Towards the efficient use of LoRa for Wireless Sensor Networks

Towards the efficient use of LoRa for Wireless Sensor Networks

THESIS

submitted in accordance with the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Martin Christiaan Bor, MSc
born in Nieuw-Beijerland, the Netherlands

School of Computing and Communications
Faculty of Science and Technology
Lancaster University
Lancaster, United Kingdom
https://www.lancaster.ac.uk/scc/

February 2020

This thesis was typeset using the XƎLaTeX typesetting system created by Jonathan Kew and the `memoir` class created by Peter R. Wilson. The body text is set with Minion Pro, designed by Robert Slimbach, which includes italics and small caps. Other fonts include Myriad Pro, designed by Christopher Slye and Fred Brady (originally designed by Robert Slimbach and Carol Twombly), Menlo, designed by Jim Lyles, and Math from Donald Knuth's Computer Modern family.

Printed in Lancaster, UK.

The paper used in this publication may meet the minimum requirements of British Standards Institution — Recommendations for the Presentation of Theses and Dissertations, BS 4821:1990.

`master@fed1874 ▪ 2020-02-28 00:12:39 +0000`

'Begin at the beginning,' the King said gravely, 'and go on till you come to the end: then stop.'

— Lewis Carroll, *Alice in Wonderland*

Declaration

This thesis is a presentation of my original research work. No part of this thesis has been submitted elsewhere for any other degree or qualification. All work is my own unless otherwise stated. The work was carried out under the guidance of Prof. Utz Roedig, at Lancaster University's School of Computing and Communications.

28th February 2020

Martin Christiaan Bor

Abstract

Since their inception in 1998 with the Smart Dust Project from University of Berkeley, Wireless Sensor Networks (WSNs) had a tremendous impact on both science and society, influencing many (new) research fields, like Cyber-physical System (CPS), Machine to Machine (M2M), and Internet of Things (IoT). In over two decades, WSN researchers have delivered a wide-range of hardware, communication protocols, operating systems, and applications, to deal with the now classic problems of resource-constrained devices, limited energy sources, and harsh communication environments.

However, WSN research happened mostly on the same kind of hardware. With wireless communication and embedded hardware evolving, there are new opportunities to resolve the long standing issues of scaling, deploying, and maintaining a WSN.

To this end, we explore in this work the most recent advances in low-power, long-range wireless communication, and the new challenges these new wireless communication techniques introduce. Specifically, we focus on the most promising such technology: LoRa.

LoRa is a novel low-power, long-range communication technology, which promises a single-hop network with millions of sensor nodes. Using practical experiments, we evaluate the unique properties of LoRa, like orthogonal spreading factors, non-destructive concurrent transmissions, and carrier activity detection. Utilising these

unique properties, we build a novel TDMA-style multi-hop Medium Access Control (MAC) protocol called *LoRaBlink*.

Based on empirical results, we develop a communication model and simulator called *LoRaSim* to explore the scalability of a LoRa network. We conclude that, in its current deployment, LoRa cannot support the scale it is envisioned to operate at.

One way to improve this scalability issue is Adaptive Data Rate (ADR). We develop two ADR protocols, *Probing* and *Optimistic Probing*, and compare them with the de facto standard ADR protocol used in the crowdsourced TTN LoRaWAN network. We demonstrate that our algorithms are much more responsive, energy efficient, and able to reach a more efficient configuration quicker, though reaching a suboptimal configuration for poor links, which is offset by the savings caused by the convergence speed.

Overall, this work provides theoretical and empirical proofs that LoRa can tackle some of the long standing problems within WSN. We envision that future work, in particular on ADR and MAC protocols for LoRa and other low-power, long-range communication technologies, will help push these new communication technologies to main-stream status in WSNs.

Contents

Acknowledgements

Writing a doctoral thesis is supposed to be a one-man project, but it cannot be done without the help from others. Therefore, I would like to express my gratitude to a number of people who have made all this possible.

First and foremost, I would like thank my supervisor, Utz Roedig, who inspired and encouraged me, over many discussions, white board sessions, and coffee breaks.

Furthermore, I would like to thank Ana Lucia Vârbănescu, John Vidler, and Tony Chung for their support and suggestions. Your help was invaluable.

Also, I would like to thank my friends and colleagues, for their chats, discussions, encouragements, lunch, dinner, and coffee breaks. In no particular order: Ethem, Hayat, Alex, Dominic, Jenny, Mehdi, Asad, Musaab, Ben, Chris, Karen, Steve, Sergio, Andy, and all the others I forgot to mention.

Lastly, but certainly not least, I would like to thank my parents and brother, for being understanding and helpful when I left the Netherlands, and visiting me frequently throughout the PhD.

Thank you.

Martin Bor

Lancaster, February 2020

Chapter 1

Introduction

The field of Wireless Sensor Networks (WSNs) has its origin in the Smart Dust Project [KKP99]. This project started in 1998 at the University of Berkeley, with the vision of building a network of tiny computers, each the size of a speck of dust, sensing their environment and communicating wirelessly. Although the hardware produced at the end of the project was non-functional, it set the stage for an entire new research field, and many new projects have grown out of it. The use of WSNs is demonstrated in a wide range of applications, including smart cities [Her+11], habitat monitoring [Mai+02], medical applications [Ott+05], and industrial process automation and control [Son+08].

A WSN is a network of spatially distributed autonomous sensor nodes, that monitor physical or environmental conditions. The sensors form an ad hoc network, by which sensor data is transported via other sensor nodes to a central location, often known as a sink or gateway. To enable high-volume deployment of WSNs, the research focus is on reducing cost and energy per sensor node, while simplifying development and maintenance. This has an effect on the used hardware, wireless communication bands, and energy source. Sensor nodes are built from inexpensive Commercial Off-The-Shelf (COTS) hardware. While cheap, they are severely resource-constrained, having processing power measured in megahertz, and memory and storage measured in kilobytes These limited resources further restrict the use of compute heavy algorithms, for data processing or security, as there is no space, time, or the energy to run them.

Sensor nodes often use Industrial, Science, and Medical (ISM) radio bands for communication. These radio bands are license-exempt, eliminating the need for an expensive license. In addition, this eases deployment in different regions, as regulations regarding ISM radio bands are often aligned across different regions. On the flip side, this makes the ISM radio band a popular wireless communication band, with users like Wi-Fi, Bluetooth, cordless phones, and Near Field Communication (NFC). Since radio bands are a shared medium, sensor nodes have to compete with all these other devices. Simply increasing transmit power to overpower any other transmitter is not possible, as it is too energy consuming. In addition, there are regulatory limitations, which only allow a fairly low power output.

To overcome these challenges, researchers have developed a wide range of communication protocols, algorithms, operating systems, and applications. They have shown how to deploy infrastructures, and how to pass data from the WSN to the Internet and *vice versa*.

This research mostly happened on the same kind of hardware, usually a Tmote Sky or equivalent. Embedded hardware has evolved, fuelled by the meteoric rise of smart phones and mobile computing, with their need for ever more powerful hardware, while still being able to operate on batteries for a full day or more. Mobile processors have seen the introduction of heterogeneous multicore processors, whereby powerful, but energy expensive processing units are combined with low-power processing units for background tasks, and specialised processing units for tasks like cryptography and machine learning.

Wireless communication has also become much faster and energy efficient. For long-range communication there are the evolving 3GPP standards 3G (UMTS, HSPA), 4G (LTE), and the upcoming 5G. For connecting to local networks there is Wi-Fi, and for short-range communication there is Bluetooth.

A recent development are new long-range, low-power communication protocols like LoRa and NB-IoT. Long-range communication opens up the possibility of making simple, single-hop networks, which have an infrastructure that is easier to manage, as there is no need for devices like cluster nodes or repeaters.

For the last decade, WSNs have relied on low-power short-range multi-hop communication protocols like Bluetooth and IEEE 802.15.4. While this led to some moderate success, questions have been raised about the scalability of this approach, especially in an industrial setting [Doh+08]. To deliver for larger IoT deployments, WSNs need to keep pace with modern technologies.

Therefore, in this work we focus on one of the most promising low-power long-range radio techniques: LoRa. LoRa is a novel wireless communication technology operating in a licence-exempt band, which makes it easy to deploy and experiment with, and further makes it both academically and industrially relevant. Early LoRa work has shown promise to resolve long standing issues, but it also introduced a new set of challenges. For example, long-range communication covers a much larger area, with a lot more nodes deployed. All these nodes, however, have to content for the same wireless medium. The question becomes how to scale this efficiently.

In this thesis we explore the recent advances in wireless communication technologies, which are of interest for WSNs (chapter 2). We further focus our attention on LoRa (chapter 3), a novel low-power long-range communication technology. We identify the most relevant problems this new technology can help resolve, and explore the scalability challenges it introduces (chapter 4). Finally, we provide one method to address the Long Range (LoRa) scalability challenge by dynamically tuning transmission parameters (chapter 5).

