LoRaWAN Multi-hop Uplink Extension

José Manuel da Custódia Dias

Thesis to obtain the Master of Science Degree in

**Electrical and Computer Engineering**

Supervisor(s): Prof. António Manuel Raminhos Cordeiro Grilo
Prof. Mário Serafim dos Santos Nunes

**Examination Committee**

Chairperson: Prof. Horácio Cláudio De Campos Neto
Supervisor: Prof. António Manuel Raminhos Cordeiro Grilo
Member of the Committee: Prof. Carlos Manuel Ribeiro Almeida

**November 2017**
Acknowledgments

I would like to start by thanking my Supervisor, Professor António Grilo, for the opportunity to develop my Master thesis and most importantly for his advices, guidance, patience and extremely quick feedback throughout the process. I admire his expertises in the subject and his great writing skills. His office’s door was always open to solve my doubts and the challenges that arose along the thesis’s development.

To my parents, to whom I express my profound gratitude for their unfailing support and encouragement to complete this thesis. To my brother for his critical thinking and writing skills, and to whom I reported about success and failure. To my girlfriend, Beatriz, who has patiently stood by me from the first day, even in the most stressful moments, always providing me with great incentive and with joyful moments.

To all my friends at Instituto Superior Técnico for sharing frustration, progress, knowledge and perspectives. Finally, to all my friends outside the university for bringing me relaxing and good times.
Resumo

A cobertura de uma rede LoRaWAN numa cidade é fortemente agravada pelo ambiente de propagação urbano. Os sensores são por vezes colocados em níveis abaixo do subsolo ou em locais com forte atenuação electromagnética. Para os utilizadores que têm contrato com um operador de rede instalar outro concentrador para melhorar a cobertura não é uma opção. Noutros casos, não existe conexão à Internet, e.g. caves. No presente trabalho, propõe-se e implementa-se um protocolo *multi-hop uplink* que pode interoperar com a especificação LoRaWAN como solução para estender a cobertura dos concentradores. Os nós folha transmitem pacotes de dados a nós intermédios, que os retransmitem ao longo de um caminho até um concentrador. Um caminho é determinado através de uma versão simplificada do protocolo *Destination-Sequenced Distance Vector* (DSDV). O protocolo foi implementado com sucesso e avaliado numa topologia linear e *bottleneck* através da medição do débito e da Taxa de Pacotes Recebidos (*Packet Reception Rate* (PRR)). Na topologia linear foi observado que o débito e o PRR sofreram pouca alteração com o aumento do número de saltos. O mecanismo de Carrier Activity Detection (CAD) foi incorporado como forma de evitar colisões. Colocaram-se dois emissores a transmitir aleatoriamente num intervalo que foi variado, por forma a variar a probabilidade de colisão. Observou-se que este mecanismo aumentou o número de pacotes recebidos para intervalos onde as colisões são altamente prováveis ou certas e, como tal, pode ser usado para evitar colisões entre emissores que estejam ao alcance um do outro.

**Palavras-chave:** LoRaWAN, *multi-hop*, *routing*, colisões, *uplink*, topologia
Abstract

The coverage of a LoRaWAN network in a city is greatly hampered by the harsh propagation environment. Sensors are sometimes placed under the ground or in places with strong electromagnetic attenuation. Also, for users who have a contract with a network operator, installing another gateway to improve coverage of blind spots is not an option. In other cases, there is no or very bad connectivity (e.g. basements). In the present work, we design and implement a multi-hop uplink solution integrated with the LoRaWAN specification, which can act as an extension to already deployed gateways. End nodes transmit data messages to intermediate nodes, which relay them to gateways by choosing routes based on a simplified version of Destination-Sequence Distance Vector (DSDV) routing. The routing protocol was successfully implemented and was assessed using a linear and bottleneck topology, where the Packet Reception Rate (PRR) and throughput were measured. On the linear topology we observed that the throughput and PRR did not decrease considerably with the increase of hops. We also addressed the use of Carrier Activity Detection (CAD) to detect and avoid collisions by placing two transmitters transmitting randomly within different intervals with different collision probabilities. It was found that the CAD greatly improved the PRR for intervals in which collisions are certain or highly likely and hence, it can be successfully used to avoid collisions when both transmitters are at range from each other.

Keywords: LoRaWAN, multi-hop, routing, collisions, uplink, topology
Contents

Acknowledgments ................................................................. iii
Resumo ........................................................................... v
Abstract ........................................................................ vii
List of Tables ................................................................. xiii
List of Figures ............................................................... xv
Glossary .......................................................................... xvii

1 Introduction ......................................................... 1
1.1 Motivation ................................................................. 1
1.2 Objectives ................................................................. 2
1.3 Contributed Work ...................................................... 2
1.4 Thesis outline ............................................................ 3

2 Background ........................................................ 5
2.1 Low Power Wide Area Network (LPWAN) ............. 5
  2.1.1 Sigfox ................................................................. 5
  2.1.2 Ingenu ................................................................. 6
  2.1.3 3rd Generation Partnership Project (3GPP) .......... 7
  2.1.4 Weightless .......................................................... 9
  2.1.5 Summary ............................................................ 10
2.2 LoRa ................................................................. 10
  2.2.1 LoRa™ Physical Layer ....................................... 10
  2.2.2 LoRaWAN ......................................................... 13
2.3 The Wireless Channel ............................................. 17
  2.3.1 Free Space Propagation ...................................... 18
  2.3.2 Log distance ...................................................... 18
2.4 Routing ............................................................... 19
  2.4.1 Routing Protocol for Low-power and Lossy Networks (RPL) ........................................... 20
  2.4.2 Destination-Sequenced Distance Vector (DSDV) ............................................. 21

3 LoRa Related Work .............................................. 25
3.1 The Things Network ................................................ 25
A.2.1 Packet Air Times in seconds ..................................................... A.2
A.2.2 Packet Wait Times in seconds based on 1% duty cycle .......... A.2

B  CAD RSSI Measurements  B.1
List of Tables

2.1 Comparison between several technologies for Low Power Wide Area Network (LPWAN) [4, 5]. ................................................................. 5
2.2 Relative comparison between Weightless standards [11]. ........................................ 9
2.3 Different path loss exponents $\gamma$ for different experiment environments [23]. ........ 19
2.4 Example of a possible routing table at node A in the topology of Figure 2.11. ............ 22
2.5 Example of a possible routing table message sent by node A in topology of Figure 2.11.. 22

4.1 Example of the routing table at node 3 in the linear topology $3 \rightarrow 2 \rightarrow 1 \rightarrow 0$. ....... 37
4.2 European regulations regarding the 868 Mhz frequency [1]. ..................................... 44

5.1 Packet Reception Rate (PRR) in percentage for transmissions with and without CAD for different intervals and 2000 transmitted packets. ................................. 57
5.2 PRR and throughput for the experiment with two hops and a simple forwarder. ........... 60
5.3 PRR and throughput for 4 experiments with a linear topology. ................................. 61
5.4 PRR and throughput for the experiment with a bottleneck topology composed by three LNs and one RN. ......................................................... 63
5.5 PRR and throughput for the experiment with a bottleneck topology composed by three LN and one RN using the same frequency in the links LN $\rightarrow$ RN and RN $\rightarrow$ Gateway. ......... 64
List of Figures

2.1 Typical architecture of a LPWAN. ........................................... 6
2.2 Different deployment modes for Narrow Band - Internet of Things (NB-IoT). ........ 8
2.3 Architecture of a LoRaWAN network [16]. .................................. 10
2.5 Structure of a LoRaTM packet [20]. ........................................... 12
2.6 Representation in frequency in the horizontal axis and time in vertical axis of LoRaTM signals [21]. .............................................................. 12
2.7 Structure of a LoRaWAN packet containing application data with lengths in bytes. .... 14
2.8 Structure of a LoRaWAN beacon packet with lengths in bytes. ......................... 14
2.9 Packet structure of the packets exchanged on a join procedure using OTTA [22] with lengths in bytes. ...................................................... 16
2.10 An example of a DAG and a DODAG. ........................................ 20
2.11 Linear topology example. ....................................................... 22

3.1 Example of the metadata of a packet as observed in the TTN dashboard. ............... 26
3.2 Geographical distribution of TTN gateways. Blue colored marks represent deployed gateways. Green colored marks represent planned gateways. ....................... 26
3.3 The Things Network architecture. ............................................ 27
3.4 Visual representation of the experiment in [2]. ...................................... 31

4.1 Timing diagram of the implemented multi-hop solution. Different colors represent different types of nodes. .................................................. 36
4.2 Packet structure of a full dump containing routing information with field lengths in bytes. 37
4.3 Structure of a application data packet within the multi-hop network with field lengths in bytes. ......................................................... 38
4.4 Top level flowchart of the program running on RNs. .................................. 39
4.5 Flowchart of the interrupts. ..................................................... 40
4.6 Medium level flowchart of a reception. ......................................... 41
4.7 Flowchart of the construction of a full dump packet. ............................... 43
4.8 Flowchart of the transmission of a packet with Collision Avoidance using Carrier Activity Detection. .................................................. 44
4.9 Wiring diagram between the HopeRF RFM96 LoRa\textsuperscript{TM} 868/915Mhz RF Transceiver's breakout board and the MCU Nucleo-STML432KC. The circle in red represents the solder bridge 1 (SB1). 48

4.10 Different views of a LoRaWAN gateway comprising a Raspberry Pi (on the bottom), an interface board (in the middle), a iC880A radio module (on top) and a GPS module. 48

5.1 Graphic of the variation of packet airtime with different lengths with the spreading factor. 52

5.2 Graphic of the variation of time between packets with different lengths with the spreading factor. 53

5.3 Map generated by TTN Mapper with the measurement location points. 54

5.4 Variation of the Received Signal Strength Indicator (RSSI) with the distance in the city of Almada, Portugal. 55

5.5 Variation of the PRR with the normalized elapsed time for a transmit interval of 128 ms, in blue, and for an interval of 8 ms, in red. 56

5.6 PRR for the different transmit time intervals tested. 57

5.7 Time interval between the start of the Beacon reception window and the start of the actual reception. 58

5.8 Diagram representing the experiment with a simple forwarder between the LN and the gateway and no used routing overhead. 60

5.9 Linear topology test configuration. In each test the number of RNs was increased by 1. 61

5.10 Bottleneck topology with one RN and 5 LNs. 62

5.11 RSSI measurements at the RN for the test in a bottleneck topology with three LNs and one RN using the same frequency in the links LN → RN and RN → Gateway. 64

B.1 Distribution of RSSI for a random transmission interval between transmissions with an upper bound of 8 and 16 ms. The presence of two clusters of RSSI values corresponds to the weak and strong transmissions. B.1

B.2 Distribution of RSSI for a random transmission interval between transmissions with an upper bound of 32 and 64 ms. The presence of two clusters of RSSI values corresponds to the weak and strong transmissions. B.2

B.3 Distribution of RSSI for a random transmission interval between transmissions with an upper bound of 128 ms. The presence of two clusters of RSSI values corresponds to the weak and strong transmissions. B.2
Acronyms

3GPP 3rd Generation Partnership Project. 7
6LoWPAN IPv6 over Low Power Wireless Person Area Networks. 21
ABP Activation by Personalization. 38, 45
AES Advanced Encryption Standard. 9
API Application Programming Interface. 47
BPSF Binary Phase Shift Keying. 6
BW Bandwidth. 46
CAD Carrier Activity Detection. x, xiii, 2, 43, 44, 46, 55–58, 63, 64, 66
CDMA Code Division Multiple Access. 6, 7
CR Coding Rate. 11, 12, 46
CRC Cyclic Redundancy Check. 11, 12, 39–41
CSS Chirp Spread Spectrum. 10, 11
DAG Directed Acyclic Graph. 20
DAO DODAG Advertisement Object. 20
DBPSK Differential Binary Phase Shift Keying. 9
DIO DODAG Information Object. 20
DIS DODAG Information Solicitation. 20
DL Downlink. 13
DODAG Destination Oriented Directed Acyclic Graph. 20
DSDV Destination-Sequence. 19–21, 23, 35
EC Extended Coverage. 8–10
eDRX  LTE Extended Discontinuous Reception. 8, 9
EGPRS  Enhanced General Packet Radio Service. 8, 9
eMTC  Long Term Evolution enhancements for Machine Type Communications. 8–10
FEC  Forward Error Check. 12
FSK  Frequency Shift Keying. 13
GMSK  Gaussian Minimum Shift Keying. 9
GPRS  General Packet Radio Service. 9
GSM  Global System for Mobile Communications. 8, 10
IETF  Internet Engineering Task Force. 20
IoT  Internet of Things. 1, 5, 7, 8, 47
IP  Internet Protocol. 6, 7
ISM  Industrial Scientific Medical. 5, 6, 9, 10
LAN  Local Area Network. 10
LLN  Low-power and Lossy Network. 20
LPWAN  Low Power Wide Area Network. xiii, xv, 5, 6, 9, 10
LTE  Long Term Evolution. 7–10
MAC  Medium Access Control. 32
MCU  Microcontroller. 46
NB-IoT  Narrow Band - Internet of Things. xv, 8, 10
OOK  On-Off Keying. 11
OTAA  Over-the-Air Activation. 17, 45
P2P  Peer to Peer. 10
PN  Pseudo Noise. 7, 11
PRB  Physical Resource Block. 8
PRR  Packet Reception Rate. xiii, xvi, 2, 56, 57, 59–66
PSM  Power Saving Mode. 8, 9
QAM Quadrature Amplitude Modulation. 9
QoS Quality of Service. 1, 19
QPSK Quadrature Phase Shift Keying. 9
RF Radio Frequency. 5, 10
RPL Routing Protocol for Low power and Lossy Networks. 19–22, 45
RPMA Random Phase Multiple Access. 6, 7, 10
RSSI Received Signal Strength Indicator. xvi, 18, 27–30, 53–55, 57, 58, 64–66, B.1, B.2
SDK Software Development Kit. 47
SF Spreading Factor. 6, 7, 11, 12, 46
SIG Special Interest Group. 9
TDD Time Division Duplex. 9
UE User Equipment. 7, 8
UL Uplink. 13
UMTS Universal Mobile Telecommunication System. 7
UNB Ultra Narrow Band. 6, 9
Chapter 1

Introduction

Each year millions of new users and devices connect to the Internet. The number of online devices had already passed the human population by 2012 and continues to grow in billions every year. What initially started as a way of individuals interacting within a global network to exchange data, evolved to connect barely any electronic device.

In the Internet of Things (IoT), devices collect data, transmit it usually using radio links to base stations, which relay it to an Application Server in the Internet. Data is stored in the Cloud and processed using machine learning and data mining techniques to retrieve useful information. This information may be used in several domain specific applications, which include smart home, education, market, industry, transportation, health and agriculture. IoT devices collect their data through one or more connected sensors and are typically powered by a battery, which results in sending sample values periodically under a power constraint. This represents a much lower data rate requirement in comparison, for instance, with streaming video or voice applications. Several compromises had to be made, leading to device constraints: complexity reduction in the end-nodes in order to reduce cost, reduce available data rate to increase range and to develop methods to increase efficiency in power consumption. There are several enabling technologies for IoT, which differ in power consumption, range, throughput, Quality of Service (QoS), scalability and cost. LoRa is one such technology operating in license-free sub-Ghz spectrum. The sub-Ghz spectrum has good propagation properties and longer range due to longer wavelength carriers than IEEE 802.11 or Bluetooth, but it is limited in European Union to a duty cycle of 1% in 868-868.6 MHz and 0.1% in 868.7-869.2 MHz [1]. As all devices are under the same restrictions, gateways need to be carefully designed to make an efficient use of the spectrum.

1.1 Motivation

Gateways are complex and costly to maintain. They need to simultaneously listen in all channels and are permanently connected to the Internet. Its density can be reduced through the use of simple relay nodes in a multi-hop scheme. Cities may have dark spots, with weak or no coverage at all, in these

---

scenarios a multi-hop solution is a simple, effective and cheap solution to improve coverage. Also, when a network user does not own the network infrastructure, deploying additional gateways is not an option. Moreover, multi-hop allows data to be collected from remote locations, such as villages or rural areas with no accessible gateways and where it is not economically viable to deploy and maintain a gateway. In addition, if a gateway fails, a multi-hop protocol could mitigate the network failure, as the messages would be forwarded by relay nodes to the nearest available gateway.

Aspects of LoRa such as capacity, scalability, propagation and multi-hop have been explored in the most recent literature. Bor et al. [2] proposed and tested a multi-hop protocol between LoRa nodes (analyzed in Chapter 3) in a challenging environment while assuming low traffic and low density of nodes. The previous multi-hop solution did not take into account the LoRaWAN specification and does not prevent duplicate packets. Adelantado et al. [3] suggests a multi-hop approach in future research which would allow a decrease in transmission power and Spreading Factor (SF).

1.2 Objectives

The main objective of the present work is to design, implement, test and assess a routing protocol between LoRa transceivers. The routing protocol is intended to interoperate with LoRaWAN gateways, make use of the LoRaWAN beacon and deliver LoRaWAN packets from end nodes to gateways through the minimal number of hops. A prototype shall be developed in order to evaluate the proposed mechanisms under relevant scenarios. The objectives are enumerated below.

- Design of a multi-hop routing protocol compatible with the LoRaWAN specification for upstream traffic.
- Development of a prototype comprising an architecture of a typical LoRaWAN architecture with the implemented aforementioned routing protocol.
- Experimental characterization of LoRaWAN propagation conditions in a urban environment comprising different altitudes.
- Performance evaluation of the prototype under a bottleneck and linear topology by measuring the PRR and throughput.
- Evaluate the performance of CAD in collision avoidance.

