensen. “Coverage and Capacity Analysis of Sigfox, LoRa, GPRS, and NB-IoT”. In: 2017, pp. 1–5.

[61] Guus Leenders, Gilles Callebaut, Geoffrey Ottoy, Liesbet Van der Perre, and Lieven De Strycker. “Multi-RAT for IoT: The Potential in Combining LoRaWAN and NB-IoT”. In: *IEEE Communications Magazine* 59.12 (2021), pp. 98–104.

[62] Lucas Ribeiro, Davi Tokikawa, João Rebelatto, and Glauber Brante. “Comparison between LoRa and NB-IoT coverage in urban and rural Southern Brazil regions”. In: *Annals of Telecommunications* 75 (2020).

[63] Antonella Lombardo, Stefano Parrino, Giacomo Peruzzi, and Alessandro Pozzebon. “LoRaWAN Versus NB-IoT: Transmission Performance Analysis Within Critical Environments”. In: *IEEE Internet of Things Journal* 9.2 (2022), pp. 1068–1081.

[64] Martin Stusek, Dmitri Moltchanov, Pavel Masek, Konstantin Mikhaylov, Jiri Hosek, Sergey Andreev, Yevgeni Koucheryavy, Pavel Kustarev, Otto Zeman, and Martin Roubicek. “LPWAN Coverage Assessment Planning Without Explicit Knowledge of Base Station Locations”. In: *IEEE Internet of Things Journal* 9.6 (2022), pp. 4031–4050.

[65] H. Mroue, A. Nasser, S. Hamrioui, B. Parrein, E. Motta-Cruz, and G. Rouyer. “MAC layer-based evaluation of IoT technologies: LoRa, SigFox and NB-IoT”. In: *Proc. 2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*. 2018, pp. 1–5.

[66] Franck Muteba, Karim Djouani, and Thomas Olwal. “A comparative Survey Study on LPWA IoT Technologies: Design, considerations, challenges and solutions”. In: *Procedia Computer Science* 155 (2019), pp. 636–641. URL: https://www.sciencedirect.com/science/article/pii/S1877050919310063.

[67] Ala’ Khalifeh, Khaled Aldahdouh Aldahdouh, Khalid A. Darabkh, and Waleed Al-Sit. “A Survey of 5G Emerging Wireless Technologies Featuring LoRaWAN, Sigfox, NB-IoT and LTE-M”. In: 2019, pp. 561–566.

[68] Samuela Persia, Claudia Carciofi, and Manuel Faccioli. “NB-IoT and LoRA connectivity analysis for M2M/IoT smart grids applications”. In: *Proc. IEEE AEIT International Annual Conf.* (2017), pp. 1–6.

[69] Yandja Lalle, Lamia Chaari Fourati, Mohamed Fourati, and João Paulo Barraca. “A Comparative Study of LoRaWAN, SigFox, and NB-IoT for Smart Water Grid”. In: 2019, pp. 1–6.

[70] 3GPP. *3GPP TS 36.300 Group Radio Access Network. Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN). Overall description*. 2018.

[71] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. “A comparative study of LPWAN technologies for large-scale IoT deployment”. In: *ICT Express* 5.1 (2019), pp. 1–7. URL: https://www.sciencedirect.com/science/article/pii/S2405959517302953.

[72] Luca Feltrin, Massimo Condoluci, Toktam Mahmoodi, Mischa Dohler, and Roberto Verdone. “NB-IoT: Performance Estimation and Optimal Configuration”. In: *European Wireless 2018; 24th European Wireless Conference* (2018), pp. 1–6.

[73] ETSI. *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification (3GPP TS 36.331 version 15.3.0 Release 15*. 2018.

[74] Mathworks. *LTE Toolbox*.

[75] Changsheng Yu, Li Yu, Yuan Wu, Yanfei He, and Qun Lu. “Uplink Scheduling and Link Adaptation for Narrowband Internet of Things Systems”. In: *IEEE Access* 5 (2017), pp. 1724–1734.