1.1 Research Questions

The advances in communication techniques, especially in long-range, low-power wireless communication, opens up new opportunities for WSNs. To better assess their overall usability, a thorough investigation is required to evaluate both their feasibility and the new challenges they introduce. To this end, the core research question addressed in this thesis is:

How feasible is LoRa for large-scale IoT deployments?

To provide a comprehensive answer to this question, we propose a research method that combines theoretical analysis with empirical research, and is driven by the following sub-questions:

R1) How do the advertised features of LoRa work in practice?
R2) Is it possible to build a multi-hop LoRa network protocol?
R3) How many devices can a LoRa network really support?
R4) How can LoRa transmissions be dynamically optimised?

Answering R3 revolves around creating representable models of a LoRa communication network, to allow large-scale simulation. Building such a simulator requires deep understanding of the physical layer (R1), and communication patterns of a LoRa network (R2). In turn, using the simulator, we uncover scalability issues, and propose an effective method to address them (R4).

1.2 Contributions

This thesis contributes to existing WSN research by providing methods, models, and experimental results for a new class of communication technology that changes long-held assumptions. Specifically, we make the following key contributions:

C1) An in-depth investigation of the physical layer of LoRa (chapter 3).

C2) Methods and tools for analysing the scalability of LoRa (chapter 4).

C3) A dynamic on-line ADR algorithm for optimising energy consumption (chapter 5).

1.3 Publications & Tools

The contributions described above are published in the following peer-reviewed papers.

Our initial exploration of LoRa (contribution C1), and the prototype of LoRaBlink, a novel multi-hop Medium Access Control (MAC) protocol, was published in *'LoRa for the Internet of Things'* [BVR16] and presented at MadCOM 2016 in Graz, Austria on Feb. 2016.

We investigated the scalability of LoRa (contribution C2) in detail in *'Do LoRa Low-Power Wide-Area Networks Scale?'* [Bor+16], which was presented at the MSWIM '16 held in Malta, Malta on Nov. 2016. The LoRaSim simulator developed for this paper has been published, with its source code, on the Lancaster University website with a permissive CC-BY license. This work has been widely cited ([ACP18; SPF18; RGS18; Zai+18; Hoe+18; CBR17; KIA17]), and many research groups have extended the simulator for their own explorations and analysis of LoRa and Long-Range Wide-Area Network (LoRaWAN) ([Ikh+18; CRO18; FP18; Pop+17]).

Our scalability analysis of LoRa revealed issues with the current setup and the ambition to have millions of devices. One approach we explored to mitigate these issues, was the use of directional antennae. This work was published in the paper

'*Mitigating Inter-Network Interference in LoRa Networks*' [Voi+17], and presented at MadCOM 2017 held in Uppsala, Sweden on Feb. 2017.

Another approach we explored was the use of transmission parameter selection (contribution C3). This work was published in '*LoRa Transmission Parameter Selection*' [BR17] and presented at DCOSS 2017 in Ottawa, Canada on June 2017.

1.4 Structure

The remainder of this thesis is structured as follows.

Chapter 2 presents the background and related work that sets the context for this thesis. We describe the WSN research field and related fields. We look into what makes a wireless sensor node, focusing on current hardware, and upcoming and future developments, that can change existing long-held assumptions for WSN.

In Chapter 3 we explore a radical new modulation technique called LoRa, which can alter the existing approach of building multi-hop sensor networks. With LoRa we can send data over kilometres instead meters, as was previously the case, for the same energy budget. This lets us build a single-hop sensor network, which is an attractive topology in terms of operation and maintenance. We investigate the unique properties of LoRa with practical experimentation, and discover its limitations. With the lessons learned, we develop a novel MAC protocol, to overcome some of LoRaWAN's limitations.

In Chapter 4 we investigate the scalability of LoRa. We develop a comprehensive link model of LoRa and a simulator to analyse its performance under various conditions. Our analysis shows that LoRa, as it is currently used LoRaWAN, does not scale well, and will not be able to support the millions of sensor nodes it claims to support. Our work proposes various ways these challenges can be mitigated.

One way we propose to increase the scalability of LoRa, is by dynamically adjusting the transmission parameters, which is explored in Chapter 5. We investigate the characteristics of a LoRa communication link more in-depth by experimentation, and show how the various parameters influence the communication performance. With the result of these experiments, we develop an online algorithm for optimising the transmission parameter configuration for LoRa.

Chapter 6 concludes this thesis with suggestions for future work.

Chapter 2

Background

This chapter describes the background and context of this thesis. It gives an introduction into WSNs, and the related research fields, and describes what characterises a WSN in terms of hardware, and network topology. It gives an overview of current and future technologies, with a focus on microcontrollers and radio communication hardware and protocols.

2.1 WSN, IoT, WoT, M2M, and CPS

The research field of WSNs was kicked off by the Smart Dust Project [KKP99], which had the vision of building a network of tiny computers, the size of a speck of dust, sensing their environment and communicating wirelessly. Although that vision only recently has come to materialise, the ability to have many cheap sensors collaboratively sensing the environment, has become are reality, and helped many other research fields, like biology [Mai+02], geology [Tal+07], and farming [LBV06].

The research of WSNs is a hybrid endeavour, taking elements from established research fields, such as distributed systems, ad hoc networking, embedded systems and wireless systems. These fields interact in complex ways when brought together, invalidating common assumptions. For example, most distributed systems research assume reliable communication (which is removed by wireless systems) and per-node resource limits similar to a desktop PC (which is not true in the embedded world, especially regarding energy limits). These disruptions of the common assumptions require new approaches to be created for problems that otherwise would be considered solved.

Over the years, new research fields closely related to WSNs emerged, either as direct offshoots, or research in other areas that became relevant to WSNs (*i.e.* cognitive radio, ad hoc wireless network). The most common related research fields are: Internet of Things (IoT), Web of Things (WoT), Machine to Machine (M2M), and Cyber-physical System (CPS). These fields all focus on a different part of the system. Sometimes, the difference between these fields is hard to distinguish, and research on a CPS may be published as research on IoT.

The following sections give a short overview for each of these related research fields.

2.1.1 IoT

Internet of Things (IoT) is an offshoot of WSNs. Where the field of WSNs mostly deals with interconnecting sensors, IoT deals with interconnecting devices (*things*) via the Internet [Ger99]. It does not only deal with sensors, but also other physical devices, like vehicles, home appliances, and actuators. IoT combines the work of embedded systems, WSN, and control systems into one field. Typical applications for IoT is smart home, transport, energy monitoring and elderly care. An IoT sensor usually directly sends its data to a central server ('in the cloud') via the Internet, where algorithms are used to analyse, decide on, and instruct other connected devices to operate. Since there is much overlap between WSN and IoT, we use the two terms interchangeably.

2.1.2 WoT

Web of Things (WoT) [Kin+00; GT16] is closely related to IoT, and focuses more on the upper layers of the network stack. Rather than reinventing new standards, WoT encourages the reuse of existing well-known Web standards, like REST, HTTP, JSON, Microdata, and WebSockets. WoT tries to simplify the creation of IoT applications.

2.1.3 M2M

Machine to Machine (M2M) is the direct communication between devices, either wired or wirelessly. This field exists since the advent of computer networking in the early 20th century. With the advancements in wireless communication, the techniques and research in this area are used in WSN and *vice versa*. The research work performed in WSN can also apply to this area, as there is quite an overlap.

2.1.4 CPS

A Cyber-physical System (CPS) is a mechanism that is controlled or monitored by computer-based algorithms, tightly integrated with the Internet and its users [09]. Examples of CPS include smart grid, autonomous automotive systems, medical monitoring, process control systems, robotics, and automatic pilot avionics. CPS is similar to IoT, sharing the same basic architecture. The difference is that CPS presents a tighter combination and coordination between physical and computational elements.

2.2 Characteristics of a WSN

This section introduces common characteristics and concepts in WSN. We first have a look at the general architecture of a *wireless sensor node*. After this we look into the various models of communication.

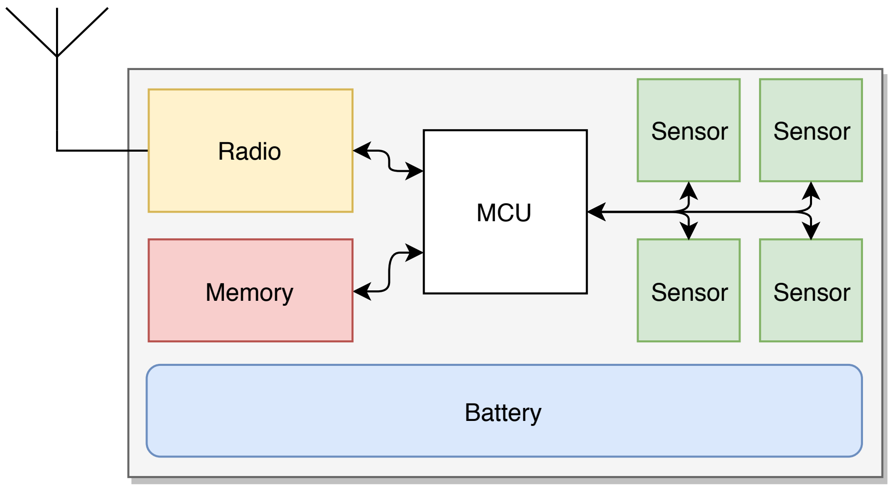


Figure 2.1: Typical architecture of a wireless sensor node.