1.3 Contributed Work

The uplink communication protocol was developed to interoperate with LoRaWAN gateways and be implemented on microcontrollers which are connected to a LoRa transceiver. The great majority of the described flowcharts are sufficiently abstract to allow its implementation in any platform with the described structures. Also, other LPWAN technologies interested in using the multi-hop concept may benefit from what is presented here. Furthermore, from the routing point of view, the protocol can support
any intermediate node as a destination, which allows traffic to flow in any direction. The protocol was evaluated using two topologies and its results provide information regarding two essential parameters of the transmission of information: rate of data reception and data loss.

1.4 Thesis outline

This document is organized as follows:

Chapter 2 (Background) provides a description of LPWAN technologies, ad hoc routing protocols and other useful concepts.

Chapter 3 (LoRa Related Work) describes an emerging crowd-sourced LoRaWAN network and an analysis to the most recent LoRa literature in various fields: scalability studies, collisions, empirical measurements and routing.

Chapter 4 (LoRaWAN Multi-hop Uplink Extension) describes the implemented communication protocols and mechanisms, and the prototype implementation.

Chapter 5 (Experimental Results) contains an assessment of the proposed solution and its integration with a collision avoidance mechanism.

Chapter 6 (Conclusions) contains the conclusions of the thesis and ideas for future work.
Chapter 2

Background

This chapter provides a description of what is a LPWAN and its components, providing examples of technologies that, as LoRa, are competing in the market of IoT. Further on, LoRa is analyzed, both the physical layer and the LPWAN specification for LoRa - LoRaWAN. Next, the wireless channel is briefly covered, with the phenomena that a signal may suffer in the propagation environment and the log distance empirical model. Finally, two distance vector routing protocols, which are elected as the most appropriate for a LoRa network out of a large set of routing protocols, are described.

2.1 Low Power Wide Area Network (LPWAN)

A LPWAN is characterized by long range, low power consumption, low cost and low throughput. End-nodes can be as far as dozens of kilometers, communicate up to dozens of kbits per second and with a battery lasting up to 10 years. In a typical architecture, as represented in Figure 2.1, end-nodes are usually battery powered devices with Radio Frequency (RF) modules which are placed in moving or static objects. Gateways are deployed several kilometers apart depending on range, node density and physical propagation environment. End-nodes communicate using their RF modules to a gateway, which relays the information to an Application Server. A summary of the different technologies and their characteristics is illustrated in Table 2.1.

<table>
<thead>
<tr>
<th>Technologies</th>
<th>LoRa</th>
<th>SIGFOX</th>
<th>Weightless-N</th>
<th>Ingenu</th>
<th>LTE-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>Sub-GHz ISM</td>
<td>Sub-GHz ISM</td>
<td>Sub-GHz ISM</td>
<td>2.4 GHz</td>
<td>LTE band</td>
</tr>
<tr>
<td>RF PHY</td>
<td>CSS and FSK</td>
<td>UNB</td>
<td>UNB</td>
<td>DSSS</td>
<td>LTE</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>0.3 - 50 kbit/s</td>
<td>100 bits/s (EU)</td>
<td>100 bits/s</td>
<td>100 kB/day</td>
<td>1 Mbit/s</td>
</tr>
</tbody>
</table>

Table 2.1: Comparison between several technologies for LPWAN [4, 5].

2.1.1 Sigfox

Sigfox offers end-to-end LPWAN connectivity through a cellular style system in the Sub-GHz Industrial Scientific Medical (ISM) spectrum using proprietary technology. An end-device transmits to a base
station using a Binary Phase Shift Keying (BPSK) modulation in an Ultra Narrow Band (UNB) subcarrier (100Hz). The base station then forwards the message to the backend servers through an Internet Protocol (IP)-based network. By concentrating the signal’s energy in a very narrow band, Sigfox achieves low noise levels, high receiver sensitivity and high capacity of the network, at the cost of a maximum throughput of 100 bits/s. A maximum of 140 messages with a payload of 12 bytes are permitted on uplink and 8 messages with a payload of 8 bytes on downlink per device per day, in order to comply with the legal limitations of the license-free spectrum. Obviously, it is impossible to acknowledge every uplink message and without additional mechanisms the radio link would be unreliable. To improve the reliability, an end-device transmits the same message multiple times, typically three, over different frequency channels, adding redundancy in frequency and time, therefore increasing the probability of a base station receiving the message. In Europe there are 400 channels 100Hz wide between 868.180 and 868.220 MHz. An end-device chooses a random frequency channel in the previous interval and transmits, while on the network side, a base station scans all the available channels simultaneously for incoming messages.

### 2.1.2 Ingenu

Ingenu, former On-Ramp Wireless, operates at 2.4 Ghz ISM band with a patented technology for the physical medium access named Random Phase Multiple Access (RPMA). RPMA is a variant of Code Division Multiple Access (CDMA). CDMA spreads a signal over a bandwidth that is much broader than the one required to transmit that signal without spreading. Spreading is accomplished with a spreading code sequence, unique for each transmitter sharing the medium, which is XORed with each bit of the data signal. The result is usually referred as the chip sequence that is transmitted with at a given chip rate. At the receiver, the incoming signal is correlated with the locally generated code sequence, which matches that used by the transmitter. Code sequences may vary in length according to the SF ratio...
defined as in equation 2.1.

\[ SF = \frac{\text{Chip rate}}{\text{Data rate}}. \]  

(2.1)

SF indicates the length of the code sequence per data bit. Logically, it derives that a greater SF increases the transmitted signal’s resilience to interference and errors but inversely decreases the data rate. Codes must be carefully chosen to minimize mutual interference between signals from different transmitters. In synchronous CDMA, true orthogonal codes are used whereas asynchronous CDMA uses quasi-orthogonal Pseudo Noise (PN) codes for each transmitter. PN sequences aim to perform with an auto correlation with statistical properties of sampled white noise. That is, taken a sample and correlating it to itself at different points in time, the distribution is Gaussian with a mean equal to the DC component. Despite of appearing random in the channel, PN codes are deterministic with known period by both sender and receiver [6]. CDMA significantly mitigates the effects of broadband and narrowband interference and, if code sequences are well chosen, it can also mitigate inter-symbol interference.

RPMA uses PN sequences described previously in CDMA with the same purpose of spreading the signal but without unique PN codes per transmitter. Instead, a single PN sequence is used for all transmitters and, in order to distinguish transmissions, a phase is selected randomly. This phase selection sets the time offset, i.e. the time a sender must wait before transmitting in that time slot. Time slots are longer than in CDMA to compensate for the maximum possible time offset [7, 8]. Despite the large set of offsets, two or more transmitters may be given the same random offset and originate a collision, in which case it may not be possible to demodulate the two or more signals and a retransmission is attempted with another randomized offset [7]. For downlink, a base station uses CDMA to broadcast, assigning unique PN sequences per each end-device to isolate channels [7, 8].

2.1.3 3rd Generation Partnership Project (3GPP)

The 3rd Generation Partnership Project (3GPP) was initially created as a strategic association between groups of telecommunications to develop the specification for the next generation of mobile communications after the 3GPP. The standard should be globally applicable and support IP for wireless communications. The collaboration continued with the development of Universal Mobile Telecommunication System (UMTS) and most recently Long Term Evolution (LTE). With the growth of IoT and M2M market there was a need to address this market by taking advantage of the already deployed cellular network and infrastructure. Three standards were developed regarding the needs of IoT.

2.1.3.1 LTE Enhancements for Machine Type Communication

As being an upgrade in the mobile communications, LTE increased the capacity and simplified the network. It offers devices characteristics not suited for IoT context: high data rate, low latency in exchange of cost and power consumption. 3GPP defines various categories that establish maximum UL and DL data rates for different User Equipment (UE). Category 0 was introduced in release 12, reducing the peak data rate to ten percent of the first category and giving an option for half duplex. Furthermore, category M1, or Long Term Evolution enhancements for Machine Type Communications (eMTC), was
introduced in the next release with even lower limits in bandwidth and data rate and, more importantly, two additional modes of operation for UEs. During normal LTE operation, UEs can be contacted by the base stations every 1.28 seconds which is called the paging cycle. The UE has to periodically wake up and check if there is queued traffic directed to it. If there is, the UE requests it, if not UE enters sleep mode again. This obviously consumes power unnecessarily for cases where DL traffic is sporadic. LTE Extended Discontinuous Reception (eDRX) addresses this by increasing the paging period to 10.24s and allowing the UE to announce to the base station the number of periods it will sleep until the next paging check. Moreover, the Power Saving Mode (PSM) allows the UE to announce that it will enter sleep mode indefinitely. When it needs to transmit, it wakes up, transmits and remains idle right after in order to be reachable by DL traffic.

2.1.3.2 NarrowBand-IoT

NB-IoT standardization was completed in June 2016. It is a new air interface also integrated in LTE standard and, as eMTC, is described in release 13. LTE air interface uses Physical Resource Block (PRB) as the smallest chunk of transmitted data, each constituted by 12 subcarriers for a single time slot. There are three different deployment modes which differ on the spectrum region used to transmit - Stand Alone, In Band and Guard Band. A graphical representation of the three is provided in Figure 2.2.

![Figure 2.2: Different deployment modes for NB-IoT.](image)

Stand-alone (Figure 2.2a) deployment utilizes a new bandwidth outside the bands used for LTE. In Band (Figure 2.2b), as the name implies, transmits in one PRB reserved for NB-IoT inside the LTE’s spectrum, which allows for an improve in efficiency of frequency use and capacity of serving more UEs. Guard-band mode (Figure 2.2c) operates within the guard band of an LTE carrier. A single NB-IoT transmission can transmit in 1, 3, 6 or all the 12 PRB subcarriers. When end devices do not need maximum data rates or the number of connected devices to a base station is high, less bandwidth is allocated to each device which allows to multiplex them in frequency. As each LTE subcarrier is 15 kHz wide, using all the 12 subcarriers results in a maximum bandwidth of 180 kHz per device. [9].

2.1.3.3 Extended Coverage GSM

Extended Coverage (EC)-Global System for Mobile Communications (GSM), formerly EC-Enhanced General Packet Radio Service (EGPRS), is an evolution of EGPRS oriented for the IoT. It features low complexity and low power for end nodes and an extended coverage by 20 dB due to increase in sensi-
tivity. An end node can use the General Packet Radio Service (GPRS)/EGPRS network in cells where EC operation is not available, taking advantage of higher data rates but less range and more power consumption. Energy efficient operation is achieved through relaxed mobility related requirements and optionally, similarly to LTE-eMTC, using eDRX and PSM. Paging monitoring is optimized and security framework is improved by both the network and the end node [10].

2.1.4 Weightless

The Weightless Special Interest Group (SIG), a non-profit global standards organization, developed three open LPWAN technologies both on licensed and unlicensed bands: N, P and W. Table 2.2 contains basic characteristics of the three and may give hints on the applications permitted by each one.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Weightless-N</th>
<th>Weightless-P</th>
<th>Weightless-W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directionality</td>
<td>1-way</td>
<td>2-way</td>
<td>2-way</td>
</tr>
<tr>
<td>Feature Set</td>
<td>Simple</td>
<td>Full</td>
<td>Extensive</td>
</tr>
<tr>
<td>Range</td>
<td>+5 km</td>
<td>+2 km</td>
<td>+5 km</td>
</tr>
<tr>
<td>Battery life</td>
<td>10 years</td>
<td>3-8 years</td>
<td>3-5 years</td>
</tr>
<tr>
<td>Terminal cost</td>
<td>Very low</td>
<td>Low</td>
<td>Low-medium</td>
</tr>
<tr>
<td>Network cost</td>
<td>Very low</td>
<td>Medium</td>
<td>Medium</td>
</tr>
</tbody>
</table>

Table 2.2: Relative comparison between Weightless standards [11].

Weightless-N utilizes a UNB technology on the sub-GHz ISM spectrum. It modulates the signal with a Differential Binary Phase Shift Keying (DBPSK). To reduce interference, frequency hopping is used. Encryption is provided via 128 bit Advanced Encryption Standard (AES) algorithm and mobility is supported.

Weightless-W uses unused portions of the spectrum allocated for analog TV - white spaces. In the regions where it is possible, Weightless-W transmits on these white spaces, where base stations query a database for used channels to select the free ones to transmit. The frequency range is between 470 MHz and 790 MHz. As the rules and regulations vary geographically this may be a constraint for Weightless-W to scale.

It uses several types of modulation, from DBPSK to 16-Quadrature Amplitude Modulation (QAM) and a set of 1024 spreading codes allow trading-off between range and data rate to match the actual needs. As the spectrum is not guaranteed, uplink and downlink share the same frequency and transmit using Time Division Duplex (TDD).

Weightless-P is an improvement in reliability for the Weightless-N, allowing two-way communications and therefore acknowledgments. It uses Gaussian Minimum Shift Keying (GMSK) and off-set-Quadrature Phase Shift Keying (QPSK) modulation.
2.1.5 Summary

In the present section, LPWAN technologies with highest presence in the market and literature were covered. The following observations provide the link with the present work as well as hints on why LoRa was chosen. NB-IoT, EC-GSM and LTE-eMTC have still not been fully launched. Nevertheless, NB-IoT, for instance, already has a module available for pre order (SARA-N2) [12]. Regarding Sigfox, Waspnute [13] is a Sigfox module that offers Peer to Peer (P2P) communication, which allows a multi-hop scheme. However, the protocol used in the P2P communication does not employ the Sigfox modulation and is intended to form a Local Area Network (LAN) behind an end-node instead of extending the LPWAN. This end-node acts as a gateway between the LAN and nearby base stations.

Nano-S100 is a RF chip employing RPMA, which consumes more power than LoRa or Sigfox [5, 14, 15] mainly by operating at 2.4Ghz ISM (shorter wavelength). This region of the spectrum is limited in transmitted power but not in duty cycle.

2.2 LoRa

LoRa™ is the short for Long Range. It is a long-range wireless communication technology intended for battery powered applications where the battery lifetime and energy consumption is important. The term LoRa can refer to the physical layer LoRa™ proprietary technology held by Semtech Corporation, or to the MAC layer protocol LoRaWAN promoted by the LoRa™ Alliance. Figure 2.3 shows the architecture of a typical LoRaWAN system.

![Figure 2.3: Architecture of a LoRaWAN network [16].](image)

2.2.1 LoRa™ Physical Layer

The LoRa™ physical layer implements a derivative of Chirp Spread Spectrum (CSS) scheme. CSS was first developed in 1951 at Bell Telephone Laboratories as a classified work for the military radars. The work aimed to offer the same efficiency in range, resolution and speed of acquisition but without the high peak power of the traditional short pulse mechanism [17].
A chirp is a signal in which its frequency increases (up-chirp) or decreases (down-chirp) over time as observed in the sinusoidal linear chirp signal represented in Figure 2.4.

![Linear chirp signal](image)

Figure 2.4: Representation of a simple sinusoidal linear up-chirp signal over time.

In CSS, each data bit is modulated with a single or a sequences of these chirps, e.g. On-Off Keying (OOK) - assigning an up-chirp to a logical 1 and no frequency variation to a logical 0 [18]. In the case of LoRa™, it employs its own LoRa™ modulation. This constitutes an alternative to the spreading method described in subsection 2.1.2 without using PN sequences. Instead, transmissions are distinguished by the time duration of each chirp reflected on the SF, i.e. higher SFs correspond to longer chirps and lower SFs to shorter chirps. Similarly to subsection 2.1.2, higher SFs increase resistance to natural interference, noise and jamming but decreases available bandwidth for data. According to [19], the definition of the SF for LoRa™ is different from equation 2.1 of subsection 2.1.2, being

$$SF = \log_2 \left( \frac{\text{Chirp rate}}{\text{Symbol rate}} \right)$$

which also means that every symbol is encoded in $2^{SF}$ chirps that cover the available bandwidth ($BW$) (one chirp per second per Hz). The duration of a symbol transmission, $T_S$, can therefore be expressed as in Equation 2.2.

$$T_S = \frac{2^{SF}}{BW}$$

(2.2)

Knowing that a symbol rate, $R_S$ is the inverse of $T_S$, that symbol rate relates with the chip rate by $R_C = 2^{SF} \times R_S$ and that each symbol encodes $SF$ bits, the expression for the useful bit rate comes:

$$R_B = SF \times \frac{1}{2^{SF}BW}.$$  (2.3)

To improve robustness to errors, LoRa™ adds redundancy by employing a Cyclic Redundancy Check (CRC) to perform forward error detection and correction. The measure of redundancy is the Coding Rate (CR) defined as the portion of the bit stream which corresponds to effective data, i.e. non-redundant. According to [19], the Rate Code is defined as

$$\text{Rate Code} = \frac{4}{4 + CR}.$$  (2.4)

By applying the additional factor of Equation 2.4 to Equation 2.3, the data rate with all the factors into consideration is given in Equation 2.5. Since in a LoRa™ one can modify the $BW$, $CR$ and $SF$, this...
expression comes at hand to calculate the available data rate to the above layers such as the application layer.

\[ R_B = SF \times \frac{4 \times CR}{BW} \] (2.5)

The structure of a Lora packet, which can be explicit or implicit, is showed on Figure 2.5. It comprises a configurable preamble for synchronization of the receiver, which is a sequence of symbols (chirps) ranging from 6 to 65535, an optional header that can provide information about the payload length, the Forward Error Check (FEC) CR and if CRC is present. Implicit mode exists to reduce overhead by eliminating the header field. This is useful in situations where both sides have prior information about the CR and the CRC or when these parameters are fixed and manually configured at both sides. Figure 2.5 allows one to notice where the previously mentioned SF and CR affect each part of the packet.

Figure 2.5: Structure of a LoRa™ packet [20].

Figure 2.6a represents the preamble of a LoRa™ modulated signal as a sequence of symbols as chirps and 2.6b represents the following data symbols. Further information is provided about the configurable parameters and range of possible values in Chapter 5.