[76] Huikang Li, Gonglong Chen, Yihui Wang, Yi Gao, and Wei Dong. “Accurate Performance Modeling of Uplink Transmission in NB-IoT”. In: 2018, pp. 910–917.

[77] 3GPP TSG RAN Meeting 69. *Narrowband IoT*.

[78] u-blox. *SARA-N2 series*.

[79] Silvia Mignardi, Konstantin Mikhaylov, Valentina Cacchiani, Roberto Verdone, and Chiara Buratti. "Unmanned Aerial Base Stations for NB-IoT: Trajectory Design and Performance Analysis". In: *Proc. IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE. 2020, pp. 1–6.

[80] 3GPP TS 22.289 V17.1.0. "Enhancement for Unmanned Aerial Vehicles". In: (2019). URL: http://www.3gpp.org/ftp//Specs/archive/22%5C_series/22.125/22125-h10.zip.

[81] 3GPP TS 22.125 V17.1.0. "Unmanned Aerial System (UAS) support in 3GPP". In: (2019). URL: http://www.3gpp.org/ftp//Specs/archive/22%5C_series/22.125/22125-h10.zip.

[82] Nian Xia, Hsiao-Hwa Chen, and Chu-Sing Yang. "Emerging technologies for machine-type communication networks". In: *IEEE Network* 34.1 (2019), pp. 214–222.

[83] Akram Al-Hourani, Sithamparanathan Kandeepan, and Abbas Jamalipour. "Modeling air-to-ground path loss for low altitude platforms in urban environments". In: *Proc. IEEE Global Communications Conference*. IEEE. 2014.

[84] A. Al-Hourani, S. Kandeepan, and S. Lardner. "Optimal LAP Altitude for Maximum Coverage". In: *IEEE Wireless Communications Letters* 3.6 (2014), pp. 569–572.

[85] E. Yanmaz. "Connectivity versus area coverage in unmanned aerial vehicle networks". In: *Proc. IEEE International Conference on Communications (ICC)*. 2012, pp. 719–723.

[86] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah. "Mobile Unmanned Aerial Vehicles (UAVs) for Energy-Efficient Internet of Things Communications". In: *IEEE Transactions on Wireless Communications* 16.11 (2017), pp. 7574–7589.

[87] Yong Zeng and Rui Zhang. "Energy-efficient UAV communication with trajectory optimization". In: *IEEE Transactions on Wireless Communications* 16.6 (2017), pp. 3747–3760.

[88] J. Lu, S. Wan, X. Chen, Z. Chen, P. Fan, and K. B. Letaief. "Beyond Empirical Models: Pattern Formation Driven Placement of UAV Base Stations". In: *IEEE Transactions on Wireless Communications* 17.6 (2018), pp. 3641–3655.

[89] K. Song, J. Zhang, Z. Ji, J. Jiang, and C. Li. "Energy-Efficiency for IoT System With Cache-Enabled Fixed-Wing UAV Relay". In: *IEEE Access* 8 (2020), pp. 117503–117512.

[90] Almudena Diaz Zayas and Pedro Merino. "The 3GPP NB-IoT system architecture for the Internet of Things". In: *Proc. IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE. 2017, pp. 277–282.

[91] Q. Wu, Y. Zeng, and R. Zhang. "Joint Trajectory and Communication Design for Multi-UAV Enabled Wireless Networks". In: *IEEE Transactions on Wireless Communications* 17.3 (2018), pp. 2109–2121.

[92] Q. Wu and R. Zhang. "Common Throughput Maximization in UAV-Enabled OFDMA Systems with Heterogeneous Delay Requirements". In: *IEEE Transactions on Wireless Communications* (2018).