2.2.1 Wireless Sensor Node

A typical architecture of a wireless sensor node is shown in fig. 2.1. A sensor node consists of a Microcontroller Unit (MCU), a radio (for wireless communication), a battery (or other power sources, like a solar panel, or other energy harvesting methods like temperature, vibration), external memory for permanent storage (that can be part of the MCU), and one or more sensors, typically connected via some bus (I^2C, SPI, GPIO).

The main design criteria for a wireless sensor node is that it should operate unattended, with some deployments requiring a lifetime of up to 10 years. This minimises maintenance cost, as no battery replacements are required during the lifetime of a sensor node. Unattended long operating time also allows sensor nodes to be placed in inhospitable locations, like volcanoes or remote islands.

There are two strategies to achieve this: using low-power components and minimising operating times. Low-power components minimise the energy used while a sensor nodes operates. This usually means the components are resource constrained, in terms of processing power, storage, communicate speed and range. Simpler components require much less energy to operate. As a consequence this means that complex processing is often not possible. Similarly sending large amounts of data is not possible, as that would quickly deplete the battery. Therefore most of the processing would be used to aggregate the data, and only send a message with a summary, a delta of the last measurement, or only send a message when something 'interesting' happens.

The other strategy to reduce energy consumption is by minimising operating time. A sensor node will be in sleep (low-power) mode the vast majority of the time (duty cycles of 0.1% or less are common), only to wake up to take a measurement and perhaps send it to the data collection end.

The other main design criteria is wireless sensors nodes should should be cheap, so many nodes can be deployed, improving resilience via redundancy. To keep the costs down, sensor nodes are made from COTS components. Custom made components may be more power efficient, and more accurate and precise, but costs significantly more. Inexpensive components introduces new challenges. For example inexpensive clock crystals, used for time keeping, are not very accurate, causing *clock drift*. A clock crystal often used in sensor nodes has an accuracy of 100 ppm, which gives an error of 8.64 s per day. Sensor nodes sleep most of the time, and only wake up on a regular interval for a short period to receive or transmit data. Transmitting a message at the right time, when time deviates so much, is challenging. More accurate crystals are available, but cost significantly more and may require more energy, for example when using a temperature compensated crystal oscillator (TCXO).

In section 2.3 we give a more detailed description of the current and future hardware of MCUs. Section 2.4 gives a detailed overview of current and future radios.

2.2.2 Communication Topologies

Instead of the arbitrary point-to-point networking standard in distributed systems, most WSNs are organised in a *source-to-sink* model of networking. This model of communication follows from the typical application of WSNs, namely gathering data. Nodes that have data (*sources*) do not have the energy or capacity to transmit very far, so external help is required to get the data to the end user. This is often achieved by having a *sink* node, sometimes called a *gateway*, *bridge* or *base station*. The sink node is a node with more capabilities: bigger batteries (or mains powered), increased processing capabilities, and often connected to a second, larger network (*e.g.* a wired network connection to the Internet). This topology, whereby a source node sends data to a sink node directly, is called a *star network*, as shown in fig. 2.2a.

Often a node is not in direct range of a sink node, and it needs help from other nodes to get its data to the sink. The common strategy is to send the data to a node that is (physically) closer to the sink node. This node can then forward the data via other nodes to get it eventually to the sink. Every transfer is called a *hop*, and often the distance of a node to a sink is measured in hops (and not in meters).

Considerable research effort is spent on making this process as efficient as possible. For one, the main strategy for extending lifetime is to just be asleep most of the time. Ideally a node should only wake up to take a reading from the sensors, and send the data towards the sink, and go back to sleep. With a forwarding network, the intended recipient (that has the same energy conservation strategy), needs to be awake at exactly the same moment and have its radio switched into receive mode (as the radios used are half duplex) to get the data. Otherwise the data is lost. As not to have every node wake up every time, a topology is required. The most common topology is a *tree network*, or *collection tree*, as shown in fig. 2.2b. This network establishes a relationship between nodes, whereby nodes are parents and children. Nodes without children are called *edge nodes*, or *leaf nodes*.

A collection tree requires intermediate nodes to be awake more often than leaf nodes, which can be quite a strain on the battery. Especially nodes closest to the sink node, which see a considerable amount of traffic, exhaust their battery much faster.

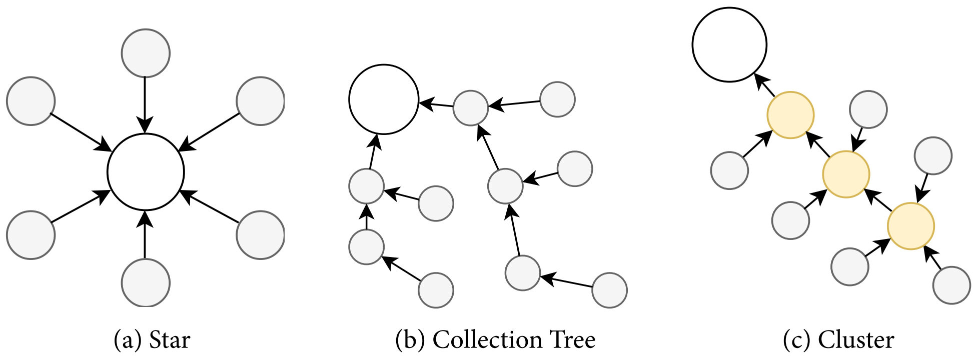


Figure 2.2: Common WSN communication topologies. Gray nodes are source nodes, yellow nodes are cluster nodes, white nodes are sinks.

This can lead to a cascading collapse of the network, as nodes closest to the base station will run out of energy first. Following the next closest nodes, which may need to exert even more effort to reach the sink node.

Another approach is the *cluster network*, as shown in fig. 2.2c. This particular network has designated *cluster nodes*, usually equipped with bigger batteries. This solves the issues of low power nodes having to be awake to forward data. Cluster networks are less flexible, as cluster nodes need to be placed in strategic positions so to cover the whole network. The collection tree can be dynamically reconfigured at runtime, as every node could take the role of parent of leaf node. With a cluster network this is harder.

2.3 MCUs

In this section we focus on the microcontroller often used in WSN, and new developments that can be of use in WSN. We give an overview of the current hardware, and an indication of interesting upcoming developments.

2.3.1 Common MCUs

A WSN node's main processing unit is the MCU. A MCU in itself is a tiny computer. It has a compute unit, internal RAM, often a ROM in the form of flash memory to store executable code, and a static memory in the form of an EEPROM for storing persistent data. The peripherals included on a MCU are timers, watchdogs, power control, and peripherals for interconnecting with other devices like sensors via UART, I^2C, SPI, and other communication protocols.

Often MCUs have an 8-bit or 16-bit computing architecture, which is quite a throwback from the 64-bit computing architecture that is common in current desktop class Central Processing Units (CPUs) and even smart phone CPUs. The reason for using an 8-bit or 16-bit computing architecture is the simpler architecture, which consumes less energy, and the lower price. Also, code density is much higher (more instructions per byte), requiring less storage.

Various companies make MCUs, each having a different architecture and instruction set. The most commonly used 8-bit and 16-bit MCUs are the Atmel (now Microchip) AVR [Mic] series and the TI MSP430 [Tex] series. The Atmel AVR is an 8-bit modified Harvard (instructions and data in separate memory systems) Reduced Instruction Set Computer (RISC) architecture. A commonly used Atmel AVR MCU is the ATmega1281, which runs at maximum of 16 MHz, has 128 kB flash, and 8 kB RAM. It is used in platforms like the Crossbow Mica2, Libelium Waspmote [Lib] and the CSIRO Fleck series [Sik+07]. The TI MSP430 is a 16-bit Von Neumann (instructions and data in same memory system) RISC architecture. It is used in platforms like the Telos [PSC05], and Zolertia Z1 [Zol].

Though Atmel AVR and TI MSP430 MCUs series are very popular, the 32-bit ARM Cortex-M series have slowly been chipping away at the 8-bit market as the prices of low-end Cortex-M chips have come down. Cortex-M have become a popular replacements for 8-bit chips in applications that benefit from 32-bit math operations, and replacing older legacy ARM cores such as ARM7 and ARM9. Although a 32-bit computing architecture would have a lower code density, the Cortex-M supports the Thumb instruction set, a variable-length instruction set providing both 32-bit and 16-bit instruction sets, making code density on par with comparable 16-bit MCUs [GS96; Joh14].