Figure 2.6: Representation in frequency in the horizontal axis and time in vertical axis of LoRa™ signals [21].
2.2.2 LoRaWAN

LoRaWAN is a network protocol specified from the data link layer above in the OSI\(^1\) model, comprising the following layers: network, transport, session, presentation and application. Typically, a LoRaWAN network consists of a single hop star topology, in which gateways in the center relay messages between the end-devices and a Network Server. An end-device can reach one or more gateways using LoRa\({\text{TM}}\) or Frequency Shift Keying (FSK) modulation through a frequency channel. Gateways are connected to the Internet via Wifi, 3G or Ethernet and a message is delivered to the Network Server through IP [22]. Different end-devices and gateways can coexist in the same geographic area by differentiating communications in frequency and also in virtual channels within the same frequency channel, as, due to spread spectrum, transmissions with different spreading factors are orthogonal. Each gateway is constantly listening on simultaneous channels and can successfully receive multiple packets as long as they are transmitted in different channels. To address applications with different needs of power consumption, LoRaWAN specifies three classes of end-devices based on their listening time, all of which with bidirectional communication [22]:

- **Class A** - An end-device only listens for Downlink (DL) traffic during two scheduled time intervals after an Uplink (UL) transmission takes place. The scheduling is performed by the end-device and is accomplished in a random fashion. This type of end-device is oriented for applications which mainly rely on UL transmissions and with the most severe power constraints. A DL message originated on the Network Server to a specific end-device will have to be queued on the gateway and wait for the receive windows that follow the next UL transmission of the destination end-device.

- **Class B** - An end-device will listen at scheduled times. A beacon is periodically sent by the gateway for synchronization. This allows an end-device to be reachable at certain intervals and removes the dependency of an UL transmission as in Class A.

- **Class C** - An end-device listens whenever it is not transmitting. This class offers the lowest latency for DL traffic at the cost of more power consumption.

There are certain rules that an end-node needs to respect:

- The choice of a frequency channel must be pseudo-random for each transmission.
- Duty cycle per sub-band and per device must be respected according to the local regulations.

2.2.2.1 Beacon

The beacon is sent by gateways in a periodic basis with \( SF = 9, \quad BW = 125 \text{ kHZ} \) and \( f = 869525 \text{ MHZ} \) for class B devices. This frequency channel, in Europe, allows for a duty cycle of 10% and +27 dBm EIRP\(^2\). Although a network operator may use other settings, these are the standard and recommended. The beacon’s packet structure is represent in Figure 2.8.

---

\(^1\) Open Systems Interconnection  
\(^2\) Equivalent Isotropically Radiated Power
The NetID field is the network identifier, the Time, Latitude and Longitude are read from the GPS module connected to the gateway and the InfoDesc is typically 0 which is the case when the gateway is connected to a single omnidirectional antenna. Otherwise, if the gateway has 3 sectored antennas, each antenna will transmit the beacon with a different InfoDesc field with its own coordinates.

2.2.2.2 Adaptive Data Rate Scheme (ADR)

The LoRa network infrastructure may have control over end-nodes's RF power and spreading factor through an Adaptive Data Rate Scheme (ADR), which directly affects the data rate. Recall that, as seen in section 2.2.1, the data rate $R_B$ is proportional to the spreading factor as represent in Equation 2.6.

$$R_B \propto \frac{SF}{2^{SF}} \quad \text{(2.6)}$$

Hence, when referring to a change in data rate in this subsection, the spreading factor is implicitly changed as well.

The ADR allows the network infrastructure to better distribute the data rate and RF power according to each node’s specific needs. In particular, it tries to assign the lowest possible spreading factor to each node, resulting in the lowest time on air of each transmission and thus, fastest data rate available. The node must verify that the gateway is receiving uplink packets. If not, the spreading factor in use is not appropriate for the quality of the radio link and thus needs to be increased to allow a longer range. For every uplink packet transmitted which is not a retransmission, two counters are incremented: a general uplink frame counter and one for data rate adjustment. The latter resets for every downlink traffic.
received. If a node loses connectivity with a gateway, the second counter will not reset and will eventually increase up to a certain limit, for which the node assumes the connectivity has been lost. Next, the node may try to attempt to recover connectivity by configuring the radio with a higher spreading factor. This increases the radio range and the probability of the next transmitted packet being received. The ADR feature can be controlled through a bit present in the Frame Control byte of Figure 2.7. This scheme is obviously not indicated for scenarios where electromagnetic attenuation changes fast and constantly, e.g. mobile nodes. In this case ADR should be deactivated and the application should handle the data rate configuration.

2.2.2.3 Security

All LoRaWAN messages are encrypted end-to-end using the Advanced Encryption Standard (AES). AES is a symmetric key block cipher which encrypts each message by encrypting 128 bits at a time. If the total length of the message is not divisible by the block length, then padding is used as complement. Each block is represented by a 4x4 matrix in a total of 16 bytes, which enters a certain amount of encryption rounds, each containing a sequence of substitutions, permutations and a XOR bitwise operation with a subkey - a variation of the key. The key is expanded in multiple subkeys that are used to perform the XOR with the state, i.e. the result of the substitutions and permutations of each particular round. The number of rounds depends on the key length. In the case of LoRaWAN, for 128 bit keys, the key is expanded in 176 bytes. Knowing that each round key is 16 bytes, this results in 11 rounds, with the first containing only the XOR operation.

If a node sends an application message, the frame payload field in Figure 2.7 is encrypted with an Application Key and only decrypted in the Application Server. On the other hand, if it is a network control message, the Network Session Key is used and the message is decrypted in the Network Server. The decision of which key to use is specified in the Frame Port, where the value 0 stands for the Network Session Key and any other value less than 256 for the application key. Using both keys allows roaming of nodes between different network operators without the risk of the application data being tampered.

The use of encryption in a LoRaWAN system ensures confidentiality by preventing eavesdropping in the radio link between the end node and the gateway and on the Internet between the gateway and the Application Server. The authenticity and integrity of a message are provided by appending a Message Integrity Code (MIC) at the MAC layer. The MIC in LoRaWAN results from encrypting (AES-CMAC\(^3\)) the frame payload plus the MAC header, Frame Header and Frame Port in Figure 2.7 with the Network Session key. This confirms that the message came from the stated source and that its content has not been changed. Replay attacks, where an attacker retransmits a previous message, are prevented by the usage of the already mentioned frame counters, both uplink and downlink. The node and the Application Server must discard packets with a frame counter lower than the expected count. The security robustness of LoRaWAN is ensured by combining AES, MIC and frame counters and therefore turning it difficult for an attacker to be successful in time efficient attacks.

\(^3\)https://tools.ietf.org/html/rfc4493
2.2.2.4 Over-the-Air Activation (OTAA)

The Over-the-Air Activation (OTAA) is a procedure that allows a node to join the network by identifying itself in order to be given the keys to encrypt data before starting the application data exchange. There is a new join procedure every time the session context information is lost. The Application Server has a root key, only known by the application provider, that generates an Application Key. This Application Key is configured on the node and is used to generate the Network Session key and the Application Session key for each join procedure.

![Packet structure of a join request packet.](image1)

![Packet structure of a join accept packet.](image2)

Figure 2.9: Packet structure of the packets exchanged on a join procedure using OTTA [22] with lengths in bytes.

The procedure starts with the node sending a join request with the information of Figure 2.9a. The Application Identifier (AppEUI) and the Device Identifier (DevEUI) are unique identifiers that must be configured in the node along with the Application Key. The Nonce is a random number generated by the node that prevents replay attacks since the Network Server keeps track of the previous nonces for each node. The join request message is not encrypted but includes a MIC computed with the Application Key over the MAC header plus the fields of Figure 2.9a. If the node is accepted to join the network, a join-accept message is sent downlink. This message is sent with a MIC as well as an Application Nonce. The Application Key is used to encrypt the payload in Figure 2.9b plus the MAC header. The remaining fields are described as follows:

- **Network ID (NetID)** - identifies the network.
- **Device Address** - identifier of the node within the network.
- **Downlink Settings (DLSettings)** - indicates the spreading factor used in the two receiving windows for downlink traffic.
- **RxDelay** - indicates the delay between the end of a device’s transmission and the start of the first receiving window, from 1s to 15s. The second receive window is 1 s after the end of the first receiving window.
- **Channel Frequency List (CFList)** - lists the frequency channels in use by that network (optional).

The Application Nonce is used to derive both the Network Session Key and the Application Session Key. This is accomplished by using the AES algorithm as show in 2.7 and 2.8.

\[
NwkSKey = E_{AES_{128}}(AppKey, 0x01 | AppNonce | NetID | DevNonce | pad_{16}) \tag{2.7}
\]
\[ AppSKey = E_{AES_{128}}(\text{AppKey}, 0x02 | \text{AppNonce} | \text{NetID} | \text{DevNonce} | \text{pad}_{16}) \] (2.8)

The padding \((\text{pad}_{16})\) appends zero octets and, as in subsection 2.2.2.3, is used when the total length of the request message is not divisible by 16. The join accept message, contrary to the join request, is encrypted with AES with the Application Key since it contains valuable information used by the node to derive the two session keys. However, the encryption is computed with AES decrypt and decrypted on the node with AES encrypt, which greatly reduces complexity in the node by limiting the implementation of AES to encrypt only.

### 2.2.2.5 Activation by Personalization (ABP)

The Activation by Personalization (ABP) procedure skips the join procedure of Over-the-Air Activation (OTAA) doing both activation and personalization in the same step. The node must be configured with the Device Address and the two session keys: the Network Session Key and the Application Session Key and is ready for exchanging application data with a specific LoRa network. Both keys are generated by the Application Server, which assigns each node with a different combination of keys. As the two keys are static and stored in the node, if one of them gets compromised, a new combination of keys must be generated and manually flashed in the node’s memory.

### 2.3 The Wireless Channel

LoRa empirical measurements described in the literature, as it will be seen in Chapter 3, are conducted mostly in flat terrain. In the present work we will measure the variation of RSSI with the distance and compare it with the existing literature, since the practical deployment areas may present significant differences in elevation. A harsher environment is expected to have an increased number of electromagnetic phenomena which affect a signal. These phenomena are the following:

- **Reflection** - when hitting a surface larger than the signal’s wavelength, some of the energy is reflected and some is absorbed by the reflective medium.

- **Refraction** - when hitting the surface of another medium, the refracted ray travels in the second medium with a different angle than the incidence ray according to the Snell’s law of refraction.

- **Scattering** - when an EM wave encounters obstacles in a medium with a size lower or of the same order of magnitude of the signal’s wavelength, e.g. trees in an urban environment, it may divide in multiple copies with different directions.

- **Diffraction** - when hitting a sharp edge, e.g. the corner of a building, or a slit, the wave front bends over the obstacle’s corner.

- **Absorption** - when traveling through a medium, an EM wave loses energy which is absorbed by the medium.
This section covers empirical propagation models used in Chapter 5. This analysis intends to provide an overview of how different models predict wireless signal behavior and variation of power loss in real environments.

2.3.1 Free Space Propagation

The free space path loss model characterizes the loss of signal power in the simplest attenuation scenario, corresponding to line of sight propagation with a single wave front. Equation 2.9 relates the received power, \( P_r \), with the transmitted power, \( P_t \).

\[
P_r = P_t G_r G_t \left( \frac{\lambda}{4\pi d} \right)^2 = P_t G_r G_t \left( \frac{c}{4\pi df} \right)^2 \tag{2.9}
\]

One may observe that the received power of a signal degrades exponentially with the second power of the distance, \( d \), due to spatial spreading. Also, it changes linearly with both transmitting and receiving antenna gains, \( G_t \) and \( G_r \) respectively. The previous equation can be written in decibel as in equation 2.10 which is more favorable to compare calculations with empiric measurements of RSSI.

\[
P_r [dBm] = P_t [dBm] - 20 \log_{10}(d) - 20 \log_{10}(f) - 20 \log_{10} \left( \frac{4\pi}{c} \right) \tag{2.10}
\]

2.3.2 Log distance

The log-distance path loss model is an evolution of the free space path loss model oriented at predicting path loss in environments with buildings or other obstacles. Equation 2.11 relates the received with transmitted power according to the log distance model.

\[
P_R [dBm] = P_T [dBm] + K [dB] - 10\gamma \log_{10} \frac{d}{d_0}. \tag{2.11}
\]

When using empirical measurements, \( K \) is calculated as the average attenuation at a reference distance \( d_0 \), using the free path loss model and assuming omnidirectional antennas,

\[
K [dB] = 20 \log_{10} \left( \frac{c}{4\pi f d_0} \right). \tag{2.12}
\]

The parameter \( \gamma \) can be obtained from the minimum mean square error (MMSE) applied to a set of empirical measurements of RSSI,

\[
F(\gamma) = \sum_{i=1}^{N} [M_{measured}(d_i) - M_{model}(d_i)]^2, \tag{2.13}
\]

in which, \( M_{measured}(d_i) \) is the path loss measured at distance \( d_i \) and \( M_{model}(d_i) \) is the expected value for a certain model that can be, for instances, the free path model. In such case, \( M_{model}(d_i) = K - 10\gamma \log_{10}(d_i) \). Function \( F(\gamma) \) is then differentiated relative to \( \gamma \) and set to zero,
\[ \frac{\partial F(\gamma)}{\partial \gamma} = 0, \quad (2.14) \]

to find the \( \gamma \) for which the quadratic error is minimum. It can also be obtained from typical path loss exponents taken from previous empirical studies as in Table 2.3, which contains several exponents for different environments, antenna heights and frequencies between 900 MHz and 1.9 GHz [23].

<table>
<thead>
<tr>
<th>Environment</th>
<th>( \gamma ) range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Urban macrocells</td>
<td>3.7 - 6.5</td>
</tr>
<tr>
<td>Urban microcells</td>
<td>2.7 - 3.5</td>
</tr>
<tr>
<td>Office Building (same floor)</td>
<td>1.6 - 3.5</td>
</tr>
<tr>
<td>Office Building (multiple floors)</td>
<td>2 - 6</td>
</tr>
<tr>
<td>Store)</td>
<td>1.8 - 2.2</td>
</tr>
<tr>
<td>Factory</td>
<td>1.6 - 3.3</td>
</tr>
<tr>
<td>Home</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2.3: Different path loss exponents \( \gamma \) for different experiment environments [23].

Regarding the previous table, within each interval, the exponent tends to be higher for higher frequencies [24] and lower for higher antenna heights [25].

### 2.4 Routing

Routing is the process of finding and selecting a path for messages from a source to a destination. If multiple paths are available, an algorithm is responsible for the selection process, which takes into consideration one or several metrics to choose the links and nodes that will form the route to the destination. Possible metrics include the number of hops, latency, reliability, bandwidth, energy and QoS.

Routing protocols are divided in two main classes: distance-vector and link-state. The two identify the types of messages exchanged and the knowledge each node possess regarding the network topology. In distance-vector routing protocols, a node shares a vector of distances (costs) to each destination it can reach. In link-state routing protocols, a node shares the state of each direct link to a neighbor node. In the end, all nodes possess information about every link and can construct a connectivity map of the topology. Distance vector protocols run with less message overhead and complexity. This is specially important in the presence of nodes with strict energy constraints, low processing power and limited memory and in a duty cycled network with legal throughput limits such as LoRa. In the present work, we consider a typical LoRa deployment, in which nodes transmit readings of sensors in a periodic fashion with a routing protocol relaying messages on intermediate nodes from the end-devices to gateways. We assume the routes to be stable and low or no mobility.

This section covers two distance vector protocols: Destination-Sequenced (DSDV) and Routing Protocol for Low power and Lossy Networks (RPL). Although other protocols were studied such as AODV\(^4\), they were not included due to be less preferable with respect to the features mentioned in the

\(^4\text{Ad-Hoc On Demand Distance Vector}\)
AODV, for instances, forms routes as needed and is suited at high mobility networks. Therefore, DSDV and RPL were concluded to offer the best trade-off in terms of low complexity, reduced number of routing messages and functionality.

2.4.1 Routing Protocol for Low-power and Lossy Networks (RPL)

RPL is a proactive distance vector routing protocol specified by the Internet Engineering Task Force (IETF) for Low-power and Lossy Network (LLN). A LLN is formed by energy, processing power and memory constrained nodes, in which the links connecting them are bidirectional, lossy, low rate and possibly unstable. The traffic patterns may be point-to-point, point-to-multipoint or multipoint-to-point.

The protocol does not rely on an a priori knowledge of the network topology. Instead, RPL utilizes control messages that are propagated in a distributed way between nodes and that contain their rank to the root node. The rank is a result of an Objective Function (OF) that minimizes a certain metric such as energy, hop count, Expected Transmission Count (ETX) or other. The topology comprises a Directed Acyclic Graph (DAG) in which its root has no outgoing edges such as node 7 and node 5 in Figure 2.10a. The previous may be constituted by several Destination Oriented Directed Acyclic Graph (DODAG) in which every node has the same destination such as node 7 in Figure 2.10b. A DODAG corresponds to an instance of RPL [26] and there is an instance for each different destination. The following control messages are used to form and maintain the network topology:

- **DODAG Information Object (DIO)** - Advertise an existent DODAG and its information to neighbor nodes. Content includes the rank of the broadcasting node, the Objective Function, DODAG-ID, etc. Message is sent downwards from the root.

- **DODAG Information Solicitation (DIS)** - Request information about existent DODAGs to nearby nodes if no DIO message was received yet. Message is sent upwards towards the root.

- **DODAG Advertisement Object (DAO)** - Request to join the advertised DODAG in a previous DIO message and send self identification within the network.

- **DAO-ACK** - Response to the previous solicitation.

![Figure 2.10: An example of a DAG and a DODAG.](image)

The formation of a DODAG starts at the destination which is considered the root node. The root advertises a DIO packet announcing the DODAG to its neighbors. Each neighbor then computes its own
rank based on the rank it received from the root node and on the cost to reach the root. These nodes may then join the existent DODAG by issuing a DAO packet which will be acknowledged back with a DAO-ACK. Each of the root neighbors will advertise the DIO with their own rank to their neighbors, those neighbors will then advertise to their neighbors and so on.