[93] Han-Ting Ye, Xin Kang, Jingon Joung, and Ying-Chang Liang. “Joint uplink and downlink 3D optimization of an UAV swarm for wireless-powered NB-IoT”. In: *Proc. IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2019, pp. 1–6.

[94] G. Castellanos, M. Deruyck, L. Martens, and W. Joseph. “System Assessment of WUSN Using NB-IoT UAV-Aided Networks in Potato Crops”. In: *IEEE Access* 8 (2020), pp. 56823–56836.

[95] M. Haenggi. *Stochastic Geometry for Wireless Networksk*. U.K.: Cambridge Univ. Press, 2012.

[96] C. Saha, M. Afshang, and H. S. Dhillon. “3GPP-Inspired HetNet Model Using Poisson Cluster Process: Sum-Product Functionals and Downlink Coverage”. In: *IEEE Trans. Commun.* 66.5 (2018), pp. 2219–2234.

[97] Jad Hajj et al. “Telecom operators in the age of drones: Preparing for the new era”. In: *strategy& Tech. Report* (2017). Online at https://www.strategyand.pwc.com/media/file/Telecom-operators-in-the-age-of-drones.pdf.

[98] Luke Dormehl. *7 Drones that can stay airborne for hours — and the tech that makes it possible*. URL: https://www.digitaltrends.com/cool-tech/drones-with-super-long-flight-times/.

[99] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, and Merouane Debbah. “Drone Small Cells in the Clouds: Design, Deployment and Performance Analysis”. In: *Proc. IEEE Global Communications Conference (GLOBECOM)*. 2015, pp. 1–6.

[100] George Dantzig, Ray Fulkerson, and Selmer Johnson. “Solution of a large-scale traveling-salesman problem”. In: *J. Operations Research Soc. of Amer.* 2.4 (1954), pp. 393–410.

[101] Mohamed-Amine Lahmeri, Mustafa A. Kishk, and Mohamed-Slim Alouini. “Artificial Intelligence for UAV-Enabled Wireless Networks: A Survey”. In: *IEEE Open Journal of the Communications Society* 2 (2021), pp. 1015–1040. URL: http://dx.doi.org/10.1109/OJCOMS.2021.3075201.

[102] Aicha Idriss Hentati and Lamia Chaari Fourati. “Comprehensive survey of UAVs communication networks”. In: *Computer Standards and Interfaces* 72 (2020), p. 103451. URL: https://www.sciencedirect.com/science/article/pii/S0920548919303411.

[103] M. H. C. Garcia, A. Molina-Galan, M. Boban, J. Gozalvez, B. Coll-Perales, T. Şahin, and A. Kousaridas. “A Tutorial on 5G NR V2X Communications”. In: *IEEE Communications Surveys Tutorials* (2021), pp. 1–1.

[104] Uygar Demir, Cenk Toker, and Ozgur Ekici. “Energy-Efficient Deployment of UAV in V2X Network Considering Latency and Backhaul Issues”. In: *BlackSeaCom*. May 2020.

[105] Houari, Abdeslam and Mazri, Tomader. “Improving V2X-6G network capacity using a new UAV-based approach in a Cloud/ICN architecture, case Study: VANET network”. In: *E3S Web Conf.* 297 (2021), p. 01019.

[106] Bodong Shang, Lingjia Liu, Junchao Ma, and Pingzhi Fan. “Unmanned Aerial Vehicle Meets Vehicle-to-Everything in Secure Communications”. In: *IEEE Communications Magazine* 57.10 (2019), pp. 98–103.

[107] Laurent Kloeker, Tobias Moers, Lennart Vater, Adrian Zlocki, and Lutz Eckstein. “Utilization and Potentials of Unmanned Aerial Vehicles (UAVs) in the Field of Automated Driving: A Survey”. In: *Proc. ICVISP*. Dec. 2021.