2.3.2 Multicore MCUs

The MCUs described in the previous sections have a single unit of computing, often called a *core*. In desktop computing, and nowadays also in smart phones, *multi-core* CPUs are widespread. Initially it was bit of a marketing gimmick, as CPU designers were hitting the physical limits of increasing the clock speeds. For a multi-core processor to be effective, the workload should be parallelisable. That is, the workload should be divisible in tasks that can run independently from each other. This may not always be the case, as some computation needs to happen in series.

A different strategy, which is mostly used in the realm of smart phones, is using an asymmetric (heterogeneous) design. Instead of having a couple of homogeneous cores, the MCU consists of a couple of high-power, high-performance cores for compute intensive tasks, and a couple of low-power, low-performance core, for sensing and background tasks. Take for example the Apple A9, which has a dual core ARMv8E, and an Apple M9 motion coprocessor [SH16]. The motion coprocessor is used to collect, store, and process sensor data when the device is asleep. It is also used to detect the keyword for activating the voice assistant (Siri). This reduces the power consumption considerably. Other examples are ARM big.LITTLE [ARM], the Parallax P8X32A Propellor [Par], and the xMOS xCORE-200 [18].

Instead of using general purpose compute cores, some MCUs are using specialised cores for particular tasks. The TI CC2538, for example, has an ARM Cortex-M3 compute core and a 2.4 GHz radio, which has its own *command-strobe processor*, used for handling packet data and MAC related tasks without using the main compute core. Similarly the TI AM3358 System on Chip (SoC) as used on the BeagleBones [Bea] has two Programmable Real-time Units (PRUs): a 32-bit 200 MHz real-time core with 8 kB of program memory. These PRUs have direct access to general I/O and are used for real-time control and communication protocols.

Specialised Companion Chips

A MCU is often made for general purpose computing. Depending on the use case, it may be beneficial to use highly specialised chips, which can perform computation for a specific set of tasks, vastly quicker and energy efficient than a MCU would be capable of.

An example of a specialised chip is a Digital Signal Processor (DSP), which is used for processing audio and video signals. Many DSP applications have constraints on latency, and operations must be completed within a fixed time, and deferred processing is not viable. An MCU would be too slow, or too energy inefficient to meet these requirements.

Other examples of specialised chips are Field-Programmable Gate Arrays (FPGAs). FPGAs contain an array of programmable logic blocks, with reconfigurable interconnects that connects the blocks together, effectively building a custom accelerator chip. Having a FPGA enables the user to modify the integrated circuit after deployment (hence the term 'field-programmable'). With a sensor node being deployed for up to a decade, or even more, this is an appealing idea. As it is hard to design a sensor node for the next ten years, an FPGA allows a node to be adapted, or even repurposed later in its life, when the application requirement changes. FPGAs chips, however, require significant board space, and quite some energy when reprogramming, making their applicability to WSN limited at this time.

With popularity of artificial intelligence, and more specifically the emergence of deep learning, specialised processors are developed to accelerate AI tasks. Like with DSPs, these chips allow the use of neural networks in a much more energy-efficient way than a MCU would be capable of. Recent processor for smart phones, like the Apple A12 Bionic and Qualcom Snapdragon 845 include dedicated *neural network hardware*. With advances in manufacturing, and growing popularity, it is expected these accelerators will become cheaper and more energy efficient, making them suitable for application in WSNs.

2.4 Radios

A major part of a WSN node (and no surprise given its name), is the radio transceiver, or *radio* for short. The radio consumes a vast majority of the energy budget, so operating it efficiently is critical. The choice of radio defines the technology, medium, and RF band that can be used, which in turn determines the performance and capability of a sensor node.

In this section we focus on radios often used in WSNs, and new developments that can be of use in WSNs. We first give an overview of the potential choices of RF bands for a WSN, and the regulatory constraints that limits what we (legally) can and cannot do. After this we look at current and upcoming technologies and associated protocols in low-power, low-data-rate wireless communication, starting from the short range (tens to hundredths of meters), to the long range (tens to hundredths of kilometres).

2.4.1 2.4 GHz vs sub-1 GHz

Typically, the higher the transmit frequency, the smaller the antenna required, which is good for miniaturisation, and the higher potential data rate. This means that nodes can send packets quicker, and go to sleep sooner, preserving energy and vastly improving operating life time. On the other hand, higher frequency are more easily attenuated by materials (concrete, wood, people, *etc.*), which limits its range. Since transmit power vs range is an quadratic relationship (in the simplest form), if you want to send twice as far, you need four times the energy. The savings in reducing transmission time from the increased data rate can quickly outpace the extra power required to actually reach the other node.

As one can expect, a lower transmission frequency works in the opposite direction. Lower frequencies require larger antennas, and usually have a lower data rate. On the other hand, lower frequencies have a much longer transmission range, typically measured in kilometres instead of hundredths of meters. Therefore picking the right frequency is a delicate balance, between range and energy consumption.

WSN nodes usually operate in an ISM band. These bands were initially not reserved for telecommunication, but for industrial, scientific, and medical purposes. These bands are license-exempt, and do not require an expensive license like other bands. Therefore they have become increasingly more popular for wireless communication.

The most popular *high frequency* band is the 2.4 GHz band. It is widely supported, and it is an open band worldwide (see section 2.4.2 for details on regulatory constraints). With this wide spread support, it is also a very popular and crowded band, with users like Wi-Fi, Bluetooth, and others. With this overcrowding, the 5 GHz band has become quite popular in recent years, especially for Wi-Fi. Its use is however somewhat limited, as the range is quite a lot less (typically around 20 m) compared to 2.4 GHz (typically around 50 m indoors). Also, since this band is sometimes used by military radar systems, devices often need to implement band scanning facilities to ensure they do not interfere.

The *low frequency* sub-1 GHz band is a lot harder to use worldwide, compared to the 2.4 GHz band. For example, the 868 MHz band is usable in Europe, but not in the US, where the 915 MHz band would be a good equivalent. Other bands like the 433 MHz band have better worldwide support, but like the 2.4 GHz band are over-crowded, and require larger antennas. Therefore the most useful sub-1 GHz bands for WSNs are 868 MHz in Europe and 915 MHz in the US, providing a good balance in usability, range, and antenna size.

2.4.2 Regulatory Constraints

Beside the physical limitations, there are also regulatory constraints on which frequencies, transmit powers and duty cycles can be used. These constraints are often defined for a particular region, and sometimes even for a particular country. This makes it a complex subject matter. Here we describe the regulations for the most commonly used bands: 2.4 GHz and sub-1 GHz band.

Constraints are defined by nations, regulatory bodies like the International Telecommunication Union (ITU), and the European Telecommunications Standards Institute (ETSI), which gets harmonised by local regulations. ETSI covers Europe, but other countries like Australia, Canada, and China, are also part of ETSI.

WSNs are often classified as a Short Range Device (SRD) and operate on license-exempt frequency bands. There are certain restrictions on access to the physical medium, imposed by the regulatory body for the particular region, which has an impact on communication performance. Wireless communication performance is thereby often limited due to regulatory constraints, and not due to technical limitations. Next we describe in more detail EU and US regulations; other countries such as China have their own regulations with often are modelled on EU or US standards.

2.4 GHz Band

The 2.4 GHz band is a worldwide unlicensed band. Therefore the regulations for its use are harmonised around the world. There are no limitations in terms of duty cycle, and channel bandwidth, or requirements like Listen Before Talk (LBT). The main limitation is that Effective Isotropic Radiated Power (EIRP) should not exceed 100 mW (20 dBm).

sub-1 GHz Band

Unlike the 2.4 GHz band, the sub-1 GHz is not a worldwide unlicensed band. The regulations on which frequency bands can be used, vary per region. In this section we describe the regulations for Europe and the United States.

Europe The constraints in Europe regarding frequency allocation and use for SRD are defined in CEPT/ERC/REC 70–03 [15]. The license-exempt band usable for WSN (863 MHz to 870 MHz) is referred to as 'Annex 1 h', and is subdivided in 7 (overlapping) subbands. Each subband has specific requirements regarding maximum EIRP, spectrum access, and channel spacing. For the majority of the subbands, the EIRP is 25 mW (14 dBm). For spectrum access there is the option of either using a duty cycle (often $\leq 0.1\%$), or a LBT transmission scheme, combined with Adaptive Frequency Agility (AFA). This is dependant on the specific subband and/or EIRP required (see [12, chapter 9] for details).

United States The Federal Communications Commission (FCC) regulates the use of frequencies for wireless communications in United States. Rules and regulations are stated in Title 47 of the Code of Federal Regulations (CFR). Part 15 (often referred to as 'FCC Rule 15') of this code deals with devices operating in unlicensed frequency bands. The license-exempt band usable for WSN is 902 MHz to 928 MHz, often referred to as the 915 MHz band.