DIO packets are sent in the setup phase and periodically to maintain routes updated. Each node has a timer to control the transmission of DIO packets. When a timeout occurs, the DIO packet is sent. The value of the timer starts at the smallest interval $I_{\text{min}}$ and exponentially increases by doubling every time a time out occurs. The maximum number of times the timer should double, $I_{\text{doubling}}$, defines the maximum interval, $I_{\text{max}} = I_{\text{min}} \times 2^{I_{\text{doubling}}}$, after which, the time out period stays constant. When there is a significant change in the topology such as a change of parent, a new DAG sequence number or a parent detected to be down, the timer resets to its first value, $I_{\text{min}}$.

Consider that the network is stable for a while and the timer is equal to $I_{\text{max}}$. A new node that is turned on and intends to join a certain DODAG does not need to wait for the next DIO packet. As this could take a considerable interval of time depending on $I_{\text{max}}$, the new node can request information with a DIS packet to nearby nodes. These will reply with the DIO packets with information of DODAGs they belong to and the new node may join with a DAO packet.

Although RPL was designed for IPv6 over Low Power Wireless Person Area Networks (6LoWPAN), it can be adapted to a LoRa network. If one assumes that the traffic in such network is mainly oriented from the end-nodes towards the gateway, there will exist one RPL instance for each gateway.

### 2.4.2 Destination-Sequenced Distance Vector (DSDV)

The DSDV routing protocol was proposed by C. Perkins and P. Bhagwat in [27]. DSDV was proposed as a routing method to allow mobile computers, which are not in range with a base station, to be able to exchange data by forming an ad-hoc multi-hop network with other mobile computers with no detriment in the cases where a base station is available.

Each node in the network stores a routing table which contains entries with the destinations reachable by that node, the metric associated with a certain path, the next node in that path and all the remaining fields present in Table 2.4. Several paths can be maintained for the same destination to maintain network connectivity when link faults occur. Nodes exchange their routing entries with the information presented in Table 2.5. This example corresponds to the routing message advertised by node A to its neighbors, e.g. node B in Figure 2.11. If the routing message indicates a better route, a receiver, e.g. node B, will increment the metric field before advertising the route. DSDV considers two types of routing information packets based on the information they carry:

- **Full dump** - Contains all the entries in the routing table.
- **Incremental** - Only contains information that changed since the last Full dump packet, which is considered as significant information.

Full dumps are to be sent relatively infrequently. When changes are more frequent and there are too many incremental packets being sent, full dumps shall be scheduled more frequently. Every node
periodically transmits a full dump when a ticker expires, similarly to DIO packets in RPL. When significant information is available, e.g. detection of a stale entry in the table, a node transmits an incremental packet. There is no assumption regarding time synchronization mechanisms to schedule the advertisements as well as the phase relationship of updates between nodes.

When receiving new routing information, say from an incremental packet, the node compares these entries or routes with the ones stored in its own table, which resulted from previous routing information packets. If a route has a more recent sequence number (i.e. higher) than the one stored, the table is updated and the old route is discarded. If both sequence numbers are equal, the one with better metric (e.g. less number of hops) is preferred and the other is discarded or marked as an alternative and less preferable route.

The freshness of an entry in a routing table is ensured by means of a sequence number originated by the destination. Even numbers represent valid paths, while odd numbers represent an unreachable node due to a broken link or a node down along the path, which typically represents a mobile node that moved within the network. The sequence number may be incremented when a topological change is detected. A time stamp field is used to detect old entries in the table. This happens when a node stops receiving any routing information from another node and detects it through a time out. Figure 2.11 illustrates an example of a simple linear topology. Table 2.4 contains the routing table stored at node A and Table 2.5 the routing table advertised by node A to its neighbors.

![Linear topology example](image)

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Metric</th>
<th>Sequence Number</th>
<th>Install time</th>
<th>Stable Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>0</td>
<td>32</td>
<td>1000</td>
<td>Ptr_A</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>1</td>
<td>102</td>
<td>1200</td>
<td>Ptr_B</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>2</td>
<td>60</td>
<td>1200</td>
<td>Ptr_C</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
<td>3</td>
<td>32</td>
<td>1200</td>
<td>Ptr_D</td>
</tr>
</tbody>
</table>

Table 2.4: Example of a possible routing table at node A in the topology of Figure 2.11.

<table>
<thead>
<tr>
<th>Destination</th>
<th>Metric</th>
<th>Sequence Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>32</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>102</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>60</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 2.5: Example of a possible routing table message sent by node A in topology of Figure 2.11.

As incremental dumps are sent upon reception of fresher sequences numbers, fluctuations can occur. Consider a node A that can reach a node Z through two independent paths that only have in common A and Z. Further consider that the two paths differ in one hop. When node A issues its new
sequence number, it may happen that node Z receives the worse metric first through the path with more hops. Node Z would then issue an incremental packet due to a fresher sequence number despite the worse metric. Later, it would receive the delayed message originated at A with the same sequence number but better metric and would issue another incremental packet. These are considered fluctuations on the advertised routing information for a certain destination, which obviously constitutes a waste, specially considering that the effect is propagated by neighbor nodes of node Z.

To damp such fluctuations, the stable data field of Table 2.4 saves a pointer to a table which holds information on how stable a route is. This table contains the three following parameters:

- Destination ID
- Last settling time
- Average settling time

The average settling time is a result of the weighted average between the instants of arrival of the first and the best route for each particular destination. Each announcement received for each destination is used to compute the average settling time, where more recent events are given higher weighting factors than less recent events. In the end, a parameter must be selected to indicate how long a route has to remain stable to be considered truly stable and therefore ready to be announced. With such mechanism, node Z will wait some time based on the average settling time before sending its incremental packet to give time for all the routes with the same sequence number to be received and for the route to be considered truly stable. Moreover, it prevents these fluctuations from rippling through the network.

When new routing information is received, new entries are inserted and existent entries are updated and time stamped. As part of the routing advertisements between nodes is periodic, i.e. full dumps, new information is expected to arrive in the near future and therefore, if no information is received, the entry times out. In such case, the node has detected an unreachable node. The entry’s sequence number is then incremented to an even number and its metric updated to infinity. As this represents a topological change, the new information is broadcasted immediately, respecting the fluctuations’ damping mechanism.

One important property of DSDV is that its routes are loop-free which is proved in [27]. Firstly, for a topology comprising \( N \) nodes with stable routes, each route forms a tree rooted at the destination, with the branches expanding with the next hop field at each node. For all the destinations this may be represented as \( N \) disjoint directed trees towards each root, i.e. destination, which are, by definition, acyclic.

Outside the stable state, a loop could happen when a node updates its metric. Recalling the previous criteria for using new routes, a node updates its routing table either when the advertised route comes with a higher sequence number or with the same sequence number but better metric. If, for instance, node D of Figure 2.11 goes down, when node C does not receive the periodic routing advertisement from node D, it increments its sequence number and sets its metric to \( \text{nil} \). However, during the same cycle, after C notices D is down but before C advertises the route as being invalid, node B may possibly advertise that it can reach D. By using sequence numbers, node C may verify that, as the incoming
routing information from node B has a lower sequence number, it is certainly not up to date. Even if B and A were not updated yet, a data packet originated at node A and with destination to node D would be forwarded by B to C but would be discarded there as the next hop to D is nil.
Chapter 3

LoRa Related Work

This Chapter begins by describing an emerging open source and crowd sourced LoRaWAN network operator. It also provides an overview of the literature related with LoRa, which covers empirical measurements, scalability, collisions and multi-hop.

3.1 The Things Network

The Things Network\(^1\) is a crowd-sourced LoRaWAN network and community based in Amsterdam, Netherlands, with more than 16000 members as of May 2017. Any member is allowed to setup its own LoRaWAN Application Server and integrate it with existent Cloud services and storage. Gateways are deployed by community members and run a TTN packet forwarder to redirect messages from devices that are within its radio range to the network server and to the Application Server. A gateway does not necessarily need to be made public. Members can freely use the public gateways of other members. Figure 3.2 gives an idea of the spatial distribution of already deployed gateways.

To receive messages from end devices, a member must create an application and register each device individually prior to its deployment. When registering, the user selects the type of activation desired, fills the parameters needed for the chosen option and the Application Server generates the keys that must be flashed to a particular device. The user then just needs to deploy the nodes and if they are in range with a public gateway, the messages will start to appear in the application dashboard. Each packet sent from the gateway to the Application Server contains information regarding:

- The packet: node address of the sender, payload and frame counter.
- The transmission: time, frequency channel used, spreading factor, type of modulation, bandwidth, coding rate and airtime.
- The several gateways that received the packet: location and identification of the gateway and reception details.

\(^1\)http://www.thethingsnetwork.org
All the previous information can be observed in the TTN application dashboard, with the metadata of the second and third bullets being presented in JSON format as observed in Figure 3.1.

```
Metadata
{
  "time": "2017-06-30T00:21:37.563051185Z",
  "frequency": 868.3,
  "modulation": "LORA",
  "data_rate": "SF7BW125",
  "coding_rate": "4/5",
  "gateways": [
    {
      "gtw_id": "eui-b827ebfffe445318",
      "timestamp": 652545835,
      "time": "2017-06-30T00:21:37.568783Z",
      "channel": 1,
      "rssi": -114,
      "snr": 2.5,
      "rf_chain": 1,
      "latitude": 38.66696,
      "longitude": -9.15653,
      "altitude": 94
    }
  ]
}
```

```
Estimated Airtime
41.216 ms
```

Figure 3.1: Example of the metadata of a packet as observed in the TTN dashboard.

TTN has a distributed infrastructure growth and no subscription plan for end users. This competes, for instance, with the Sigfox network, which offers a subscription plan with low data rate and needs considerable investments in infrastructure in order to scale effectively.

![Geographical distribution of TTN gateways Worldwide.](image1)

(a) Geographical distribution of TTN gateways Worldwide. (b) Geographical distribution of TTN gateways in Europe.

Figure 3.2: Geographical distribution of TTN gateways. Blue colored marks represent deployed gateways. Green colored marks represent planned gateways.

The website is also a repository of various information also crowd sourced, which includes tutorials on how to setup gateways of various brands or end nodes, wikis explaining the architecture of TTN and LoRaWAN in detail and a forum for the community to discuss and share ideas or questions.
3.1.1 Architecture

Packets follow a path along a set of distributed components until reaching the Application Server. Figure 3.3 shows the architecture of TTN. Packets received by gateways are encapsulated and forwarded along with specific metadata related to the reception, e.g. the RSSI, SNR, SF, bandwidth, frequency channel and timestamp. Periodically, a gateway sends its GPS coordinates and number of packets transmitted and received. Gateways from different manufacturers can coexist in the TTN network and, as each one is generally developed for a specific backend, different protocols are expected to be used by different manufacturers. As such, bridges translate from each specific protocol to a common protocol used by TTN. The common protocol is used to communicate between the several components and consists in a set of gRPCs (Remote Procedure Call). A gRPC allows a component (client) to execute a function on another component (server) by defining an interface describing only the parameters and return types and thus offering abstraction of the function’s implementation language to the client that invokes it.

Figure 3.3: The Things Network architecture.

Router

When a packet reaches a local router, it is redirected to one or several brokers based on the device address contained in the packet. Each broker periodically advertises its list of address prefixes to routers, which performs a matching process and, as the device address is non unique and only identifies the node within a given network, redirects a packet to all brokers that advertise a matching prefix with the packet. With respect to downlink, a router schedules the transmission of each packet and redirects it to the gateway that has the best score if several gateways are in range with the target device. The score takes into consideration the following parameters to better distribute the traffic:

- RSSI of the last packet received from the target device, which represents an indicator of the prob-
ability of the the downlink packet being received by target device when sent from that gateway.

- **Airtime.** Regarding the duty cycle, packets with less airtime result in shorter wait times, which consequently allows a higher number of packets to be transmitted.

- **Gateway utilization.** A gateway may not receive while it is transmitting. When a gateway is under high uplink load, it will be less preferred in order to decrease the probability of losing uplink packets during the transmission time.

- **Already scheduled transmissions,** in order to reduce the delay of the downlink packet.

**Broker**  Brokers have a buffer which stores a packet for 200 ms in order to received duplicate packets that may arrive with some delay. The buffer delay of 200ms was determined based on experimental results: a maximum of 300ms and an average of 100ms was observed for the last duplicate to arrive to the Broker. Duplicate packets result from the reception by more than one gateway and may be used to locate a device, which saves energy by placing the calculations in the network. This can be accomplished either by using the RSSIs to estimate the distance to a gateway and then perform trilateration, or by using the difference between times of arrival and then perform multilateration. To evaluate if a packet is a duplicate from a previous one, the Broker computes the MD5 hash of the packet payload and compares it with the ones in the buffer, under the assumption of low probability of a non duplicate packet with the same hash arriving within the de-duplication period. Frame counters can not be used, since the encryption holds until the Application Server.

After de-duplication process, the broker needs to determine the application to which the device that sent the packet is registered to. As addresses are non unique, the network server sends a list of network session keys of all the devices that match the packet's node address. The broker then computes the MIC with each key provided in the list until a match is found with the MIC included in packet.

Brokers also check frame counters, which, as mentioned in subsection 2.2.2.3, prevent replay attacks. Packets with a frame counter lower than the last one received are discarded. On the other hand, packets with a frame counter much higher than the last frame counter are discarded. As specified by LoRaWAN, it results in the following condition:

\[
\text{Last } FC < FC < \text{Last } FC + 16384
\]

If the packets passed the counter and MIC checks, the duplicates need to be merged to discard redundant information. Transmission specific information such as the modulation, SF and frequency channel are common to all gateways that received the packet and therefore only one copy needs to be maintained. Reception information specific to each gateway is appended to the single packet to be sent to the handler. The device state is then updated by the network server which verifies, for instance, if the device is transmitting at the lowest data rate possible and can send MAC commands downlink if needed.

**Handler**  The handler is responsible for decrypting the uplink application payload. It was implemented as a separated component to be able to run in a private network. This way, the decrypted data does not leave the private network and therefore reinforces the application security. After being decrypted, the data may be converted to a friendlier format to be handled to the application, e.g. assign each byte of the payload to a variable or convert between units. The conversion is accomplished through a set of payload functions written in JavaScript, which are expected to be binded to a particular Frame Port in the future, in order to allow a separation of use cases within the same application.

Finally, the handler publishes the payload and metadata in JSON format to the topic `<app_eui>/devices/<dev_eui>/up` on a MQTT broker. Any application that subscribes to the previous topic will receive incoming packets, which greatly increases diversity of supported platforms. As expected, integration with widely known cloud providers such as Azure IoT Hub or Amazon IoT are also possible for data visualization, data analytics, storage or other services. For development of application solutions from scratch, SDKs are available for Go, Java, Node-RED and Node.js. The broad list of options to export the application data increases interoperability and ensures a good integration of TTN in the IoT ecosystem.

### 3.1.2 TTN Mapper

TTN Mapper\(^3\) is both a web and mobile application that allows a member of TTN with an already created application and a running device to be able to study the coverage in a given area by creating a color map based on the RSSI and on the geographical coordinates of the transmission. It also stores the data in a CSV format to be processed according to other needs.

TTN mapper subscribes to a TTN's MQTT broker specific application and device topics and then is notified when an uplink packet is received. The reception information of each gateway is stored individually along with the GPS coordinates provided by a GPS module connected to the transmitting device. By combining, for instances, the GPS coordinates and the RSSI of several receptions, it is possible to better understand the behavior of a LoRa signal in a given environment such as a city through empirical measurements. The application displays a colored map based on the RSSI at reception and other informations regarding a transmission point, e.g. transmission details or distance from that point to a gateway.

With the mobile app, the GPS of a smartphone may be used to geographically stamp the uplink packets under the assumption that the device is near the smartphone and obviously, that the smartphone has Internet connection. This frees the end-device from the need to have a GPS module. There is the possibility of publicly contributing to the global coverage map or to keep self coverage data private on a separated map, greatly facilitating the setup of an experiment. In this case, only the private data will appear on the map and the CSV used to draw it is available to the user, which may be used, for example, to address the RSSI behavior over distance for different gateways separately.

\(^3\)[http://ttnmapper.org/](http://ttnmapper.org/)
3.2 Empirical Measurements

In [28], two sets of empirical measurements were conducted in the coastal city of Oulu, Finland, which is mainly flat and in which the highest buildings are 12 floors high. One of the sets was performed on water in a moving boat and the other on land in a moving car at the maximum spreading factor of 12 and a power of 14 dBm. First, the authors reached a path loss exponent $n$ of 2.32 and 1.76 for land and water respectively. Secondly, on land, more than 80% of the packets were successfully delivered for distances less than 5 km and 60% for distances ranging between 5 to 10 km. On water, the success rate was approximately 70% for distances below 15 km.

In [29], the authors conducted two experiments within a building to study the packet loss and signal attenuation in the presence of reinforced concrete. The building has 7 entrances, 6 floors, a ground floor and a basement. One of the experiments was carried with the receiver on the roof and the other in the basement aligned with the 5th entrance (center), with the transmitter being put in each of the entrances (horizontal variation) and along the several floors (vertical variation) with 10 messages being sent from each position at $SF = 12$, $BW = 125kHz$ and output power of 20 dBm. The results show, firstly, that the roof position was reachable from the flats of all entrances at the receiver’s approximate floor level while the basement was not for the ground level transmitters. Secondly, even with the highest spreading factor and power being used, a LoRa gateway was not able to cover an entire reinforced concrete building, with the packet loss decreasing rapidly with each entrance.

In [30], an experiment in line of sight was conducted with 12 points of transmission at different distances from the receiver, ranging from 276 m to 8519 m. The data was collected with $SF = 10$, $CR = 4/5$, $BW = 250kHz$, a packet length of 10 bytes and no information regarding the power of transmission. The mean RSSI for the minimum distance was of $-80$ dBm, $-126$ dBm at 7482 m with a PER equal to zero and $-130$ dBm at 8149 m with a PER equal to 44 %. In the range between 276 m and 7482 m, all the packets transmitted were received and for the maximum distance, none of the transmitted packets were received. Packet lengths of 50 and 100 bytes were also tested under the same conditions and exhibited a worse behavior, experiencing non zero PER slightly earlier than the 10 bytes case.