[108] Jinna Hu, Chen Chen, Lin Cai, Mohammad R. Khosravi, Qingqi Pei, and Shaohua Wan. “UAV-Assisted Vehicular Edge Computing for the 6G Internet of Vehicles: Architecture, Intelligence, and Challenges”. In: *IEEE Communications Standards Magazine* 5.2 (2021), pp. 12–18.

[109] Petros S. Bithas, Emmanouel T. Michailidis, Nikolaos Nomikos, Demosthenes Vouyioukas, and Athanasios G. Kanatas. “A Survey on Machine-Learning Techniques for UAV-Based Communications”. In: *Sensors* 19.23 (2019). URL: https://www.mdpi.com/1424-8220/19/23/5170.

[110] Sebastian Thrun. “Lifelong learning algorithms”. In: *Learning to learn*. Springer, 1998, pp. 181–209.

[111] Chelsea Finn, Pieter Abbeel, and Sergey Levine. “Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks”. In: *Proc. ICML*. Sydney, Australia, Aug. 2017.

[112] Glen Berseth, Zhiwei Zhang, Grace Zhang, Chelsea Finn, and Sergey Levine. “CoMPS: Continual Meta Policy Search”. In: *Proc. NeurIPS*. 2021.

[113] Bo Zhang, Jie Hu, Qin Yu, and Kun Yang. “Trajectory Design of UAV Aided Wireless Information and Energy Provision”. In: *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*. IEEE 94th Vehicular Technology Conference. ISSN: 2577-2465. Sept. 2021, pp. 1–5.

[114] Dinh-Hieu Tran, Thang X. Vu, Symeon Chatzinotas, Shahram ShahbazPanahi, and Björn Ottersten. “Coarse Trajectory Design for Energy Minimization in UAV-Enabled”. In: *IEEE Transactions on Vehicular Technology* 69.9 (2020), pp. 9483–9496.

[115] Jongyul Lee and Vasilis Friderikos. “Interference-aware path planning optimization for multiple UAVs in beyond 5G networks”. In: *Journal of Communications and Networks* 24.2 (2022), pp. 125–138.

[116] Seongah Jeong, Osvaldo Simeone, and Joonhyuk Kang. “Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning”. In: *IEEE Transactions on Vehicular Technology* 67.3 (2017), pp. 2049–2063.

[117] Mirco Theile, Harald Bayerlein, Richard Nai, David Gesbert, and Marco Caccamo. “UAV Path Planning using Global and Local Map Information with Deep Reinforcement Learning”. In: *Proc. ICAR*. Ljubljana, Slovenia, Dec. 2021.

[118] Harald Bayerlein, Mirco Theile, Marco Caccamo, and David Gesbert. “UAV Path Planning for Wireless Data Harvesting: A Deep Reinforcement Learning Approach”. In: *Proc. Globecom*. Taipei, Taiwan, Dec. 2020.

[119] Jingzhi Hu, Hongliang Zhang, Lingyang Song, Zhu Han, and H. Vincent Poor. “Reinforcement Learning for a Cellular Internet of UAVs: Protocol Design, Trajectory Control, and Resource Management”. In: *IEEE Wireless Communications* 27.1 (2020), pp. 116–123.

[120] Md Moin Uddin Chowdhury, Fatih Erden, and Ismail Guvenc. “RSS-Based Q-Learning for Indoor UAV Navigation”. In: 2019, pp. 121–126.

[121] Harald Bayerlein, Mirco Theile, Marco Caccamo, and David Gesbert. “Multi-UAV Path Planning for Wireless Data Harvesting With Deep Reinforcement Learning”. In: *IEEE Open Journal of the Communications Society* PP (May 2021), pp. 1–1.

[122] Lijun Deng, Gang Wu, Jingwei Fu, Yizhong Zhang, and Yifu Yang. “Joint Resource Allocation and Trajectory Control for UAV-Enabled Vehicular Communications”. In: *IEEE Access* 7 (2019), pp. 132806–132815.