Compared to the European regulations, the FCC allows a higher peak output power of 1 W (30 dBm), but requires a bandwidth of at least 500 kHz. For lower bandwidths, the device operates in 'hybrid mode', which combines the regulations for digital modulation techniques (like LoRa, see section 2.4.5 for more details) with those

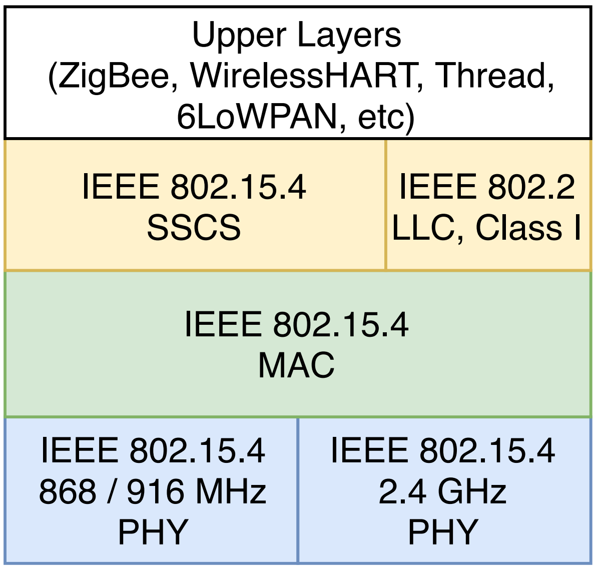

Upper Layers
(ZigBee, WirelessHART, Thread, 6LoWPAN, etc)
IEEE 802.15.4 SSCS
IEEE 802.2 LLC, Class I
IEEE 802.15.4 MAC
IEEE 802.15.4 868 / 916 MHz PHY
IEEE 802.15.4 2.4 GHz PHY

Figure 2.3: IEEE 802.15.4 protocol stack.

for Frequency Hopping Spread Spectrum (FHSS). An important limitation for FHSS systems, is the maximum dwell time of 400 ms. This limits its use for some of the lower data rate radio modulations, like LoRa, as they can take more than 400 ms to transmit a symbol.

2.4.3 Low Rate Wireless Personal Area Networks

Low-Rate Wireless Personal Area Networks (LR-WPANs) are a class of networks for interconnecting devices centred around an individual person's workspace. These networks often have a transmission range of 10 m to 100 m, and a transmission range of 10 kbit/s to 250 kbit/s.

IEEE 802.15.4

IEEE 802.15.4 [16a] is a technical standard for LR-WPANs, defining the physical and MAC layer. It defines the modulation scheme, wireless spectrum, and MAC algorithms. An overview of the protocol stack is show in fig. 2.3.

IEEE 802.15.4 physical layer supports three unlicensed bands: 868 MHz in Europe, 915 MHz in North America, and 2.4 GHz worldwide. Data rates are from 20 kbit/s to 250 kbit/s and the transmission range of a device varies between 10 m to 100 m. The original 2003 version only defined Direct-Sequence Spread Spectrum (DSSS) as the modulation scheme, with either Binary Phase-Shift Keying (BPSK) for the 868 MHz and 915 MHz band, and Offset-Quadrature Phase-Shift Keying (OQPSK) for the 2.4 GHz band. Later revisions added Parallel-Sequence Spread Spectrum (PSSS), Gaussian Frequency-Shift Keying (GFSK), Direct-Sequence Ultra Wide Band (DS-UWB), and Chirp Spread Spectrum (CSS).

The MAC layer of IEEE 802.15.4 offers the data services (transmission of MAC frames) and also manages functions that controls the management access to physical channel and network beaconing. The MAC layer can operate in a slotted Carrier-sense Multiple Access with Collision Avoidance (CSMA/CA), whereby the network is organised by a Coordinator node, transmitting beacons on a regular interval. The beacons (*superframes* in IEEE 802.15.4) announce the length of the contention period, whereby any device can send (via CSMA/CA), and the length of the contention-free period, whereby a device can send in its assigned slot. Devices need to wake up regularly to receive the beacon, which may not be as energy-efficient. The use of beaconing mode does guarantee the latency and delivery of packets. As an alternative, the MAC layer can also operate in unslotted CSMA/CA, whereby devices can follow their own sleep schedule. This mode does not need a Coordinator node, and no beacons are transmitted. This mode can be more energy-efficient, but has no bounds on latency or delivery.

The higher layers of IEEE 802.15.4 are not defined, and other technologies like Zigbee, ISA100.11A, WirelessHART, MiWi, Snap, Thread, and 6LoWPAN extend the standard based on their application requirements.

Proprietary, Non-Standard, and Custom Protocols

Although using standards like IEEE 802.15.4 helps with interoperability, some implementers explicitly choose to implement their own proprietary protocol. The proprietary protocols are tailored for their specific application, stripping out all the unnecessary features of other protocols, to minimise overhead, maximise performance, and energy efficiency. These protocols often use modulations like Amplitude-Shift Keying (ASK), On-off Keying (OOK), or GFSK. These modulation schemes are not as advanced as the ones employed by IEEE 802.15.4, but are often much cheaper in terms of hardware. For example a TI CC1201 (2/4-(G)FSK/MSK/OOK) costs around $1.90 (when buying 1000 units), while a TI CC2520 (Zigbee/802.15.4 transceiver) costs around $2.48 (when buying 1000 units).

Bluetooth

Bluetooth is a popular, short range wireless communication protocol for interconnecting accessories to a personal device. With the introduction of Bluetooth Low Energy (BLE) it became much more energy-efficient, although it uses a completely different, incompatible modulation scheme compared to traditional Bluetooth.

Bluetooth uses FHSS, to counteract narrowband interference problems. The spectrum is divided into 79 channels, each with a 1 MHz bandwidth. While transmitting, it hops between these channels, up to 1600 hops per second. Originally, Bluetooth used GFSK, but later revisions added $\pi/4$-Differential Quadrature Phase Shift Keying (DPSK) and 8DPSK. Bluetooth is packet-based protocol, in a master-slave architecture. One master communications with up to seven slaves in a *piconet*. Piconets can be interconnected into a *scatternet* when a device (either a master or slave) elects to participate as a slave in a different piconet. Scatternets are not part of the Bluetooth standard, and is an area of active research.

2.4.4 Wireless LAN

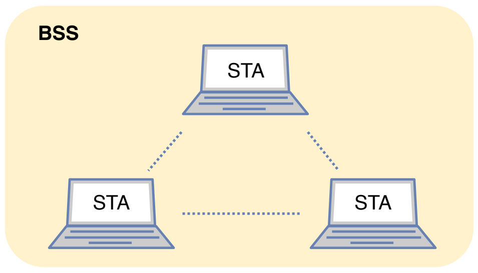
Wireless Local Area Network (WLAN) is a class of networks that links two or more devices in a communication network. It differs from LR-WPAN in that the range and data rate are often much higher. Most of the modern WLANs are based on IEEE 802.11 standards, which is marketed under the Wi-Fi brand name.

Wi-Fi

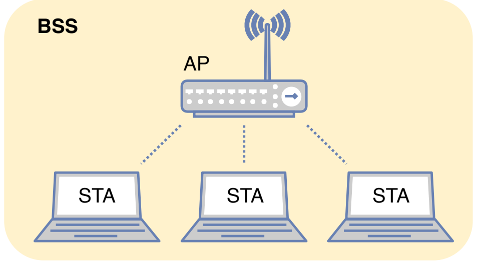
IEEE 802.11, or Wi-Fi as it is better known, was first released in 1997. It is designed to be as transparent as possible, similar to Ethernet for the upper layers, only replacing the lower layers of the OSI model. It is the standard for wireless communication on computers. Over the years, many amendments have been made to the standard, mainly focusing on improving throughput and security, but not so much on improving energy efficiency. Therefore Wi-Fi was often dismissed for IoT applications, as it is too power hungry, even though (ironically) Wi-Fi was initially designed for battery operated devices. Recent advances, both in hardware and amendments to the standard, have improved this situation. Using Wi-Fi for IoT is an attractive option, as the communication infrastructure, especially within buildings, often already exists.

Network Architecture In Wi-Fi, a network consists of nodes, often referred to as stations (STAs). Each network consists of a Basic Service Set (BSS), an area with the same medium access characteristics (*i.e.* radio frequency, modulation scheme), in which STAs can communicate. An Independent Basic Service Set (IBSS), shown in fig. 2.4a, is the the simplest form of a BSS, consisting of two (or more) STAs that communicate directly with each other (*peer-to-peer*). This is often referred to as an *ad-hoc network*.