3.3 Scalability

In [31], the authors conducted simulations using different combinations of frequency channels and spreading factors to see how the throughput and packet error ratio varied with an increase in number of nodes with a single gateway. First, the authors observed and compared the packet error rate and throughput for the following combination of settings:

- Single frequency channel, single spreading factor - total of 1 channel: $PER = 90\%$ for 1000 nodes.
- Single frequency channel, multiple spreading factors - total of 6 channels: $PER = 68\%$ for 1000 nodes. In comparison with the previous result, there is evidence of the orthogonality of the 6
spreading factors. The throughput achieved was better in this case than the previous, meaning it was similar to the previous experiment with $SF = 7$ and higher than when $SF = 12$.

- Multiple frequency channels, single spreading factors - total of 3 channels: $PER = 75\%$ for 1000 nodes. The result is higher than the previous, which may be explained by lower number of channels (6 vs 3).

- Multiple frequency channels, multiple spreading factors - total of 18 channels: $PER = 32\%$, for 1000 nodes. This result enhances the importance of LoRa spreading factors in the scalability of the technology.

The last result was then compared with a simulation of a network transmitting in a pure Aloha scheme - assuming colliding transmissions to be lost. The result was $PER = 90\%$ for 1000 nodes, much worse than when considering collisions as non destructive under certain limits. For the same number of nodes, the throughput was approximately 6 times lower in the pure Aloha scheme.

### 3.4 Collisions

LoRa modulation is referred to have non destructive concurrent transmissions when using the same physical (frequency) and virtual (SF) channel. To address this topic, [2] set up an experiment with two LoRa transceivers configured with a difference of power of 1 dBm to differentiate a strong from a weak transmitter. Other parameters were the following: $SF = 12$, $BW = 125 \, kHz$, $CR = 4/5$, payload was set to 10 bytes and the preamble to 12.25 symbols, which resulted in an packet airtime of 991 ms. The experiment consisted on varying the offset time of the strong transmitter relative to the weak transmitter, starting with the time interval correspondent to the airtime of one packet earlier and finishing with the airtime of one packet after as represented in Figure 3.4.

![Figure 3.4: Visual representation of the experiment in [2].](image)

Results are described below for a packet with 60.25 symbols:

- Both strong and weak transmissions were received successfully when the strong transmission takes place more than 57 symbols before than the weak transmission. In this case, the three last symbols of the strong transmission overlap the 3 first symbols of the preamble of the weak transmission.
• The strong transmission is received if it initiates less than 57 symbols earlier but not more than 3 symbols after the weak transmission. Within those intervals the weak transmission is not decoded.
• If the strong node initiates the transmission 3 symbols after the weak node, neither of the two is received up to 57 symbols. After that value, the strongest transmission is received but the weak transmission continues not to be received. This case corresponds to the overlapping of the strong transmission with the CRC of the weak transmission in the end of the packet. The weak transmission only starts to be received again when the strong transmission initiates approximately when the weak transmission has ended, which corresponds to more than 60 symbols given the packet length.

The experiment was repeated with both transmitters configured with the same power and in this case, either one is perceived as the strongest with similar results. The authors concluded that one of two concurrent transmissions is received with very high probability if the offset between the two is not greater than 3 symbols.

3.5 LoRaBlink

LoRaBlink is a multi-hop protocol for LoRa™ transceivers proposed in Bor et al. to address multi-hop, low-energy, resilience and low-latency, which are not addressed in current LoRa protocols such as LoRaWAN. It combines Medium Access Control (MAC) and routing, using beacons for time synchronization and to communicate the distance in number of hops to the gateway or sink.

The first beacon is transmitted by the sink and then there is a flooding process in which each node propagates to its neighbors the number of hops it is away from the sink. As a time synchronization mechanism, the first beacon indicates the start of an epoch. An epoch is constituted by $N$ slots, in which the first $N_B$ are beacon slots used to transmit hop distance beacons and $N_D$ are data slots. $N_B$ establishes the maximum number of hops. For instance, if a node received a beacon with a hop count of 4, it received it in the 4th slot and will convey (if this is the lowest value to date) its own hop count of 5 in the 5th slot. A powering up node stays in listen mode until it receives a beacon. After a beacon is received, the node stores the information and waits for the next beacon slot within the same epoch to transmit its hop count to the neighbors. Concurrent transmission can occur if two nodes, namely, node 1 and node 2, transmit their beacon within the same time slot. If a third node is within range of both previous nodes, as LoRa transmissions are non-destructive, either the transmission from node 1 or 2 will, with high probability, be decoded by the third node (depending on time offset and perceived signal strength at the receiver [2]).

For the implementation in [2], $N_B = 5$ and $N_D = 55$ were used with an epoch length of 5 minutes - 60 slots, each of 5 seconds. This gives each node time to receive, process and be ready to transmit in the next slot if needed. Each node was configured with $SF = 12$ and $BW = 125$ kHz and to transmit a packet with 10 bytes in a random slot within each epoch. Such configuration implies a very low data rate, with packets being spaced in time, at least, 1 minute and 31 seconds based on an airtime of 925
ms and according to the duty cycle limitations. The authors decided for a longer period of, on average, 5 minutes. When a message with data is sent, all the neighbor nodes check its origin in terms of hop distance and if they are closer to the sink, they relay the message. Several transmissions may arrive at the gateway through different relaying paths, which introduces redundancy and therefore wasted power consumption. Some assumptions were considered by the author when developing the protocol, such as low density and low traffic volume of the network. Nonetheless, LoRaBlink contains features worthed of being considered in the present work, e.g. the synchronization.
Chapter 4

LoRaWAN Multi-hop Uplink Extension

The scope of the present thesis is to design, implement and test a routing protocol which allows typi-
cal LoRaWAN end nodes, named Leaf Nodes (LN) throughout the Chapter, which are not in range of
a gateway, to send their data to intermediate nodes located between them and the gateway. Interme-
diate nodes, named Routing Nodes (RN), run an instance of the routing protocol, exchanging routing
information packets with neighbor RNs as needed in order to find routes to forward packets from LNs
to gateways. We assume RNs are not energy constrained (an energy analysis to assess its effective-
ness outside this assumption is left for future work). The implemented protocol will have its performance
evaluated in Chapter 5.

In this chapter, the developed routing solution will be firstly described using a top down approach. All
the components of a LoRaWAN architecture plus the multi-hop part are described to provide guidance
to future work on criteria of choice of hardware and component availability.

4.1 Routing Protocol

4.1.1 Overview

The protocol chosen to be implement was a simplified version of DSDV, described in Chapter 2. The dif-
ferences stem from the need to optimize a protocol originally designed for mobile nodes to a LoRaWAN
network. Firstly, RNs are not intended to be mobile, but static and extend the radio coverage of the
network in specific spots. For this reason, there is no pointer for stable information regarding routes in
each node as in the original DSDV. Topology changes are therefore expected to be very infrequent and,
when they do happen, it is either a new RN advertising in the network or a faulty one that went down
and generated time outs on its neighbors. Secondly, for the same reason of low mobility and infrequent
topological changes, there is no concept of truly stable routes. As fluctuations are not expected in a
LoRaWAN network in a regular basis, routes are considerable stable when they are advertised. Lastly,
there is synchronization between nodes, contrary to the DSDV, which is set by a gateway. Gateway
neighbors transmit their full dumps upon a beacon reception and not when a ticker expires. Other RNs
will transmit their full dump upon reception of the first full dump from a neighbor within each beacon
period as represented in the first and third sequence of arrows in Figure 4.1. Other full dumps may be received after a node transmits its own and, in this case, the information received after will be stored normally and sent in the full dump of the next beacon period.

Figure 4.1: Timing diagram of the implemented multi-hop solution. Different colors represent different types of nodes.

LNs construct a LoRaWAN packet and append a routing overhead to forward the packet within the multi-hop extension until being received by a gateway neighbor, which will drop the overhead and send only the LoRaWAN part. A representation is given in the second sequence of arrows with origin in Leaf Node 1 in Figure 4.1. The packet will appear in the Application Server as being sent by the node that sent it and its encryption is end-to-end as a LoRaWAN single hop network. Figure 4.1 represents an example of what is described in the current and previous paragraph. The three colors represent the three different components in the network. A gateway, in blue, sends its LoRaWAN beacon periodically, most commonly, each 128 seconds. RN1, in dark orange, is close to the gateway and will receive the beacon, update its routing table and transmit a full dump as soon as possible. RNs that did not receive the beacon, in light orange, receive the full dump from a neighbor RN, as RN1 for instance, update their routing table and transmit their full dump. LN1, in green, transmits data periodically. The data packet comprises the LoRaWAN part and the header until it reaches RN1, which contains a gateway in its routing table. RN1 removes the header and forwards only the LoRaWAN part so that the packet is correctly interpreted by the gateway.

Each RN stores a routing table containing a list of destinations, the metric to reach each destination, the next hop in the path, the sequence number issued by the destination and a time stamp to detect stale entries. Since there is only uplink traffic in the current implementation, a destination corresponds to a gateway, and only the gateway with the less number of hops (metric) is stored. The protocol is, however, prepared to support other destinations such as RNs or LNs, which allows downlink traffic in future work to be delivered from a gateway to a RN or LN along the correct path. When full dumps are received, the information is compared with the stored routing table, which is updated when there
are new destinations, better metrics for the same sequence number or a newer sequence number. The comparison is described in more detail in Subsection 4.1.3. Table 4.1 represents an example of the routing table at node 3 in a linear topology with three nodes and a gateway \((3 \rightarrow 2 \rightarrow 1 \rightarrow 0)\). The Install Time is the same for the three entries because node 3 received information from nodes 0 and 1 in the full dump issued by node 2.

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Metric</th>
<th>Sequence Number</th>
<th>Install time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>3</td>
<td>53200</td>
<td>279</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>16</td>
<td>279</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>279</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1: Example of the routing table at node 3 in the linear topology \(3 \rightarrow 2 \rightarrow 1 \rightarrow 0\).

RNs can be divided in those that are gateway neighbors and those that are not. The difference is whether a RN will change its channel parameters of reception to listen for a beacon in a periodic basis or not. Gateway neighbors schedule a short reception window right after receiving a beacon in which the channel parameters are changed to those used by the gateway to transmit the beacon as indicated in Figure 4.1. Other RNs also schedule the next instant from which they expected the first full dump of the next beacon period after which a RN should send its own full dump.

### 4.1.2 Packet Structure

The packets exchanged within the multi-hop extension may be of two types: full dumps containing routing information or data packets containing application data. Nodes exchange full dumps in order to form routes, with each node storing only the next node along the path and its metric to a given destination as in a typical distance vector protocol. Sequence numbers issued by each destination are also included as they ensure loop freeness. The full dump packet structure is represented in Figure 4.2.

![Figure 4.2: Packet structure of a full dump containing routing information with field lengths in bytes.](image)

The sender is identified in the Source ID field, which can only contain an ID from one of the receiving node’s neighbor. When a destination is added to the routing table of the receiving node, the Source ID field is added as the next hop to reach that destination, since that was the node which advertised it. The number of entries is included for the receiving node to know how to parse the entries. Each entry contains the ID of a destination, its metric and the latest sequence number generated by that destination as far as the advertising node’s knowledge. The list of routing entries represents a short version of the routing table of the sender node. The maximum length of a LoRa packet is 256 bytes which gives
a maximum number of \( \frac{256 - 3}{4} \approx 63 \) destinations in a single packet. In the presence of more than 63 destinations, fragmentation is needed.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of packet</td>
<td>Unicast ID</td>
<td>Destination ID</td>
<td>LoRaWAN</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.3: Structure of an application data packet within the multi-hop network with field lengths in bytes.

A leaf node sends a data packet with the structure of Figure 4.3. Note that there is no Source ID field in a data packet. That is due to the fact that the LoRaWAN part contains the node address, which is not encrypted and can be used to identify the sender. The Unicast ID field identifies the next hop node, i.e. to whom the packet shall be forwarded. This ensures packets will only travel to one path towards their destination when reaching a RN. When a LN transmits a packet, this field is assigned a value correspondent to broadcast as the LN does not possess knowledge regarding the network. This results in all nodes receiving, processing and possibly forwarding such packet. The destination identification generally corresponds to the ID of the gateway, i.e. 0. The remaining part, LoRaWAN, contains the data payload and the fields specified in Chapter 2. Activation by Personalization (ABP) must be used since the current implementation does not support downlink.

### 4.1.3 Components

In this section, the protocol is described from the point of view of a RN. The explanation follows a top down approach, starting with how the algorithm components were divided and which data structures were used and further providing more detail regarding each component. Each component is dedicated a separate subsection.

#### 4.1.3.1 Threads

Figure 4.4 represents a high level flowchart of the running program on RNs. The figure is divided in zones to better visualize which part performs each task. Grey boxes represent wait system calls, which stop the execution of the current thread and wait for other threads or an ISR to execute before continuing. In this particular case, the other threads or an ISR may send a signal or place a new mail in the mailbox, after which, the current thread will then continue its execution. In subsequent flowcharts from other components, the boxes that send a signal or put a mail in the mailbox have a background color associated with the thread which they send the signal to. Boxes with double lines, such as “Parse received packet” and “Transmit packet”, are represented in greater detail on separated flowcharts.

There exist three threads running in parallel with separated concerns:

- **Beacon Thread** - manages the beacon receiving window when the RN is a gateway neighbor. It controls a timer which fires an interrupt when it times out, indicating the start of the receiving window, and configures the channel parameters for the beacon reception.
Figure 4.4: Top level flowchart of the program running on RNs.

- Transmission Thread - manages all transmissions, including full dumps and data packets, and waits to transmit the next packet by calculating its air-time according to the duty cycle limitations. Packets to be transmitted are taken from a mailbox and are put both from the Beacon Thread (full dump) or by the Data Reception Thread (data packet).

- Data Reception Thread - Processes all received packets which are not beacons from gateways, including full dumps and data packets. Note that the box associated with parsing the received packet, with double lines, is common both to the Beacon Thread and to the Data Reception Thread.

When powered up, a RN will set its channel parameters to attempt the reception of a beacon, and will wait in reception mode for a time interval equal to the beacon period. If a beacon is not received within that interval and a timeout occurs, the Beacon Thread enters the beacon reception mode again. Since the receiving time on startup is so long, other packets in the same beacon frequency and spreading factor may be received. Whether a packet is received on that time with a CRC error or with a valid CRC but not a beacon, the consequence will be the same until reaching the maximum number of attempts. Beyond this maximum number of attempts the node is considered not a gateway neighbor and will only listen for full dumps and data packets. During normal operation, if a beacon reception timeout occurs, the next scheduled reception window will be longer to account for the previous timeout, which is equal to the time duration of the reception window, and will be started earlier. When a RN fails the reception of a beacon consecutively up to a certain limit, the RN stops being considered a gateway neighbor, and the gateway is removed from the RN’s routing table.
4.1.3.2 Interrupts

Interrupts are signals to the processing unit indicating an event which requires immediate attention. Interrupt Service Routines (ISR) are subsequently run with higher priority over threads. In the present implementation, interrupts may have two separate origins as represented in Figure 4.5:

- LoRa radio transceiver, colored in green, indicating a successful or timed-out transmission or reception, or a packet received with errors, i.e. CRC failed. These functions’ prototypes are defined in the SX1276 library¹.

- An MCU timer that timed out, colored in blue. There are two timers running permanently with an ISR associated: one to notify that the beacon window should start, in case the RN is a gateway neighbor, or, in case it is not, to reset a flag which indicates if the full dump was already sent during the current beacon period; The other timer, indicated as Backoff Timeout, is used to avoid collisions during a transmission and is explained later in Section 4.1.3.5.

Figure 4.5: Flowchart of the interrupts.

The ISRs mainly result in a signal being sent to a thread, whose color is associated with the box background color. All interrupts are configured with the same priority. Each ISR contains as few steps as possible to not cause delay between processing of events. A state variable is assigned a value which represents the type of interrupt that identifies the event within the transceivers’ different interrupts. Despite being very unlikely, if a transceiver’s interrupt occurs while the MCU is inside a Timer ISR, the second interrupt will wait for the first ISR to finish its execution.

¹This library provides an abstraction layer to communicate with an HopeRF or Semtech LoRa transceiver. For implementations with other LoRa transceivers, the appropriate library should be used.
4.1.3.3 Parsing of received packet

When a packet is received, the packet is verified to be one of the following possible packets: a full dump, a data packet or a beacon. The flow of actions is represented in Figure 4.6. This routine runs both in the Beacon Thread and on the Data Reception Thread. In either case, it is called after an interrupt signals either thread and after confirming that the state variable indicates a packet received with correct CRC. The three vivid colors in Figure 4.6 represent the mentioned 3 different cases of reception and the gray region represents a common set of operations for routing information packets. The different cases are explained below individually.

- Beacon from a gateway. The parameters contained in a full dump, namely the destination, sequence number and metric are extracted from a normal beacon packet specified in LoRaWAN.
The identification of the beacon is performed by comparing the first three bytes which correspond to the NetworkID field of the beacon. If there is a match with the expected result, the destination is set to 0, the metric to 1 and the sequence number is extracted from the 2 least significant bytes of the time field. This corresponds to an absolute maximum of 65535 in the sequence number field. As the time stamp is taken from a GPS module in each gateway, the synchronization is ensured and all nodes distancing 1 hop from a gateway will advertise the same sequence number.