[123] Xiao Liu, Yuanwei Liu, and Yue Chen. “Reinforcement Learning in Multiple-UAV Networks: Deployment and Movement Design”. In: *IEEE Transactions on Vehicular Technology* 68.8 (2019), pp. 8036–8049.

[124] Bote Jiang, Sidney N. Givigi, and Jean-Alexis Delamer. “A MARL Approach for Optimizing Positions of VANET Aerial Base-Stations on a Sparse Highway”. In: *IEEE Access* 9 (2021), pp. 133989–134004.

[125] Moataz Samir, Dariush Ebrahimi, Chadi Assi, Sanaa Sharafeddine, and Ali Ghrayeb. “Trajectory Planning of Multiple Dronecells in Vehicular Networks: A Reinforcement Learning Approach”. In: *IEEE Networking Letters* 2.1 (2020), pp. 14–18.

[126] Xiao Liu, Yuanwei Liu, Yue Chen, and Lajos Hanzo. “Trajectory Design and Power Control for Multi-UAV Assisted Wireless Networks: A Machine Learning Approach”. In: *IEEE Transactions on Vehicular Technology* 68.8 (2019), pp. 7957–7969.

[127] Paulo Valente Klaine, João Nadas, Richard Souza, and Muhammad Imran. “Distributed Drone Base Station Positioning for Emergency Cellular Networks Using Reinforcement Learning”. In: *Cognitive Computation* 10 (Oct. 2018).

[128] Pouya Pourbaba, K. B. Shashika Manosha, Samad Ali, and Nandana Rajatheva. “Full-Duplex UAV Relay Positioning for Vehicular Communications with Underlay V2V Links”. In: *Proc. IEEE 89th Vehicular Technology Conference*. IEEE 89th Vehicular Technology Conference. 2019, pp. 1–6.

[129] Moataz Samir, Dariush Ebrahimi, Chadi Assi, Sanaa Sharafeddine, and Ali Ghrayeb. “Leveraging UAVs for Coverage in Cell-Free Vehicular Networks: A Deep Reinforcement Learning Approach”. In: *IEEE Transactions on Mobile Computing* 20.9 (2021), pp. 2835–2847.

[130] Lijun Deng, Gang Wu, Jingwei Fu, Yizhong Zhang, and Yifu Yang. “Joint Resource Allocation and Trajectory Control for UAV Enabled Vehicular Communications”. In: *IEEE Access* PP (Sept. 2019), pp. 1–1.

[131] Moataz Samir, Chadi Assi, Sanaa Sharafeddine, Dariush Ebrahimi, and Ali Ghrayeb. “Age of Information Aware Trajectory Planning of UAVs in Intelligent Transportation Systems: A Deep Learning Approach”. In: *IEEE Transactions on Vehicular Technology* 69.11 (2020), pp. 12382–12395.

[132] Omar Oubbati, Mohammed Atiquzzaman, Abdullah Baz, Hosam Alhakami, and Jalel Ben-Othman. “Dispatch of UAVs for Urban Vehicular Networks: A Deep Reinforcement Learning Approach”. In: *IEEE Transactions on Vehicular Technology* PP (Oct. 2021).

[133] Ye Hu, Mingzhe Chen, Walid Saad, H. Vincent Poor, and Shuguang Cui. “Meta-Reinforcement Learning for Trajectory Design in Wireless UAV Networks”. In: *Proc. GLOBECOM*. Taipei, Taiwan, Dec. 2020.

[134] Ye Hu, Mingzhe Chen, Walid Saad, H. Vincent Poor, and Shuguang Cui. “Distributed Multi-Agent Meta Learning for Trajectory Design in Wireless Drone Networks”. In: *IEEE Journal on Selected Areas in Communications* 39.10 (2021), pp. 3177–3192.

[135] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wiessner. “Microscopic Traffic Simulation using SUMO”. In: *Proc. 21st International Conference on Intelligent Transportation Systems (ITSC)*. 2018, pp. 2575–2582.