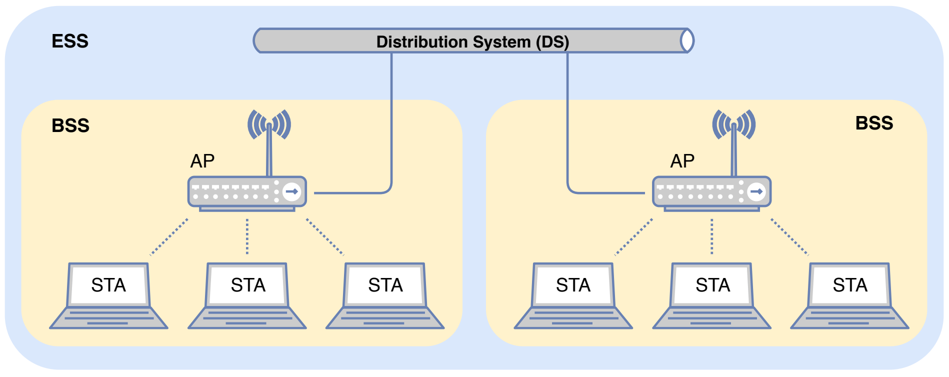
A more common architecture is Infrastructure BSS, show in fig. 2.4b. In infrastructure mode, STAs only communicates via a redistribution point, like an access point (AP) or a mesh node. In Infrastructure-BSS, an AP periodically broadcasts a



(a) Independent BSS



(b) Infrastructure BSS



(c) Extended Service Set

Figure 2.4: Common Wi-Fi communication architectures.

beacon to announce its presence to STAs within the BSS. A STA connects (*associates*) to this AP using the information from the beacon.

Multiple BSSs can be combined into an Extended Service Set (ESS) using a Distribution System (DS) (like Ethernet) to form a logical network segment. Figure 2.4c shows the architecture of an ESS. This architecture allows a STA to move transparently from one participating BSS to another BSS, within the same ESS. An ESS makes a couple of distribution services possible, like centralised authentication (as opposed to have each AP provide this service), seamless roaming for STAs between BSSs, and steering STAs to connect to particular APs that can provide a better Quality of Service (QoS).

Versions Over the years, many versions of Wi-Fi have been developed, which has been published as amendments to the original standard from 1997.

IEEE 802.11b, released in 1999, was the first widely adopted Wi-Fi standard. Most current Wi-Fi devices are backward compatible with this version. In IEEE 802.11b, the channel bandwidth is 22 MHz, and the minimum transmission power is 0 dBm. This standard introduced DSSS, which increased the maximum data rate from 2 Mbit/s from the original 1997 standard to 11 Mbit/s.

IEEE 802.11g was released in 2003, and introduced Orthogonal Frequency-Division Multiplexing (OFDM) modulation, increasing the data rate to 54 Mbit/s. In deployments where a 802.11g network must coexist with 802.11b devices, messages are transmitted with an 802.11b-compatible DSSS packet header, which reduces the maximum data rate. This version also introduced extensive security features, such as Wi-Fi Protected Access (WPA), to replace the broken Wireless Equivalent Privacy (WEP).

IEEE 802.11n, published in 2009, was the next major evolution of Wi-Fi. It significantly increased the bandwidth from 54 Mbit/s to 600 Mbit/s, by adding support for Multiple-Input Multiple-Output (MIMO), 40 MHz channels, and frame aggregation. IEEE 802.11n can also be used in the 5 GHz band, helping speed and connectivity in dense deployments.

One of the most recent amendments is IEEE 802.11ah, also known as *Wi-Fi HaLow*. It uses the sub-1 GHz band (868 MHz in Europe, 915 MHz in USA), as opposed to the more conventional 2.4 GHz and 5 GHz band. IEEE 802.11ah offers an extended range, with features to minimise contention via sectorisation and restricted window access, and improving energy consumption by target wake time. The protocol is designed to support IoT applications like smart metering, supporting data rates between 0.3 Mbit/s to 347 Mbit/s. While the amendment was published in 2017, at the time of writing, Wi-Fi HaLow chipsets are still in development, and no commercial chipsets are available.

2.4.5 Low Power Wide Area Networks

Low-Power Wide Area Network (LPWAN) is a type of wireless communication wide-area network designed to allow a long range communication at a low bit rate, for devices operated on a limited energy budget. As opposed to regular LR-WPAN technologies that have ranges up to 100 m, a Low-Power Wide Area Network (LPWAN) typically have ranges measured in kilometres. The various LPWAN technologies fall broadly into two categories: ultra-narrow band and wide band.

Ultra-narrow band tries to pierce through the noise, by concentrating all its energy on a very narrow band. This works well in noisy environments, but when there is local interference, this does not work so well any more. Wide band tries the opposite approach, spreading its energy over a wide band, whereby the signal can be retrieved even from below the noise floor. Sigfox and LoRa are two the most popular LPWAN technologies, and discussed in more detail in the following sections.

Sigfox

Sigfox is a French company that builds a wireless network based on a proprietary ultra-narrow band technology. When Sigfox started in 2009, it was one of the first companies to set up a (commercial) LPWAN network targeted at inexpensive, low-power sensor nodes. The Sigfox protocol operates in the sub-1 GHz ISM band, using the 868 MHz band in Europe and the 915 MHz band in the USA. The protocol supports up to 140 uplink messages a day, whereby each message can carry 12 B of payload, up to 4 downlink messages per day, whereby each message can carry 8 B of payload. Messages are sent at a bitrate between 100 bit/s and 600 bit/s.

LoRa

LoRa is a proprietary spread-spectrum wide band modulation technology, developed by Cycléo SAS, and acquired by Semtech in 2012 [Sem12]. LoRa is a derivative of

CSS with integrated Forward Error Correction (FEC). The wide band transmissions counter interference and handle frequency offsets caused by low cost crystals. A LoRa transceiver can decode transmissions 19.5 dB below the noise floor, enabling communication over very long distances, typically 10 km or more in rural areas, for a relatively small energy budget. LoRa key properties are: long range, high robustness, multipath resistance, Doppler resistance, and low power. LoRa transceivers available today can operate between 137 MHz to 1020 MHz, and thus can also operate in licensed bands. However, they are often deployed in ISM bands (EU: 868 MHz and 433 MHz, USA: 915 MHz and 433 MHz).

The LoRa physical layer may be used with any MAC layer. For example, Aerts has ported ContikiMAC to LoRa [Aer16], and DASH7 could be used with LoRa [Nor15]. However, LoRaWAN is most common MAC for LoRa. An alternative is Symphony Link by Links Labs, which is mostly targeted at the North American market.

LoRaWAN and Symphony Link are discussed in more detail in the following sections. The LoRa modulation scheme is discussed in more detail in chapter 3.

LoRaWAN

LoRaWAN is a MAC and a network layer protocol for managing communication between LPWAN gateways and end-node devices. It is the de facto standard for LoRa networks. The LoRaWAN specification is maintained by the not-for-profit LoRa Alliance, who also offer a certification program to guarantee interoperability.

The topology of a LoRaWAN network is a one-hop star network. The network consists of three basic network elements, as show in fig. 2.5:

End-device

A device deployed by the end-user, typically with a sensor that sends its data to one or more gateways.

Gateway

A powerful device that is capable of receiving and decoding multiple concurrent transmissions. It forwards the data received from an end-device to the Network Server. Gateways are operated by network operators.

Network Server

The Network Server de-duplicates and verifies a received packet. Based on the headers it forwards the data to the appropriate Application Server.

Application Server

The server operated by the end-user that processes the data.

The topology of LoRaWAN follows the model of a typical cellular network operator, whereby the end-user are in control of the mobile phones and web server (end-device and application server), while the towers and infrastructure are operated and maintained by third parties (telecommunication operators).

A LoRaWAN topology is strongly oriented towards an upstream message flow, that is: messages go from end-devices to application server. For messages being sent downstream, from the application server to the end-device, the network capacity is much less. For example, a gateway cannot receive data while it is transmitting data. Acknowledgements are therefore disabled by default, and its use is discouraged.

Although the current standard does provide space for a repeater, by limiting the packet size when a repeater is involved, only one repeater is allowed, and it is not specified how this repeater should be implemented.

The LoRaWAN specification defines three distinct classes of devices: (i) Class A (ii) Class B, and (iii) Class C. A LoRaWAN device always starts as a Class A device, but can later switch to another class if required. Class B and Class C, however, are mutually exclusive.