- Full dump. The sender's ID and and type of packet are extracted from the expected fields according to Figure 4.2. The type is now checked to contain an ‘R’. Next, the packet is iterated to extract the several advertised destinations, their metrics and sequence numbers, which are compared with the existing routing table. A destination is added to the table if it does not exist and updated if for the same sequence number, the metric is better or if the sequencer number is greater. Whenever there is an insertion or update in the routing table, the row containing that destination is time stamped in the field Install Time, which is used to detect expired entries before sending a full dump. Additionally, if one of the advertised destinations has a higher but odd sequence number than the one in the routing table, that represents a destination that is now unreachable. Consequently, it is updated and removed from the table as soon as the next full dump is transmitted.

- Application data packet. If the packet contains application data, the field type should contain a 'M' character. According to Figure 4.3, the two IDs are extracted and, firstly, the Unicast ID is compared with the Unicast ID of the RN that received it. If there is a match, the RN iterates over its routing table to look for the destination specified in Destination ID field. If the destination is found, the node checks the next hop field indicated in the routing table. If the next node in the path is a gateway, the three bytes of overhead must be eliminated. If the next hop is not a gateway, the packet's Unicast ID field is updated to the value of the next hop indicated in the routing table. In both cases, which are highlighted in a pink colored area, the resulting modified packet is put in the mailbox for the Transmission Thread to transmit it.

4.1.3.4 Full Dump

Full dumps are transmitted right after receiving a beacon or the first full dump within a beacon period. The routing table is iterated and all entries are put in a full dump packet to be broadcasted to neighbors. The flowchart can be seen in Figure 4.7. While iterating, the program first checks if the current entry is stale by comparing its Install Time with the current value of a timer. If the difference is greater than a number of beacon periods, the entry is considered stale. If the entry is stale, the sequence number is incremented to indicate an unreachable destination, the row is added to the current full dump packet and then removed from the table. If the sequence number is even, that destination was received as unreachable from a full dump packet in the previous beacon period. As such, it was not transmitted yet and was maintained in the routing table to be transmitted in the next beacon period. A destination in this condition is added to the packet and removed from the routing table. If all destinations have been read, the packet is put in the mailbox for the Transmission Thread to transmit it.
4.1.3.5 Packet Transmission

Every transmission is accomplished by the Transmission Thread. This subroutine is represented in Figure 4.8 and corresponds to the box entitled “Transmit packet” in Figure 4.4. It starts with a Carrier Activity Detection analysis on the medium to detect a valid LoRa preamble. If a preamble is not detected, it assumes the medium is free and transmits. On the other hand, if a preamble is detected, the transmitter will follow an exponential backoff, i.e. wait a random amount of time until checking the medium again and doubling the upper bound of the waiting time when preambles are consecutively detected. The minimum waiting time, $a$, should be the correspondent to the air-time of one packet. The maximum, $b$, should be the minimum time plus the random number. The upper bound of such random number doubles due to an increment of the number of bits used to represent the number. When it finally transmits, the upper bound is reset. This mechanism tries to avoid collisions at the receiver and has its effectiveness assessed in Chapter 5.

4.1.4 Design Choices

This section provides a description of the choices taken when developing certain features of the protocol together with their respective justifications and alternative options.

4.1.4.1 Carrier Activity Detection

According to the datasheet of the transceiver, the current consumption in CAD mode, approximately 8.2 mA, is slightly lower than receiving mode, 10.3 mA. The difference comes from the fact that the CAD mode is divided in two phases: receiver and processing [20]. First, the transceiver is set to receiver mode for just over half time of the CAD period, in which the current consumption is between 11.5 and
13.8 mA (depending on the selected bandwidth). Next, the transceiver performs a correlation between the radio samples and a LoRa preamble waveform, in which the current consumption is between 6 and 8.3 mA (again depending on the selected bandwidth) [20]. Thus, one idea could be to use the CAD to continuously check the medium until a valid preamble is found instead of being in continuous receive mode. However, when the CAD finishes, an interrupt is fired and the CAD does not repeat even if no CAD was detected. As each CAD action needs to be sent individually, one after the other, this would imply that the MCU could not sleep by having to call the CAD requests. This contrasts with the receiving mode, in which the MCU can be sleeping while the transceiver is waiting for a packet. By checking the MCU current consumption in the datasheet and making some math, one may verify that the CAD mode it is not more energy efficient than the full receiving mode in this case.

### 4.1.4.2 Frequency Channels

LoRa operates in the 868 MHz unlicensed spectrum, which is free to use and bounded by region specific regulations depending on the frequency channels. Table 4.2 lists the EU regulations, where the maximum permitted duty cycle and power per sub band are indicated. The frequency sub band with the least of all requirements is typically used to transmit a beacon. Routing nodes transmit in one of the

<table>
<thead>
<tr>
<th>Frequency Bands (MHz)</th>
<th>Power (dBm)</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>863 to 868.0</td>
<td>14</td>
<td>0.1% or LBT+AFA</td>
</tr>
<tr>
<td>868.0 to 868.6</td>
<td>14</td>
<td>1% or LBT+AFA</td>
</tr>
<tr>
<td>868.7 to 869.2</td>
<td>14</td>
<td>0.1% or LBT+AFA</td>
</tr>
<tr>
<td>869.4 to 869.65</td>
<td>27</td>
<td>10% or LBT+AFA</td>
</tr>
</tbody>
</table>

Table 4.2: European regulations regarding the 868 Mhz frequency [1].

Figure 4.8: Flowchart of the transmission of a packet with Collision Avoidance using Carrier Activity Detection.
frequency channels between 868.0 and 868.6. As each RN may only listen on one channel at a time and there is no synchronization between nodes, there is no channel hopping between transmissions. This means that the frequency in which the nodes exchange the full dumps and the application data messages is constant and should be defined when RNs are flashed.

However, a RN which is neighbor of a gateway and encapsulates the packets in LoRaWAN, transmits to the gateway in a random channel in the set 868.1, 868.3, 868.5 MHz to comply with the LoRaWAN specification. Note that the gateway listens to the channels 867.1, 867.3, 867.5, 867.7, 868.1, 868.3, 868.5 MHz but frequencies between 867.1 and 867.5 have the 0.1% duty cycle limitation and, according to the LoRaWAN specification, are reserved for joining messages when OTAA is used instead of ABP.

4.1.4.3 Downlink

Downlink is not supported in the presented work, which means that it is not possible to send messages from the Application Server to a LN through the multi-hop network. LNs do not advertise themselves to RNs. Routing nodes can only know about LNs’ existence by storing the Node Address field contained in a LoRaWAN application data packet when received with a broadcast identification in the unicast field, which means that RN is the first hop along the path. Another approach would be to store the node IDs of RNs that forward the application data packet inside the packet itself. When the packet reached a neighbor of a gateway, the path would be extracted and stored in a cache. When receiving a downlink packet, the RN would lookup in cache for the path to the destination and the packet would be forwarded along the inverse path.

If we consider that the RNs are subject to duty cycle constraints, it becomes evident that sharing resources between uplink and downlink turns it difficult to guaranteed downlink packet delivery to LNs unless they are put in continuous reception. Otherwise, even if there was some sort of synchronization mechanism between a RN and a LN to schedule receiving windows at certain periodic instants, it would be hard to guarantee that the RN would be able to satisfy the transmission at those instants due to the packet having to possibly go through more than one RN.

The present work serves the purpose of developing a proof of concept of a multi-hop uplink LoRaWAN network maintaining LNs as Class A devices without receiving windows, which already suits a great variety of applications. Nevertheless, assessing the option to add downlink capability is relevant and is left for future work.

4.1.4.4 Beacon Timing

We initially thought of implementing an exponential timer between beacons similar to RPL. Considering the static profile of a LoRaWAN network, the topology would converge to a stable state faster during startup due to reduced time between beacons and would increase the period when no changes were noticed. The advantage would be the reduction of overhead and consequently higher data throughput with the consequence of faulty nodes being detected and advertised with a higher delay. The drawback was that the gateway’s packet forwarder had to have its beacon timer modified from a constant to a
dynamic period, which does not correspond to the LoRaWAN specification and counters the notion of extension.

Another solution for the beacon was to make it a downlink packet sent from the Application Server as an alternative to avoid modifying the gateway. However, there are two main drawbacks related with such option. If we consider two or more independent multi-hop systems, deployed at different instants, each one will originate an instance on the Application Server, which results in separate downlink messages for each independent system. If each independent multi-hop system needs a separate downlink packet, the downlink traffic destined to LoRaWAN single hop nodes will decrease, since gateways must respect the duty cycle. Furthermore, there is no guarantee of message delivery within a certain time. Note that this is due to the fact that the Network Server manages the gateway utilization and scheduling for downlink packets as explained in Chapter 2. As such, a RN at range of more than one gateway could receive the same sequence number at different instants. A RN that can reach a gateway through single hop and another gateway through a neighbor RN, may receive a fresher sequence number firstly from the neighbor RN than directly from the gateway if the delay is large enough. This may originate fluctuations on routes and unnecessary advertisements.

4.2 Prototype Description

The routing protocol described previously was implemented and tested in a prototype system. This section covers the choice of hardware and software to integrate the protocol in the several components of an LPWAN: the nodes, RN but also LNs, the gateway, the Network Server and the Application Server. Different options for each component are described with the respective reasons for choosing one over another. Also, the most significant steps towards reaching full functionality are described to provide guidance to future LoRa implementations.

4.2.1 Nodes

The selected LoRa™ radio module was the HopeRF RFM96 868/915Mhz RF transceiver module, which communicates with the Microcontroller (MCU) through SPI. There are several parameters, which can be configured by modifying the module’s registers such as the SF, CR and the Bandwidth (BW). Different operation modes are possible other than simple transmit and receive such as a sleep mode and a preamble sensing mode, i.e. CAD. Further information can be found in [15]. There is a great variety of boards in the market with an integrated MCU that could address the needs in terms of energy, memory, processing power and communication protocols of an RN. Typical Arduino boards such as Uno, Nano, Micro are not indicated for RNs due to their reduced flash memory (32Kb) and operating voltage, which is different from 3.3V. Arduino M0 and Zero could be used but were not considered due to their price range. The STM32L432KC board, which includes an embedded ARM® 32-bit Cortex®-M4 CPU, was selected. The selection was based on its cheap cost, ultra low power operating modes, user-friendly

---

2Serial Peripheral Interface
Software Development Kit (SDK) and Application Programming Interface (API)s, compatibility with the shields for Arduino Nano and extent of open source code and libraries. As it needs a power supply between 1.71 V and 3.6 V, it can be powered by two AA Ni-MH rechargeable batteries (1.2V each) connected in series without an additional power converter. Communication with peripherals can be established through several simultaneous USART\(^3\), SPI or I2C\(^4\), which is more than enough to support the RFM96 module (SPI) and a sensor. Further information of the STM32L432KC is available in [32].

The transceiver was soldered to a breakout board to convert from 2mm pitch to 2.54mm. Some pin headers were also soldered to plug the board into a breadboard along with the Nucleo board. The wirings were done with jumper cables as in Figure 4.9. The micro USB connector was used both to program the MCU and receive the input of a power supply, which was a power-bank with 5V or a generic 5V charger. To be able to use the connector as a power supply only, the solder bridge 1 (SB1), identified in Figure 4.9 by a red circle, must be soldered. The Nucleo board is Mbed enabled. Mbed is an online programming platform created by ARM targeting IoT with a range of services and features, including an online compiler, a crowd sourced repository of open source code, libraries and a forum for discussion.

For the RNs, three libraries are used:

- SX1276Lib library is used to communicate with the transceiver since the Semtech SX1276 registers are similar to the HopeRF RFM96.
- Matrix library is used to provide matrices as data structures to store the routing information as a table in each node.
- RTOS\(^5\) library is used to provide all the instruments of a typical operating system, e.g. threads, mailbox and mutexes.

To increase diversity, LNs are composed by the LoRa module indicated previously and an Nucleo Board or an Arduino Pro Mini, depending on the testbed. There are two nodes with the Arduino Pro Mini and four nodes with the Nucleo Board. The Pro Mini has 32kb which is enough to support the RFM96 library and some lines of code to send a few bytes from time to time. It also operates at 3.3V with an on board voltage regulator.

### 4.2.2 Gateway

There are several vendors of gateways in the market (e.g. Cisco, Link Labs, Kerlink, Multitech) with different price ranges, available features and business models. Some of the gateways are not open source, since they offer integrated solutions and linkage between the LoRaWAN devices and a Cloud Platform. The chosen option was to buy the radio transceiver and the processing board separately. The LoRa concentrator is the iC880A-SPI board manufactured by IMST, which supports up to 8 frequency channels at all spreading factors. This board has an integrated Semtech SX1301 as a demodulator and two SX1257 transceivers. The concentrator was assembled to a Raspberry Pi 3 with the help of

---

\(^3\)Universal Synchronous Asynchronous Receiver Transmitter  
\(^4\)Inter-Integrated Circuit  
\(^5\)Real-Time Operating System
an interface board but could have been wired with jumper cables. The result is shown in Figure 4.10. A GPS was added to obtain a time reference and locate the gateway, by connecting the transmitting pin of the GPS to a USART receiver of the raspberry, e.g. pin number 10 (GPIO15). The previous connection corresponds to the white wire of Figure 4.10b, the red and black wires to the 3.3V and Ground respectively and the blue wire links the GPS PPS output to the PPS input pin of the iC880A board. Ultimately, the antenna used was $\frac{3}{2}\lambda$ and 2 dBi gain.

(a) Sideview of the gateway without wirings.  (b) Top view of the gateway with the antenna and the GPS.

Figure 4.10: Different views of a LoRaWAN gateway comprising a Raspberry Pi (on the bottom), an interface board (in the middle), a iC880A radio module (on top) and a GPS module.

The Raspberry Pi was running the Raspbian Jessie Lite operating system. A detailed guide from the TTN Zurich was used to configure the packet forwarder. It consisted in connecting to the Wi-Fi Network, enabling SSH to access remotely without the need for a keyboard or screen, enable SPI which is the protocol used to communicate with the iC880A board and, finally, cloning a TTN repository that is a fork from the Semtech packet forwarder repository on github.

The packet forwarder is written in C language. When started, the main thread loads the configuration parameters of the gateway by parsing two files in JSON format:

- global.conf.json, contains information common to several gateways of the same operator, e.g. Network Server address.

\footnote{https://github.com/ttn-zh/ic880a-gateway/wiki}

\footnote{https://github.com/Lora-net/packet_forwarder}
- local_conf.json, contains specific information of each gateway, e.g. the EUI. In the presence of parameters defined in both files, the local_conf.json overwrites the parameters set in global_conf.json.

Important parameters are the server's address and upstream and downstream ports to which packets should be forwarded to or received from respectively. After loading the configuration, 4 additional threads are created:

- **Upstream traffic** - Forwards packets from a node to the Network Server.
- **Downstream traffic** - Forwards packets from a Network Server or an Application Server to a node, depending on whether the message contains application data or MAC controls. Note that, for each server, a different downstream thread is created.

And, if GPS is enabled in one of the previous JSON files:

- **GPS thread** - Communicates with the GPS module and parses the NMEA\(^8\) messages, extracting the time reference and coordinates.
- **Time validity** - Checks if the time reference is valid by comparing with the last value.

### 4.2.3 Network Server

TTN, as already described in Chapter 3, is a LoRaWAN network with an incorporated Network Server. Its variety of open content and code and the fact that it is free to use makes it very favorable candidate for academic purposes and therefore was chosen for the present work. To communicate from an end node to the Network Server, the device must be registered in the backend. An application was created in the TTN console, and a device was registered in it with a given EUI and an Application Key was generated for that particular device. Any gateway registered in TTN can receive messages from the device and forward them to the application. However, a gateway was also registered in the TTN console with its unique identifier, the EUI. When the packet forwarder starts to run in the gateway it is possible to confirm the status of the gateway as active in the backend.

### 4.2.4 Application Server

TTN dashboard does not persist data and only displays incoming packets in real time. Past events are eliminated and thus, there is a need of using one of the available types of integration to store data from tests for later analysis. The software programming tool Node-Red was chosen for such task, which allows the creation of an MQTT broker that subscribes to the specified Application EUI and one or more Device Addresses. Different functions, according to the test, were then written in JavaScript to parse the JSON packets and put the extracted data in a CSV file.

---

\(^8\)GPS format messages developed by the National Marine Electronics Association that contains the position, velocity and time.
Chapter 5

Experimental Results

The present chapter firstly presents a duty cycle analysis on the limitations and their impacts on the packet air time to provide clearance on the possibilities of a multi-hop technology in such limited network. Next, the coverage is experimentally tested in both line-of-sight and non-line-of-sight environments. Lastly, the protocol is evaluated with a reduced number of nodes in both a linear and a bottleneck topology.

5.1 Time on Air and Duty cycle

LoRa and other technologies that operate in the unlicensed spectrum have duty cycle regulations that limit the maximum data rate available per node. This upper limit is much lower than the maximum data rate imposed by the modulation and coding employed. In a multi-hop network where nodes with the same restrictions forward packets from other nodes, it is critical to identify what is possible when designing under such restrictions. To better understand the limits on the technology in the present work, the packet air times were calculated for various spreading factors while maintaining the payload constant. The on-air time $T_{OA}$, i.e. the time taken to transmit or receive a packet, has a significant impact on the final achievable throughput, since a node, to respect the duty cycle, has to wait $99 \times T_{OA}$ after a transmission to transmit again. Equation 5.1 was used to compute the several $T_{OA}$ [20].

\[
T_{OA} = T_{preamble} + T_{packet}
\]

\[
= T_{symbol}(L_{preamble} + L_{PHDR} + L_{PHDRCRC} + L_{PHYpayload} + L_{PHYCRC})
\]

\[
= \frac{2^{SF}}{BW} \left( NP + 4.25 \right) + \left( SW + \max \left( \left[ \frac{8PL - 4SF + 28 + 16CRC - 201H}{4(SF - 2DE)} \right] (CR + 4), 0 \right) \right)
\]

(5.1)

Where:

- $NP$, programmed preamble length in number of symbols.
- $SW$, length of synchronization word in bits. For LoRa, $SW = 8$ bits.
• PL, number of payload bytes (0 to 255)

• SF, spreading Factor (7 to 12).