[136] 3GPP. “Technical Specification Group Radio Access Network; Study on channel model for frequencies from 0.5 to 100 GHz”. In: *TR 38 901 version 16.1.0* (Dec. 2019).

[137] Vijay Kumar Salvia. *Antenna and wave propagation*. Laxmi, 2007.

[138] Volodymyr Mnih et al. “Playing Atari with Deep Reinforcement Learning”. In: *CoRR* abs/1312.5602 (2013). URL: http://arxiv.org/abs/1312.5602.

[139] *Huber Loss, PyTorch implementation*. https://pytorch.org/docs/stable/generated/torch.nn.HuberLoss. Accessed: 2022-01-28.

[140] Hado van Hasselt, Arthur Guez, and David Silver. “Deep Reinforcement Learning with Double Q-Learning”. In: *Proc. AAAI Conference on Artificial Intelligence*. Vol. 30. 1. 2016.

[141] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. “Dueling Network Architectures for Deep Reinforcement Learning”. In: ICML’16 (2016), pp. 1995–2003.

[142] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. *Continuous control with deep reinforcement learning*. 2015. URL: https://arxiv.org/abs/1509.02971.

[143] R. S. Sutton and A. G. Barto. “Reinforcement learning: An introduction”. In: *MIT Press* (2018).

[144] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. “Policy Gradient Methods for Reinforcement Learning with Function Approximation”. In: *NIPS* 12 (1999). Ed. by S. Solla, T. Leen, and K. Müller.

[145] Osvaldo Simeone. *Machine Learning for Engineers*. Cambridge University Press, 2022.

[146] Thomas Degris, Martha White, and Richard S Sutton. “Off-policy actor-critic”. In: *arXiv preprint arXiv:1205.4839* (2012).

[147] Russell Mendonca, Abhishek Gupta, Rosen Kralev, Pieter Abbeel, Sergey Levine, and Chelsea Finn. “Guided meta-policy search”. In: *Proc. NIPS*. Vancouver, Canada, Dec. 2019.

[148] 5GAA. “C-V2X Use Cases Volume II: Examples and Service Level Requirements”. In: (June 2019).

[149] Moataz Samir, Sanaa Sharafeddine, Chadi Assi, Tri Minh Nguyen, and Ali Ghrayeb. “Trajectory Planning and Resource Allocation of Multiple UAVs for Data Delivery in Vehicular Networks”. In: *IEEE Networking Letters* 1.3 (2019), pp. 107–110.

[150] Moataz Samir, Dariush Ebrahimi, Chadi Assi, Sanaa Sharafeddine, and Ali Ghrayeb. “Leveraging UAVs for Coverage in Cell-Free Vehicular Networks: A Deep Reinforcement Learning Approach”. In: *IEEE Transactions on Mobile Computing* (2020).

[151] Ahmed Al-Hilo, Moataz Samir, Chadi Assi, Sanaa Sharafeddine, and Dariush Ebrahimi. “UAV-Assisted Content Delivery in Intelligent Transportation Systems-Joint Trajectory Planning and Cache Management”. In: *IEEE Transactions on Intelligent Transportation Systems* (2020).

[152] Chunyu Pan, Jirong Yi, Changchuan Yin, Jian Yu, and Xuehua Li. “Joint 3D UAV Placement and Resource Allocation in Software-Defined Cellular Networks With Wireless Backhaul”. In: *IEEE Access* 7 (2019), pp. 104279–104293.

[153] Quoc-Viet Pham, Nadia Iradukunda, Nguyen H. Tran, Won-Joo Hwang, and Sang-Hwa Chung. “Joint Placement, Power Control, and Spectrum Allocation for UAV Wireless Backhaul Networks”. In: *IEEE Networking Letters* 3.2 (2021), pp. 56–60.