Class A is mandatory for every LoRaWAN device. This is the most energy efficient communication class, and suitable for battery powered sensors and actuator with no

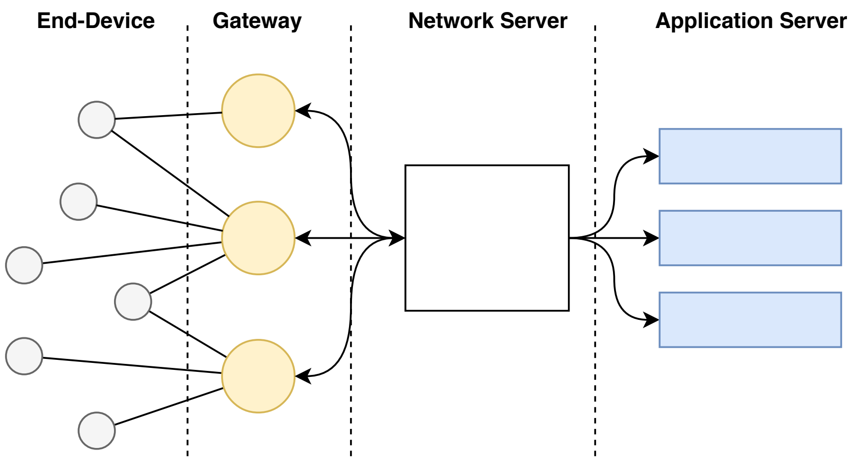
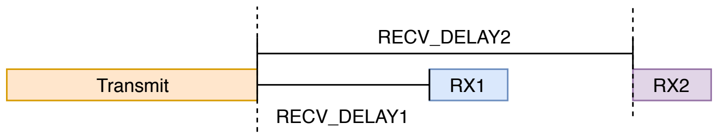


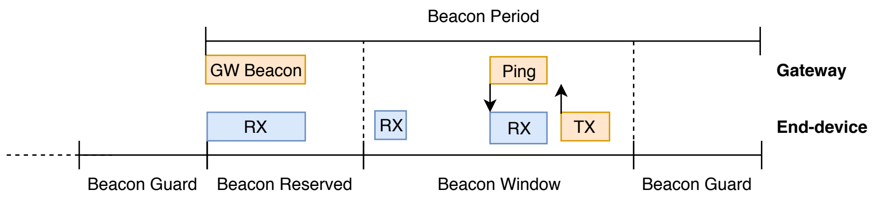
Figure 2.5: LoRaWAN network architecture.

latency constraints. In Class A, a device always initiates the communication. When a device transmits a message, it opens two receive windows after specific delays. Timings of these delays and the lengths of the receive windows are subject to regional constraints. A timing diagram is shown in fig. 2.6a. The first receive window will listen on the same configuration as is used for transmitting the message. When nothing is received, the device opens the second window, where it listens on a slower, more robust configuration. The receive windows are used to acknowledge the message (if requested), and is the opportunity for the Application Server to send data back to a device. Although this approach is very energy efficient, as a sensor node only wakes up to send something, and does not periodically have to wake up to receive data, it is limited on how often a node wakes up. Therefore this Class A is not very suitable for latency sensitive applications, although a node is allowed to send empty (null) data frames.

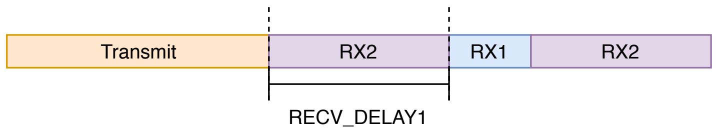
Class B allows for receive slots at scheduled times, called ping slots. For this purpose, the gateway sends out a time synchronous beacon every 128 seconds. All Class B nodes are assigned one or more time slot within the 128 second cycle to listen. This makes sure the device is listening at a particular time. A timing diagram is shown



(a) Class A



(b) Class B



(c) Class C

Figure 2.6: LoRaWAN timing diagrams for the various device classes.

in fig. 2.6b. Beacon guard (3 s) is the time preceding each beacon, and no ping slot can be placed in that time period. Beacon reserved (2.210 s) is a time period when the actual beacon is sent. Beacon window (122.880 s) is the time period when one can open and assign ping slots. A ping slot is a 30 ms unit of time. Class B devices are suitable for latency constrained services.

Class C is for devices that are mains powered. These devices remain in listening mode when they are not transmitting. Class C will listen at RX2 window as often as possible. After a transmission it will open a RX1 window like a Class A node, but in-between it will listen on a RX2 window. A timing diagram is show in fig. 2.6c.

Symphony Link

Symphony Link is a proprietary MAC by Link Labs [Lin]. It is a synchronised Time-Division Multiple Access (TDMA) protocol, with a fixed packet size of 256 bytes. It uses frequency hopping with LBT and AFA, allowing a Symphony Link network to transmit far more data than LoRaWAN, as it is not limited by the 1% duty cycle required in Europe and other regions (see section 2.4.2). This is particularly useful for over-the-air firmware updates, as that requires transferring large amounts of data in a short period of time. Both uplink and downlink message are acknowledged by default, making links much more reliable and suitable for industrial applications.

The network architecture of Symphony Link is similar to LoRaWAN, in that nodes communicate with gateways, which then forward the message to a central Network Server, or *Conductor* as it is called in Symphony Link. The Conductor then forwards the data to the relevant end-user application server.

Gateways in a Symphony Link network are much more powerful compared to LoRaWAN. In LoRaWAN, gateways are simply bridges between the LoRa network and backend network, providing little intelligence, to reduce costs and complexity. In Symphony Link, a gateway is given a much bigger role, decentralising the network, and making it more resilient to an internet outage. A gateway is responsible for assigning time slots for downlink and uplink messages to the nodes, based on their requirements (throughput, latency) and QoS class.

Both gateways and nodes in Symphony Link dynamically adjust their operation in real time to maximise performance and reliability. Gateways regularly scan the spectrum, marking channels with high levels of RF energy. This allow coexistence with other Symphony Link and LoRaWAN gateways, minimising collisions and avoiding interference from other RF systems.

Nodes dynamically adjust their data rate and output power for every transmission, based on reverse link budget. This enables a node to dynamically adjust its data rate and output power, to optimise the link quality without sacrificing network capacity. This approach allows nodes to quickly respond to fading channels, well before a gateway would notice.

2.4.6 Cellular IoT

The previous sections describe wireless communication standards that all operate in license-exempt bands. One of the wireless communication technologies we have not discussed yet, and the one that is the most ubiquitous of all, is cellular networks. LP-WAN technologies like LoRa and Sigfox require significant investments to be able to give a country-wide wireless coverage. Current cellular networks offer exactly that. However, traditional cellular options, such as 3G and LTE, consume too much power and do not fit well with applications of WSNs. 3rd Generation Partnership Project (3GPP), the force behind the standardisation of cellular systems, tries to solve this with Cellular IoT. In this section we describe the three most relevant cellular IoT standards under development: NB-IoT, LTE-M and EC-GSM-IoT [Lib+18].

NB-IoT

Narrowband IoT (NB-IoT), also known as LTE Cat-NB1, was added in 3GPP's Rel 13. It is the merger of two competing standards: Narrowband Cellular IoT (NB-CIoT), supported by Huawei, Vodafone and China Telecom, and Narrowband LTE (NB-LTE), supported by Nokia Networks, Ericsson and Intel. NB-CIoT is a clean slate approach, requiring new chipsets, and is not backwards compatible with any Long-Term Evolution (LTE) network older than Rel 13. NB-LTE, on the other hand, can be integrated into existing LTE networks, and works with current LTE bands.

NB-IoT focuses specifically on indoor coverage, low cost, long battery life, and high connection density. NB-IoT uses a subset of the LTE standard, but limits the bandwidth to a single narrow band of 180 kHz. This makes it very attractive for telecom operators to deploy NB-IoT, LTE is quite wide band (1.4 MHz to 20 MHz), and there are plenty of 200 kHz GSM spectrum bands that are unused (*e.g.* guard bands). NB-IoT is half-duplex, and uses OFDM modulation for downlink communication and Single Channel FDMA (SC-FDMA) for uplink communications. The maximum downlink date rate is 250 kbit/s, and the maximum uplink data rate is 20 kbit/s for single-tone and 250 kbit/s for multi-tone. The latency is relatively high, anywhere between 1.6 s to 10 s.

LTE-M

LTE-M, also known as LTE Machine Type Communication (LTE-MCM), Enhanced Machine-Type Communication (eMTC) and LTE Cat-M1, is a low-power variant of LTE. Compared to NB-IoT it offers higher data rates, up to 1 Mbit/s, shorter latency (10 ms to 15 ms). It is backwards compatible with existing LTE networks, requiring only a software update. LTE-M allows to be connected to a network, while not actually maintaining a physical connection. There are two power saving modes that help in saving energy: Extended Discontinuous Reception (eDRX) and Power Saving Mode (PSM).

A LTE device normally has to check in every 1.28 s. With eDRX, a device can tell the network how many frames it would like to sleep. The maximum can be set by the telecom operator, but should be at least 40 min.

PSM allows a device to tell the network it is going to deep sleep, for up to 413 days. On wake up, it transmit a (possibly empty) frame, and stays in receive mode for 4 frames. Combined with the high transmit rate, this mode can be extremely energy efficient for nodes that transmit for example only once a day.

EC-GSM-IoT

Extended Coverage GSM IoT (EC-GSM-IoT) is similar to LTE-M in that it is designed to operate in existing networks. In this case it is designed to operate in existing 2G eGPRS GSM networks, requiring only a software update. EC-GSM-IoT is half-duplex, having downlink and uplink data rates of 70 kbit/s to 240 kbit/s, depending on the used modulation scheme.

2.5 Conclusions

In this chapter we gave an overview of the research field of WSN and related research field. We described what constitutes and characterises a typical wireless sensor node, and how the communication architecture is organised. From this we gave a detailed description of the microcontroller commonly used, and highlighted new developments which can help solve existing and new challenges. Similar for the other major component of a sensor node, the radio, we described the current state of the art communication techniques, its capabilities and limitations, and future hardware. One challenge that arises from this context, is an expected massive growth to billions of nodes in a LPWAN network and the challenge of scalability in communication. We further explore and address this aspect in the next chapters.