• BW, bandwidth (125 to 500 kHz).

• CR, coding Rate (1, for 4/5, to 4, for 4/8).

• CRC, CRC = 1 if CRC is present, CRC = 0 otherwise.

• IH, IH = 0 for enabled header, IH = 1 otherwise.

• DE, DE = 1 if using low data rate optimization, DE = 0 otherwise. It is mandatory for SF = 12 and SF = 11 with BW = 125 kHz.

Six payloads were calculated, which generated six curves, ranging from 8 bytes to 256 byte lengths resulting in Figure 5.1. Knowing the packet airtime and for a typical duty-cycle of 1%, the times a device

![Graph of the variation of packet airtime with different lengths with the spreading factor.](image)

Figure 5.1: Graphic of the variation of packet airtime with different lengths with the spreading factor.

must wait before transmitting again were calculated and results provided in the graph of Figure 5.2. One can see that in both figures, for each payload length, the increase on the packet air time is exponential due to the fact that an increase in 1 unit in the SF doubles the amount of chips used to spread the signal. For a given SF, if the payload length doubles, the packet airtime also approximately doubles. Four tables in Appendix A list the packet air times and wait times for two bandwidths: 125 kHz and 250 kHz. On those tables one can observe the several possible periodicity of messages according to each choice of parameters. Figures 5.1 and 5.2 were calculated with a BW = 125 kHz, CR = 4/5 and considering the use of explicit header.

5.2 Empirical LoRa coverage measurements

This section provides a brief study of the behavior of a LoRa signal in a city with a different environment than those mentioned in Section 3.2 of Chapter 3. The city of Almada is constituted by trees, buildings
with different heights and a considerable amount of changes in elevation ranging between 5m and 115m. Two experiments were conducted, one to verify the variation of the RSSI in the aforementioned city and one in line of sight as reference.

For the first experiment, the gateway was positioned on the 7th and last floor of a building, with the building being on a location with an elevation between 50 and 70m, which may be considered as a possible deployment of a gateway for network operators, due to being located at one of the highest points in the city. A mobile node constituted by an Arduino Pro Mini and a RFM95 LoRa transceiver configured with a transmit power of 14 dBm, $BW = 125$ KHz, $SF = 7$ and $CR = 4/5$, was carried in a car and driven through the city at speeds not exceeding 50 km/h. TTN Mapper was used as tool to geographically stamp each packet that arrived at the Application Server, with the GPS coordinates from the smartphone.

Figure 5.3 shows the map of the city where the data was collected. The convergence point represents the location of the gateway, the colors indicate the RSSI in the gateway and the black dotted ellipses represent dark spots, where more than one transmission was lost. Lines that start in the points, but do not converge only to a single point in the figure, correspond to transmissions that were received by other public TTN gateways located in Lisbon. The white box represents the information of a particular reception, which includes reception parameters, gateway information and geolocation. The coordinates were used to compute the distance from each transmission point (sample) only to our gateway through the Haversine formula and then plotted along with the measured RSSI in the graph of Figure 5.4, where the blue points represent the samples. The red curve represents a fit to the data, knowing that the RSSI follows a logarithmic decrease with the distance, which results in the expression of Equation 5.2.

$$RSSI \approx -60.8511 - 14.2744 \log_{10}(d)$$ (5.2)

We then need to rewrite it in the form of the log-distance prediction model in order to extract the path loss exponent $\gamma$. We first calculate $K$ with the Equation 5.2 at a reference distance $d_0$ equal to 100m.
and then identify $\gamma$, which results in Equation 5.3.

$$RSSI \approx -89.39 - 10 \times 1.427 \log_{10} \left( \frac{d}{100} \right), \quad \sigma \approx 10 \text{ dBm} \quad (5.3)$$

The achieved $\gamma$ is equal to 1.427 which is lower than the achieved in [28] for the land experiment, where the terrain was mainly flat. This environment is expected to have more multi path components, shadowing and fading, due to having differences not only in building heights but also in elevation. The results in the graph also show a great variability. For very small distances, it was to expect an RSSI greater than -80. Considering that the antenna is on a 7th floor and that the measurements started at the ground floor, these low values can be explained by the height of the antenna and its radiation pattern. The used antenna was a $\frac{1}{2}\lambda$ monopole which radiates the maximum power in the horizontal plane and then rapidly decreases with the increase of the angle in the horizontal plane. This may also be an explanation for the maxima observed around the 3000 m, which was a high spot with line of sight to the gateway. Using the Fresnel zone formula to compute the radius of the first Fresnel zone, we obtain a value of 16.6m for a distance of 3200m. The place where the samples were taken was most certainly without obstacles inside its first Fresnel zone, contrary to the rest of the samples.

For the second experiment, the gateway was placed at a 1.5m above the ground and the node was put on top of the car and driven towards the gateway from approximately 1000m to 5m to evaluate the LOS situation. The street in which the experiment took place comprises houses in both sides, a few parked cars along the street, a few lighting poles and no moving cars other than ours. Similarly to the previous experiment, we extracted the expression and the parameter $\gamma$. The resulting expression can
be found in Equation 5.4.

\[
RSSI \approx -35.02 - 10 \times 3.815 \log_{10} \left( \frac{d}{100} \right), \quad \sigma \approx 5.6 \text{ dBm}
\] (5.4)

This $\gamma$ is much higher than one obtained in the previous experiment and higher than the exponent of the Free Path Loss. The signal suffers reflection mostly from the ground but also from houses on the side of the street and other objects. Note that the variability of the samples in this experiment was much lower than in the previous case and, as such, can be better modeled. This was expected due to the inexistent differences in elevation and diversity of objects blocking the line of sight ray, which results in the reduction of propagation phenomena such as shadowing.

### 5.3 Evaluation of specific functionalities

In this section, two proposed functionalities described in Chapter 4 are tested to assess their effectiveness and how parameters can be chosen to improve it: CAD (see Section 5.3.1) and Beacon reception window management.

#### 5.3.1 CAD in Collision Avoidance

A CAD implementation to avoid collisions was described in Chapter 4. Three experiments were conducted to evaluate the effectiveness of this feature, i.e. if CAD can be used to successfully avoid collisions. The first experiment is used as reference and comprises one transmitter without CAD and one receiver. The second experiment comprises two transmitters without CAD and one receiver. The third
The second and third experiments consisted on having two synchronized transmitters which would transmit randomly within a given interval of time. The time intervals were varied from 8 ms to 128 ms in the set 8, 16, 32, 64, 128 with 2000 transmissions in each\(^1\). The transmitted packets had a payload of 20 bytes, with explicit header, \(CR = 4/5\), \(BW = 250\) kHz and a programmed preamble of 8 symbols. By using Equation 5.1, the resulting air time becomes \(ToA \approx 28.29\) ms. Comparing the packet \(ToA\) with the testing intervals one may observe the following:

- For 8 and 16 ms, collisions are certain
- For 32 ms, collisions are highly probable
- For 64 and 128 ms, collisions are less probable

At boot time, the two nodes are configured to be in reception mode and wait for a packet from a third node, which is no more than a trigger. When this packet is received, a Timer is configured to notify the MCU at the start of each second that a packet must be transmitted, and the test begins. The time between transmission cycles, 1 second, needs to include a minimum number of backoff times in case of consecutive carrier detections. Since the same code runs on both transmitters, they are expected to keep synchronization. To verify this, we timestamped each reception and plotted the LoRaWAN uplink packet counter of each received packet with time, which is represented in Figure 5.5. Only the 8ms and the 128 ms were plotted to keep the figure simple. We can see the linearity in the packet reception through time. The PRR for the experiments with CAD (triangles) and without CAD (circles) is represented in Figure 5.6. The colors represent the different transmits intervals. The points in this figure correspond to the last point of the curves in Figure 5.5, normalized to the total number of transmitted packets. As

\(^{1}\)In order to reduce the time needed to run the experiment, the frequency was set to 869.525 MHz, allowing a 10% duty cycle.
expected, in the experiment without CAD, the PRR decreased with the shortening of the interval, due to an increase in the probability of collisions. Also when shortening the interval, the distance between points for the same intervals, with and without CAD, is higher. The results also show that the PRR stays approximately constant at 27% for intervals where collisions are highly likely and certain. For every interval, being collisions probable or certain, the CAD is observed to greatly increase the number of received packets when compared with the case where no CAD was used. The obtained numeric results are presented in Table 5.1.

<table>
<thead>
<tr>
<th>Time interval (ms)</th>
<th>PRR Without CAD</th>
<th>PRR (With CAD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>27.1%</td>
<td>79.6%</td>
</tr>
<tr>
<td>16</td>
<td>26.0%</td>
<td>84.3%</td>
</tr>
<tr>
<td>32</td>
<td>26.8%</td>
<td>90.1%</td>
</tr>
<tr>
<td>64</td>
<td>53.4%</td>
<td>94.8%</td>
</tr>
<tr>
<td>128</td>
<td>71.6%</td>
<td>96.8%</td>
</tr>
</tbody>
</table>

Table 5.1: PRR in percentage for transmissions with and without CAD for different intervals and 2000 transmitted packets.

In [2], two transmitters with different transmit power were used. In the present experiment, nodes were configured with the same transmitted power but were placed at two different distances from the receiver, which leads to a weak transmission (more distanced from the receiver) and a strong transmission (less distanced from the receiver). Figures B.1, B.2 and B.3 in Appendix A represent the distribution of RSSI for the various scenarios. It can be observed that for intervals where collisions are almost certain, the graph without CAD shows a clear concentration of RSSI in the higher (less negative) values of RSSI. This indicates that only the strongest transmission was received, which corresponds to the capture ef-
fect. With CAD, the distribution of RSSI flattens and distributes both around -37 and -53 for the 8ms interval, which is due to the fact that the transmissions begin to alternate. When the weak transmission is transmitting, the strong transmission will detect it and wait, allowing the weak transmission to be received.

5.3.2 Beacon Reception Window Accuracy

The beacon is transmitted by gateways on a different frequency than those used by nodes to transmit application messages in LoRaWAN. Thus, the time a RN stays in reception mode in the beacon channel, may affect the performance of the multi-hop network, since nodes receive and transmit in a single channel at a time and may lose application packets if the beacon reception interval is too long. To evaluate how shrinkable this interval can be, an experiment was set up comprising 4 independent tests. Each test consisted in configuring a single RN with a different beacon period (64, 128, 256, 512). In each test we measured the time interval between the start of the reception window and the actual reception of the beacon. The results are shown in Figure 5.7, where the obtained times for each beacon period are represented. In each graph, the central bar represents the median, the extremes of the box represent the lower and upper quartile from left to right respectively, and the outer dotted lines represent the minimum and maximum of the samples. The distance between minimum and maximum in each case was not more than 6ms. We can clearly see that, for the used hardware, the Nucleo’s clock drifts

![Figure 5.7: Time interval between the start of the Beacon reception window and the start of the actual reception.](image)

relative to the gateway’s clock and that the delay increases with the beacon’s period. We computed a
linear regression over 4 points given by the mean of time until reception for each beacons’ period. The resulting expression with the corresponding error of estimate $\sigma_{est}$ is displayed on Equation 5.5 plus a reference time $t_0$ equal to 700 ms. This time corresponds to the instant at which the reception window started, measured from the end of the beacon period. For instance, in each of the four experiments, the window started 700 ms before $T_{beacon}$.

$$t_{RX \ start} \approx -0.575 \times T_{beacon} + 683.952 + t_0 \ [ms], \quad \sigma_{est} \approx 0.133 \ ms$$ (5.5)

$t_{RX \ start}$ represents the start of the next beacon reception window relative to the beacon period. For instance, for a beacon period of 256 seconds, $t_{RX \ start \ 256} \approx 536.5 - 700 \approx -163.5 \ ms$, which indicates that the next reception window should start at instant $256 + t_{RX \ start} \times 10^3$. By inspecting the slope of Equation 5.5, we conclude that the drift increases approximately 0.6 ms per additional second in the beacon period.

### 5.4 Multi-hop Performance Evaluation

The current section presents a study on the performance of the implemented routing protocol. PRR and throughput were the indicators used to assess such performance. In each of the following experiments, we logged each transmission in each RN and also on the Application Server. In each RN, a counter for the number of received packets from each LN was maintained and, in the Application Server, each received packet, containing the packet counter, node identification, time and other parameters, was stored in a *.csv file with a Node-Red script. This section describes the results obtained for the following scenarios:

- Single hop;
- Two hops with a simple forwarder - no routing protocol;
- Linear topology with routing protocol, from two to five hops;
- Bottleneck topology with routing protocol, two hops with three LNs.

From Figure 2.7 we can see that a LoRaWAN adds, at least, 12 bytes of overhead to a packet. As the Frame Port field was used, the total LoRaWAN overhead sums to 13 bytes. To choose the payload length for subsequent tests, we take other technologies for reference, such as Sigfox which allows 12 byte messages. We choose a total length of 31 bytes: 15 bytes of application data, 3 bytes of routing overhead and 13 bytes of LoRaWAN overhead. A bandwidth of 250 kHz was chosen due to being the maximum bandwidth allowed to be received by the gateway in use. By using the highest bandwidth for the same payload length, the information to be transmitted is more distributed in frequency than in time, resulting in a lower airtime than for lower bandwidths and consequently in a higher throughput. The rest of the configured parameters were $CR = 4/5$, a programmable preamble of 8 symbols and the use of explicit header. In each experiment, a total of 500 packets were transmitted.
5.4.1 Single Hop - LoRaWAN Benchmark

For this first experiment, we assessed the performance of one node reaching a gateway through single hop such as in a typical LoRaWAN network. The maximum theoretical throughput is limited by the technology, by the duty cycle and by the LoRaWAN overhead. With the already mentioned transceiver’s configurations, by applying Equation 5.1, we obtain a packet airtime \( T_{oA_{ref}} = 33.41 \text{ms} \). If we consider that, in order to comply with the duty cycle limitation of 1%, at maximum, a device may transmit a single packet in the interval \( T_{oA} \times 100 \), we get the maximum theoretical throughput for the given parameters as follows:

\[
\text{Throughput}_{ref} = \frac{L \times 8}{T_{oA_{ref}} \times 100} = \frac{15 \times 8}{33.41 \times 100} \approx 35.92 \text{ bit/s}.
\]  

(5.6)

The resulting experimental throughput, calculated based on the time between the arrival of the first and the last packet, \( \Delta t \), and on the total number of packets received \( N \), resulted in Equation 5.7. The PRR was 99.4%.

\[
\text{Throughput}_{ref} = \frac{L \times 8 \times N}{\Delta t} = \frac{15 \times 8 \times 497}{1685} \approx 35.39 \text{ bit/s}.
\]  

(5.7)

Such result was similar to the theoretical result of Equation 5.6. One of the reasons that explains the small difference is that three packets were lost, which is not considered in the theoretical equation. This result may be used as a reference for the tests that follow.

5.4.2 Two Hops without Routing Protocol

The second experiment consisted on evaluating the throughput and PRR when one node forwards LoRaWAN packets from the LN to the gateway. A diagram representing the present experiment is given in Figure 5.8. The LN1 (green) will send LoRaWAN packets with no overhead, which will be received by the forwarder (orange) and retransmitted to the gateway. We do not name the forwarder node an RN due to not running the implemented routing protocol. Instead, the forwarder solely transmits what it receives. The resulting throughput, as calculated according to Expression 5.7, and the PRR, both calculated at the Application Server, can be observed in Table 5.2. The resulting PRR and throughput are slightly

<table>
<thead>
<tr>
<th>PRR</th>
<th>Throughput (bit/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>97.6%</td>
<td>34.73</td>
</tr>
</tbody>
</table>

Table 5.2: PRR and throughput for the experiment with two hops and a simple forwarder.
worse than the single hop case. The datasheet lists a PRR of 99% for the used radio module, which indicates that more losses are expected with the introduction of one more hop.

### 5.4.3 Linear Topology

In this subsection, we analyze the routing protocol overhead and the impact of increasing the number of hops in a linear topology, from two hops to five hops, as represented in Figure 5.9. The figure represents the four conducted tests, in which the number of RNs between the gateway and the LN5 was increased by one. The numbers in brackets correspond to the number of the test.

![Figure 5.9: Linear topology test configuration. In each test the number of RNs was increased by 1.](image)

The first experiment was similar to that of the previous subsection, with the difference that the forwarder node had the routing protocol implemented, constituting an RN. In this case, the forwarder node waits for a beacon and sends its full dump. Also, each packet originated from the LN to the RN has three bytes appended to the LoRaWAN packet. Hence, the throughput and PRR are expected to decrease.

The subsequent experiments consisted of increasing the number of RNs between the LN and the gateway, one by one. RNs that were in the middle were configured to only fire the reception interrupt if a routing packet came from an adjacent RN neighbor or, in case of a data packet, if the Unicast ID field had their ID. For the RN adjacent to the LN, the previous Unicast ID condition was changed to the Broadcast ID. This way, we are sure that the path taken by a data packet passes through all the nodes.