[154] Stefania Bartoletti, Barbara Mavi Masini, Vincent Martinez, Ioannis Sarris, and Alessandro Bazzi. “Impact of the Generation Interval on the Performance of Sidelink C-V2X Autonomous Mode”. In: *IEEE Access* 9 (2021).

[155] ETSI. “Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Analysis of the Collective Perception Service (CPS); Release 2”. In: *ETSI TR 103 562 V2.1.1* (2019).

[156] Ouns Bouachir, Alinoe Abrassart, Fabien Garcia, and Nicolas Larrieu. “A mobility model for UAV ad hoc network”. In: *Proc. International Conference on Unmanned Aircraft Systems (ICUAS)*. 2014.

[157] 3GPP. “NR Overall description”. In: *3GPP TR 38.300* ().

[158] Mohamed Rawidean Mohd Kassim. “IoT Applications in Smart Agriculture: Issues and Challenges”. In: *Proc. IEEE Conference on Open Systems (ICOS)*. 2020, pp. 19–24.

[159] Wan-Soo Kim, W. Lee, and Y. Kim. “A Review of the Applications of the Internet of Things (IoT) for Agricultural Automation”. In: *Journal of Biosystems Engineering* (2020).

[160] Antonis Tzounis, Nikolaos Katsoulas, Thomas Bartzanas, and Constantinos Kittas. "Internet of Things in agriculture, recent advances and future challenges". In: *Biosystems Engineering* 164 (2017), pp. 31–48. URL: https://www.sciencedirect.com/science/article/pii/S1537511017302544.

[161] Enrico Testi, Elia Favarelli, and Andrea Giorgetti. "Reinforcement Learning for Connected Autonomous Vehicle Localization via UAVs". In: *Proc. IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*. Trento, Italy, online, 2020, pp. 13–17.

[162] *Water in Agriculture*. [Online] Available: https://www.worldbank.org/en/topic/water-in-agriculture. 2020.

[163] N.R. Peplinski, F.T. Ulaby, and M.C. Dobson. "Dielectric properties of soils in the 0.3-1.3-GHz range". In: *IEEE Transactions on Geoscience and Remote Sensing* 33.3 (1995), pp. 803–807.

[164] Agnelo R. Silva and Mehmet C. Vuran. "Development of a Testbed for Wireless Underground Sensor Networks". In: *EURASIP Journal on Wireless Communications and Networking* 2010 (2010). URL: https://doi.org/10.1155/2010/620307.

[165] Xiaoqing Yu, Wenting Han, and Zenglin Zhang. "Path Loss Estimation for Wireless Underground Sensor Network in Agricultural Application". In: *Agricultural Research* 6 (2016).

[166] Yu Xiaoqing, Zhang Zeng-lin, and Han Wen-ting. "Experiment Measurements of RSSI for Wireless Underground Sensor Network in Soil". In: *IAENG International Journal of Computer Science* (2018).

[167] Fabrizio Balducci, Donato Impedovo, and Giuseppe Pirlo. "Machine Learning Applications on Agricultural Datasets for Smart Farm Enhancement". In: *Machines* 6.3 (2018). URL: https://www.mdpi.com/2075-1702/6/3/38.

[168] Shikha Prakash, Animesh Sharma, and Sitanshu Shekhar Sahu. "Soil Moisture Prediction Using Machine Learning". In: *Proc. International Conference on Inventive Communication and Computational Technologies*. 2018, pp. 1–6.

[169] Yu Cai, Wengang Zheng, Xin Zhang, Lili Zhangzhong, and Xuzhang Xue. "Research on soil moisture prediction model based on deep learning". In: *PLOS ONE* 14.4 (2019), pp. 1–19. URL: https://doi.org/10.1371/journal.pone.0214508.

[170] Amin Elshorbagy and Kamban Parasuraman. "On the relevance of using artificial neural networks for estimating soil moisture content". In: *Journal of Hydrology - J HYDROL* 362 (2008), pp. 1–18.