We first let the topology be constructed by switching the nodes on and waiting for the first beacon to be received. The first beacon will trigger the series of full dumps within the multi-hop topology. The obtained results for the PRR and the throughput, again calculated from Equation 5.7, are presented in Table 5.3. We observe that from the simple forwarder with no protocol to the RN with implemented protocol, the throughput decreases 6.5%. This may have two reasons. The first comes from the fact that the three bytes of overhead appended to the packet by the LN increase the packet air time, also

<table>
<thead>
<tr>
<th>Number of RNs</th>
<th>Application Server</th>
<th>RN1</th>
<th>RN2</th>
<th>RN3</th>
<th>RN4</th>
<th>Throughput (bit/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>98.2%</td>
<td>99.4%</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>32.48</td>
</tr>
<tr>
<td>2</td>
<td>94.8%</td>
<td>95.0%</td>
<td>96.8%</td>
<td>-</td>
<td>-</td>
<td>31.36</td>
</tr>
<tr>
<td>3</td>
<td>93.4%</td>
<td>94.2%</td>
<td>96.2%</td>
<td>96.8%</td>
<td>-</td>
<td>30.87</td>
</tr>
<tr>
<td>4</td>
<td>91.6%</td>
<td>94.2%</td>
<td>94.4%</td>
<td>96.2%</td>
<td>98.0%</td>
<td>30.26</td>
</tr>
</tbody>
</table>

Table 5.3: PRR and throughput for 4 experiments with a linear topology.
increasing the denominator of Equation 5.7 without increasing the numerator, which only takes into consideration the application payload. The second reason may be the packet loss, since the RN is not only receiving data packets from the LN but also switching frequency at a period of 128 seconds to received the beacon. During this interval, the RN cannot receive packets originated by the LN. Moreover, full dumps have priority over data packets and, if the queue is full, a packet must be dropped in order to transmit the full dump every 128 seconds. However, as shown in Table 5.3, the PRR was not lower than the case with the simple forwarder. Regarding the results with different number of RNs, we found that the PRR at the Application Server decreases on average 2.2% and the throughput 2.3% with each new hop.

5.4.4 Bottleneck Topology

The bottleneck topology is constituted by a RN within the range of the gateway and a set of neighbor LNs. This case is particularly important in a duty cycled network, where a RN is under the same restrictions as the other nodes. Consider the experiment’s topology depicted in Figure 5.10. If each of the $K$ nodes

![Figure 5.10: Bottleneck topology with one RN and 5 LNs.](image)

transmits at the maximum rate of one packet per time interval of $ToA \times 100$, the queue of the RN, in yellow, increases in every such interval, $ToA \times 100$, by $(K - 1)$ packets. This results from the fact that the RN receives the $K$ packets, one from each LN, minus the packet that it forwards to the gateway. Thus, the maximum delay $t_{delay}$ for the $N_{th}$ transmitted packet comes,

$$t_{N_{th}} = (N - 1) \times ToA \times 100.$$ (5.8)

Unless every LN only transmits a packet every $ToA \times 100 \times K$ seconds, the RN will suffer congestion and will eventually run out of memory and start to discard packets. The question is whether this deployment can still reach the total throughput of a single node, i.e. which drift will exist between the throughput of an LN directly connected to the gateway and the throughput of an RN with more than one LN connected to itself. Moreover, if the throughput per node is of use when compared with similar technologies, which application scenarios it can fit and what is the value of $K$ leaf nodes for which the throughput becomes too low.

We have set up an experiment constituted by three LNs and one RN, in which we maintained the same packet parameters but configured each LN to transmit every $ToA \times 100 \times 3 = 10.79 \text{ seconds}$,
where the packet air time is $T_{oA} = 35.97 \, ms$. Also, the transmit power was decreased to the minimum (0 dB). All LNs were configured to be in reception mode when started. Upon reception of the first packet (the full dump from the RN), each LN started a periodic timer at the same time to be synchronized with the other two LNs. Each transmission occurs in a random interval within the first 6.4 seconds of each interval of 10 seconds. The LNs and the RN were positioned at a distance of 5 cm from each other and at the same distance from the RN, which introduces the possibility of collisions. The obtained results for the PRR and throughput are presented in Table 5.4.

<table>
<thead>
<tr>
<th>Application Server</th>
<th>RN1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PRR</strong></td>
<td></td>
</tr>
<tr>
<td>LN1</td>
<td>87.2%</td>
</tr>
<tr>
<td></td>
<td>88.7%</td>
</tr>
<tr>
<td><strong>Throughput (bit/s)</strong></td>
<td>29.56</td>
</tr>
</tbody>
</table>

Table 5.4: PRR and throughput for the experiment with a bottleneck topology composed by three LNs and one RN.

The obtained throughput at the Application Server for the ensemble of three nodes is lower for the bottleneck topology than for the linear topology with one RN. The difference may be explained by the difference in the PRR, which is also lower. The difference of 9.5% in the PRR may be explained by concurrent transmissions from the LNs colliding at the RN. Despite the CAD mechanism being implemented in the LNs to avoid collisions, and although it dramatically improves the PRR as already seen, it is still not able to avoid all collisions. If a transmission is not detected by the CAD and the LN transmits, then both packets may interfere destructively and may not be received by the RN. For an interval of 128 ms the PRR was 96.6% in the CAD test. The interval in the current test is 6.4 seconds, which is fifty times larger than the 128 ms interval, but, on the other hand, it also has another transmitter added. The probability of collisions decreases as the interval increases, but it increases with the addition of more transmitters. When observing the independent PRRs for each of the three LNs, we observe that the differences are at most 1.4%, which is not considered significant.

Another reason for such a decrease when compared with the linear topology is that packets from either of the LNs may arrive at the RN while it is transmitting a packet, either a full dump, where there is no difference with the linear topology, or an application data packet. Note that in the linear topology this was unlikely to happen considering that there was no randomness associated with the transmissions of the LN. Hence, when a packet was received by a RN, it would be quickly processed and forwarded if the wait time for the previous packet had been fulfilled. As the RN removes the routing overhead, the wait time is lower for the RN than for the LN and thus, when the RN would be forwarding the packet from the LN, the same LN would be waiting to transmit the next packet. This stability is only compromised when a beacon is received and a full dump must be sent.

All the tests so far were conducted with a frequency channel separation between the gateway and the RN, and the RN and the other RNs or LNs. A LN transmits, for instance, in 868.5 MHz. The packet travels
through the multi-hop network in the same frequency until reaching a RN that is a gateway neighbor. The last RN in the path, transmits on a different frequency to prevent collisions at the gateway, due to the nodes being close to each other. We set up a last test to determine the effect of having the three LNs associated with the bottleneck possibly colliding with the transmission from the RN to the gateway since no frequency separation was used during this test. Note that when the RN performs a CAD and detects activity, it backs off, loosing the incoming packet from the LN and causing a delay which contributes to decrease the throughput. The results are presented in Table 5.5.

<table>
<thead>
<tr>
<th>Application Server</th>
<th>RN1</th>
</tr>
</thead>
<tbody>
<tr>
<td>LN1</td>
<td>LN2</td>
</tr>
<tr>
<td>PRR</td>
<td>89.0%</td>
</tr>
<tr>
<td>Throughput (bit/s)</td>
<td>87.4%</td>
</tr>
</tbody>
</table>

Table 5.5: PRR and throughput for the experiment with a bottleneck topology composed by three LN and one RN using the same frequency in the links LN → RN and RN → Gateway.

The throughput and the PRR were lower than previously, confirming the expected lower performance of the not using frequency separation. The difference between the PRR of the three LNs was at most 10.8% between LN2 and LN3, measured at RN1. Such difference started in link LN → RN and was not significantly changed in the link RN → Gateway. This could be attributed, for instance, to the mean of RSSI values at the RN being the highest for LN2 and the minimum for LN3. As we keep track of the receptions at the RN, each identified by the LN and with the corresponding RSSI, it was possible to plot the result in Figure 5.11. In the boxplot we can see that the RSSI values of LN2 were located in the highest values, while the RSSI measurements of LN3 were located in the lowest values.

![Figure 5.11: RSSI measurements at the RN for the test in a bottleneck topology with three LNs and one RN using the same frequency in the links LN → RN and RN → Gateway.](image-url)
We proposed a routing solution which can be used with the already existent LoRaWAN networks for uplink transmissions. The protocol can be used in the presence of dark spots to extend the coverage of a gateway, either when the deployment of another gateway is not possible or when the deployment of a gateway does not justify the investment. The protocol routes LoRaWAN packets from LNs to the nearest gateway using the number of hops as metric. LNs append a small overhead to the LoRaWAN packet in order to forward it through the intermediate nodes (RNs).

In one of our tests, we proved the existence of dark spots in a city with different building heights and changes in elevation. The gateway was placed in one of the highest points in the city, however, several losses of consecutive packets were observed. In most of the cases, the receptions immediately before a dark spot did not indicate a RSSI close to the sensitivity threshold for that spreading factor. Thus, one could not predict the loss of packets that followed. Smart meters that use a wireless link and are located under the ground or in locations with strong electromagnetic attenuation may have the same or worse behavior. For the aforementioned scenarios, a multi-hop solution is an economic way of extending a gateway coverage.

The protocol was assessed using the PRR and throughput and was tested in linear and bottleneck topologies. It was observed that:

1. The PRR decreased on average 2.2% with each new added hop. The PRR varied from 98.2% (one RN) to 91.6% (four RNs).

2. The throughput decreased on average 0.74 bit/s per each new hop. The throughput varied from 32.48 bit/s (one RN) to 30.26 bit/s (4 RNs).

In the bottleneck topology, the obtained PRR was 88.7% and the throughput was 29.56 bit/s. The PRR for each LN did not significantly differ. The experience was then repeated using the same frequency channel in the links LN → RN and RN → Gateway. In these conditions, the PRR for each LN was significantly different, with a maximum difference of 10.8%. This is explained by the differences in the average RSSI of the several LNs at the RN. The overall results of the multi-hop experiments show, firstly, that a routing solution complying with the LoRaWAN specification is feasible. Secondly, the obtained
values for throughput are satisfactory for most of the applications in IoT, in which uplink packets contain mostly sensor readings. Also, for non critical applications, a PRR of 90% may still be acceptable without worsening data analysis and machine learning applications.

CAD is designed as an alternative to continuous reception mode when using packets with long preambles. In the current protocol, we implemented each transmission to be preceded by a CAD analysis to avoid that the packet to be transmitted collides with other transmission. This feature was observed to greatly improve the PRR (from 26.5% to 81.9%). We therefore conclude that the CAD can successfully in collision avoidance where two transmitters are at range of each other.

The beacon reception window for the used hardware, Nucleo board, needs to be chosen considering the beacon period, in order to prevent the loss of beacons because of short reception window or in contrast, the loss of data packets due to a excessive long reception window. We tested four beacon periods: 64, 128, 256 and 512 and found that the reception window should be increased by 0.6 ms for each second added in the beacon period.

We conclude that a LoRaWAN multi-hop uplink extension is feasible even with the duty cycle restriction of the 868 MHz band. This solution delivers an improved range at the cost of a reduction in the throughput and PRR, which benefits applications with sporadic traffic, in which the loss of a few packets is not critical, and which are located in areas with clear identified dark spots.

6.1 Future Work

There exist some points that can be explored to continue and possibly upgrade what was accomplished in the present work:

- The RNs were assumed to be powered by an external power source. An analysis on the energy consumption on these nodes may possibly extend the application’s scenarios to remote areas where an AC plug is unavailable but, for instance, a solar panel maybe used to power a RN.

- Eventually add downlink capability in order to allow LNs to be connected to actuators.

- Integrate a Transmit Power Control (TPC) mechanism , i.e. adequate the transmission power in each transmission to the quality of the link. This could be accomplished based on the RSSI of the last received packet from the destination node and would result in an increase on power efficiency.

- Consider new metrics other than the number of hops to chose routes. One metric could be, for instance, the cumulative RSSI of each link along the path.

- Combine several packets that arrive at an RN and retransmit a single aggregated packet, which contains those packets. As the final destination is common to all the uplink traffic (the gateway), packets with different origins can be combined in a single packet to reduce the packet overhead and consequently, increase the throughput.
Bibliography


Appendix A

Packet Air Times and Wait Times

### A.1 CR = 4/5, Explicit Header, BW = 125 kHz

#### A.1.1 Packet Air Times in seconds

<table>
<thead>
<tr>
<th>SF</th>
<th>T(8 bytes)</th>
<th>T(16 bytes)</th>
<th>T(32 bytes)</th>
<th>T(64 bytes)</th>
<th>T(128 bytes)</th>
<th>T(256 bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0.03610</td>
<td>0.05146</td>
<td>0.07194</td>
<td>0.11802</td>
<td>0.21530</td>
<td>0.39962</td>
</tr>
<tr>
<td>8</td>
<td>0.07219</td>
<td>0.09267</td>
<td>0.13363</td>
<td>0.21555</td>
<td>0.37939</td>
<td>0.70707</td>
</tr>
<tr>
<td>9</td>
<td>0.12390</td>
<td>0.16486</td>
<td>0.24678</td>
<td>0.39014</td>
<td>0.67686</td>
<td>1.25030</td>
</tr>
<tr>
<td>10</td>
<td>0.24781</td>
<td>0.32973</td>
<td>0.45261</td>
<td>0.69837</td>
<td>1.23085</td>
<td>2.29581</td>
</tr>
<tr>
<td>11</td>
<td>0.49562</td>
<td>0.65946</td>
<td>0.90522</td>
<td>1.39674</td>
<td>2.37978</td>
<td>4.34586</td>
</tr>
<tr>
<td>12</td>
<td>0.99123</td>
<td>1.15507</td>
<td>1.64659</td>
<td>2.62963</td>
<td>4.43187</td>
<td>8.03635</td>
</tr>
</tbody>
</table>

#### A.1.2 Packet Wait Times in seconds based on 1% duty cycle

<table>
<thead>
<tr>
<th>SF</th>
<th>T(8 bytes)</th>
<th>T(16 bytes)</th>
<th>T(32 bytes)</th>
<th>T(64 bytes)</th>
<th>T(128 bytes)</th>
<th>T(256 bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>3.57</td>
<td>5.09</td>
<td>7.12</td>
<td>11.68</td>
<td>21.31</td>
<td>39.56</td>
</tr>
<tr>
<td>8</td>
<td>7.15</td>
<td>9.17</td>
<td>13.23</td>
<td>21.34</td>
<td>37.56</td>
<td>70.00</td>
</tr>
<tr>
<td>9</td>
<td>12.27</td>
<td>16.32</td>
<td>24.43</td>
<td>38.62</td>
<td>67.01</td>
<td>123.78</td>
</tr>
<tr>
<td>10</td>
<td>24.53</td>
<td>32.64</td>
<td>44.81</td>
<td>69.14</td>
<td>121.85</td>
<td>227.28</td>
</tr>
<tr>
<td>11</td>
<td>49.07</td>
<td>65.29</td>
<td>89.62</td>
<td>138.28</td>
<td>235.60</td>
<td>430.24</td>
</tr>
<tr>
<td>12</td>
<td>98.13</td>
<td>114.35</td>
<td>163.01</td>
<td>260.33</td>
<td>438.76</td>
<td>795.60</td>
</tr>
</tbody>
</table>
A.2 CR = 4/5, Explicit Header, BW = 250 kHz

A.2.1 Packet Air Times in seconds

<table>
<thead>
<tr>
<th>SF</th>
<th>T(8bytes)</th>
<th>T(16bytes)</th>
<th>T(32bytes)</th>
<th>T(64bytes)</th>
<th>T(128bytes)</th>
<th>T(256bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0.01805</td>
<td>0.02573</td>
<td>0.03597</td>
<td>0.05901</td>
<td>0.10765</td>
<td>0.19981</td>
</tr>
<tr>
<td>8</td>
<td>0.03610</td>
<td>0.04634</td>
<td>0.06682</td>
<td>0.10778</td>
<td>0.18970</td>
<td>0.35354</td>
</tr>
<tr>
<td>9</td>
<td>0.06195</td>
<td>0.08243</td>
<td>0.12339</td>
<td>0.19507</td>
<td>0.33843</td>
<td>0.62515</td>
</tr>
<tr>
<td>10</td>
<td>0.12390</td>
<td>0.16486</td>
<td>0.22630</td>
<td>0.34918</td>
<td>0.61542</td>
<td>1.14790</td>
</tr>
<tr>
<td>11</td>
<td>0.24781</td>
<td>0.28877</td>
<td>0.41165</td>
<td>0.65741</td>
<td>1.14893</td>
<td>2.09101</td>
</tr>
<tr>
<td>12</td>
<td>0.49562</td>
<td>0.57754</td>
<td>0.82330</td>
<td>1.31482</td>
<td>2.21594</td>
<td>4.01818</td>
</tr>
</tbody>
</table>

A.2.2 Packet Wait Times in seconds based on 1% duty cycle

<table>
<thead>
<tr>
<th>SF</th>
<th>T(8bytes)</th>
<th>T(16bytes)</th>
<th>T(32bytes)</th>
<th>T(64bytes)</th>
<th>T(128bytes)</th>
<th>T(256bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>1.79</td>
<td>2.55</td>
<td>3.56</td>
<td>5.84</td>
<td>10.66</td>
<td>19.78</td>
</tr>
<tr>
<td>8</td>
<td>3.57</td>
<td>4.59</td>
<td>6.61</td>
<td>10.67</td>
<td>18.78</td>
<td>35.00</td>
</tr>
<tr>
<td>9</td>
<td>6.13</td>
<td>8.16</td>
<td>12.22</td>
<td>19.31</td>
<td>33.50</td>
<td>61.89</td>
</tr>
<tr>
<td>10</td>
<td>12.27</td>
<td>16.32</td>
<td>22.40</td>
<td>34.57</td>
<td>60.93</td>
<td>113.64</td>
</tr>
<tr>
<td>11</td>
<td>24.53</td>
<td>28.59</td>
<td>40.75</td>
<td>65.08</td>
<td>113.74</td>
<td>207.01</td>
</tr>
<tr>
<td>12</td>
<td>49.07</td>
<td>57.18</td>
<td>81.51</td>
<td>130.17</td>
<td>219.38</td>
<td>397.80</td>
</tr>
</tbody>
</table>
Appendix B

CAD RSSI Measurements

Figure B.1: Distribution of RSSI for a random transmission interval between transmissions with an upper bound of 8 and 16 ms. The presence of two clusters of RSSI values corresponds to the weak and strong transmissions.
Figure B.2: Distribution of RSSI for a random transmission interval between transmissions with an upper bound of 32 and 64 ms. The presence of two clusters of RSSI values corresponds to the weak and strong transmissions.

Figure B.3: Distribution of RSSI for a random transmission interval between transmissions with an upper bound of 128 ms. The presence of two clusters of RSSI values corresponds to the weak and strong transmissions.