[171] Oliviu Matei, Teodor Rusu, Adrian Petrovan, and Gabriel Mihuţ. "A Data Mining System for Real Time Soil Moisture Prediction". In: *Procedia Engineering* 181 (2017). 10th International Conference Interdisciplinarity in Engineering, pp. 837–844. URL: https://www.sciencedirect.com/science/article/pii/S1877705817310603.

[172] Florian Liedmann and Christian Wietfeld. "SoMoS — A multidimensional radio field based soil moisture sensing system". In: *2017 IEEE SENSORS* (2017), pp. 1–3.

[173] Florian Liedmann, Christoph Holewa, and Christian Wietfeld. “The radio field as a sensor — A segmentation based soil moisture sensing approach”. In: *2018 IEEE Sensors Applications Symposium (SAS)* (2018), pp. 1–6.

[174] Azhar Hasan, Rahul Bhattacharyya, and Sanjay Sarma. “Towards pervasive soil moisture sensing using RFID tag antenna-based sensors”. In: *Proc. IEEE International Conference on RFID Technology and Applications (RFID-TA)*. 2015, pp. 165–170.

[175] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016.

[176] “Low-power Wide-Area Networks: A Comparative Analysis Between LoRaWAN and NB-IoT”. In: *CNIT Technical Report-05: Internet of Things: Technologies, Challenges and Impact* (2020).

[177] Riccardo Marini, Walter Cerroni, and Chiara Buratti. “A Novel Collision-Aware Adaptive Data Rate Algorithm for LoRaWAN Networks”. In: *IEEE Internet of Things Journal* 8.4 (2021), pp. 2670–2680.

[178] Riccardo Marini, Konstantin Mikhaylov, Gianni Pasolini, and Chiara Buratti. “LoRaWANSim: A Flexible Simulator for LoRaWAN Networks”. In: *Sensors* 21.3 (2021). URL: https://www.mdpi.com/1424-8220/21/3/695.

[179] Silvia Mignardi, Riccardo Marini, Roberto Verdone, and Chiara Buratti. “On the Performance of a UAV-Aided Wireless Network Based on NB-IoT”. In: *Drones* 5.3 (2021). URL: https://www.mdpi.com/2504-446X/5/3/94.

[180] Riccardo Marini, Enrico Testi, Chiara Buratti, Andrea Giorgetti, and Roberto Verdone. “A Sensing-without-Sensors System for Soil Moisture Estimation”. In: *2021 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*. Dec. 2021, pp. 192–196.

[181] Riccardo Marini, Konstantin Mikhaylov, Gianni Pasolini, and Chiara Buratti. “Low-Power Wide-Area Networks: Comparison of LoRaWAN and NB-IoT Performance”. In: *IEEE Internet of Things Journal* 9.21 (2022), pp. 21051–21063.

[182] Silvia Mignardi, Danila Ferretti, Riccardo Marini, Francesca Conserva, Stefania Bartoletti, Roberto Verdone, and Chiara Buratti. “Optimizing Beam Selection and Resource Allocation in UAV-Aided Vehicular Networks”. In: *2022 Joint European Conference on Networks and Communications/6G Summit (EuCNC/6G Summit)*. 2022, pp. 184–189.

[183] Riccardo Marini, Leonardo Spampinato, Silvia Mignardi, Roberto Verdone, and Chiara Buratti. “Reinforcement Learning-Based Trajectory Planning For UAV-aided Vehicular Communications”. In: 2022 30th European Signal Processing Conference (EUSIPCO). 2022, pp. 967–971.

[184] Riccardo Marini, Sangwoo Park, Osvaldo Simeone, and Chiara Buratti. “Continual Meta-Reinforcement Learning for UAV-Aided Vehicular Wireless Networks”. In: *accepted at ICC 2023* (2022).