Chapter 3

LoRa for the Internet of Things

LoRa is a proprietary spread-spectrum modulation technology, developed by Cycléo SAS, and acquired by Semtech in 2012 [Sem12]. It is a LPWAN technology that operates in the ISM band, and acquired quite some traction in the last couple of years. LoRa is interesting in that it allows to communicate over long distances, for a small energy budget. It allows us to transform existing multi-hop networks into star networks, changing the whole paradigm of WSN, and introducing new opportunities, and challenges.

This chapter investigates how LoRa works on the physical layer in theory, and in practice. We provide an extensive description of its properties, parameters, and peculiarities, which lays the groundwork necessary for the next chapters. Using LoRa's unique features, we build a novel MAC protocol called *LoRaBLink* that tries to address some of LoRaWAN's shortcomings.[1]

3.1 From Radio Waves to Bytes

How exactly the LoRa modulation works is not documented. There are patents ([SS13; Hor07]) and reverse engineering efforts, mostly by Knight et al. (see [KS16; Kni16])

[1]This chapter is based on the papers 'LoRa for the Internet of Things' [BVR16] and 'Do LoRa Low-Power Wide-Area Networks Scale?' [Bor+16].

that give some insights in how LoRa exactly works. This section describes what we know so far.

3.1.1 CSS Modulation

LoRa is based on CSS modulation, a rather classical technique in radar systems, which was proposed for the first time for communication systems by Winkler in 1962 [Win62] and barely used since. CSS uses a much larger bandwidth than needed for the given data rate. It is a subclass of DSSS, which uses controlled frequency diversity to recover data from weak signals, even under the noise floor. This helps in increasing the communication range, as receivers can be less sensitive, at the cost of a reduced data rate.

In DSSS, data is spread by modulating the message signal using a bit sequence, whereby a symbol is divided into N small chips. The sequence of chips used by the transmitter is known by the receiver, which uses this information to search for the pattern in the signal.

In contrast, CSS uses a continuously varying carrier frequency to spread the signal. At its most basic form, a transmitter can send one bit by either sending an up-chirp (linear increasing frequency over time) or a down-chirp (linear decreasing frequency over time). At the receiver end, the signal is multiplied with an up-chirp. The multiplication of a received up-chirp by a transmitted up-chirp result in an up-chirp, with instantaneous frequencies added. Multiplying a received up-chirp with a transmitted down-chirp, however, results in a narrow peak at twice the carrier frequency. By detecting the presence or absence of this narrow interference peak, the receiver can receive one bit per chirp.

The CSS modulation as used in LoRa is slightly more involved. LoRa improves on the basic CSS by encoding up to $\mathrm{SF} = 12$ bit per chirp. To achieve this, for each of the 2^{SF} symbols, a specific frequency trajectory is defined, by shifting the frequency

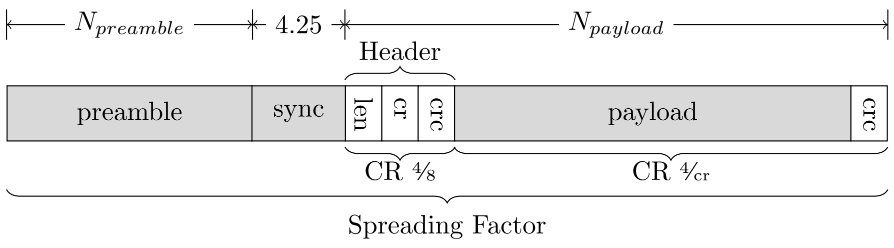


Figure 3.1: LoRa packet structure. Grey shaded areas are required, white shaded areas are optional.

ramp based on the symbol value. Each coded chirp is obtained by a cyclic shift of the reference chirp. Like in the basic form, multiplying the received signal with an up-chirp (or down-chirp) results in a narrow interference peak. This peak is now shifted on the spectrum, based on the value of the chip.

3.1.2 Packet Structure

The LoRa packet structure is shown in fig. 3.1. A packet starts with the preamble, programmable from 6 to 65535 symbols, to which the radio adds 4.25 symbols for the sync word. Thereafter follows an optional header, which describes the length and FEC rate of the payload, and indicates the presence of an optional 16-bit Cyclic Redundancy Check (CRC) for the payload. The header is always transmitted with a 4/8 FEC rate, and has its own CRC. After the optional header, there is the payload, which can contain 1 to 255 bytes. At the end of the payload an optional 16-bit CRC may be included.

3.2 Transmission Parameters

LoRa has five transmission parameters: Carrier Frequency (CF), Transmission Power (TP), Spreading Factor (SF), Bandwidth (BW) and Coding Rate (CR). The values of these parameters determines data rate, transmission range, energy consumption and resilience to noise and (narrow band) interference. The description of these parameters in the following sections is mostly based on documentation for the Semtech SX1272, but also applies to other sub-1 GHz LoRa radio transceivers.

3.2.1 Carrier Frequency

Carrier Frequency (CF) is the centre frequency used for the transmission band. A typical LoRa radio operates in the sub-1 GHz band, and is configurable in the range of 137 MHz to 1020 MHz, programmable in steps of 61 Hz. Depending on the regional requirements, this operating range is often limited to the local ISM bands. Newer LoRa radio chips like the Semtech SX1280 operate in the 2.4 GHz ISM band, which is more universally available, compared to the sub-1 GHz band.

3.2.2 Transmission Power

Transmission Power (TP) on a LoRa radio can be adjusted from −4 dBm to 20 dBm, in 1 dB steps. The LoRa radio chips often have a low power and high power antenna output, and since usually only one is connected, the range is often limited to 2 dBm to 20 dBm. In addition, power levels higher than 17 dBm requires special handling, changing the values for the over current protection, and often a 1% duty cycle is recommended.

3.2.3 Spreading Factor

Spreading Factor (SF) is the ratio between symbol rate and chip rate. A higher spreading factor increases the Signal-to-Noise Ratio (SNR), and thus sensitivity and range, but also increases the airtime of the packet. The number of chips per symbol is calculated as 2^{SF}. For example, with an SF of 12 (SF12) 4096 chips/symbol are used. Each increase in SF halves the transmission rate and, hence, doubles transmission duration and ultimately energy consumption. Spreading factor can be selected from 6 to 12. SF6, with the highest transmission rate, is a special case and requires specific operations, and packets with implicit headers. As in this case the receiving side needs to know the CR and packet size beforehand, this mode is not used often in practice. Radio communications with different SFs are orthogonal to each other and network separation using different SFs is possible.

3.2.4 Bandwidth

Bandwidth (BW) is the range of frequencies in the transmission band. Higher BW gives a higher data rate (thus shorter time on air), but a lower sensitivity due to integration of additional noise. Inversely, a lower BW gives a higher sensitivity, but a lower data rate. A lower BW also requires more accurate crystals (less parts per million (ppm)). Data is send out at a chip rate equal to the bandwidth. So, a bandwidth of 125 kHz corresponds to a chip rate of 125 kc/s. The SX1272 has three programmable bandwidth settings: 500 kHz, 250 kHz, and 125 kHz (BW500, BW250, and BW125 respectively). More advanced LoRa radios like the Semtech SX1276 can be programmed in the range of 7.8 kHz to 500 kHz, though bandwidths lower than 62.5 kHz requires a TCXO.

3.2.5 Coding Rate

Coding Rate (CR) is the FEC rate used by the LoRa modem and offers protection against bursts of interference. A higher CR offers more protection, but increases time on air. Radios with different CR (and same CF, SF and BW), can still communicate with each other, as long as packets are send with explicit headers. CR can be selected from 1 to 4 (CR1 to CR4), corresponding to a FEC rate of $4/(\mathrm{CR}+4)$. CR of the payload is stored in the header of the packet, which is always encoded at $4/8$. It is not specified what the FEC algorithm is, but from reverse engineering work we know that a (reduced) Hamming code is used [Kni16].

3.3 Derived Parameters

From the previous parameters, we can derive a couple of other parameters.

3.3.1 Modulation Bit Rate

The modulation bit rate R_b in bit/s is defined as follows:

$$R_b = \text{SF} \times \frac{1}{\frac{2^{\text{SF}}}{\text{BW}}} \tag{3.1}$$

where SF is the spreading factor and BW is the bandwidth in Hertz.

3.3.2 Effective Bit Rate

The effective bit rate R_e in bit/s is determined by multiplying the gross bit rate R_b with the CR as follows:

$$R_e = R_b \times \frac{\text{CR} + 4}{4} \tag{3.2}$$

3.3.3 Packet Size

The payload size of a packet in symbols depends on the SF, BW and CR, and is defined as follows:

$$N_{payload} = 8 + \max\left(\left\lceil \frac{8\text{PL} - 4\text{SF} + 28 + 16\text{CRC} - 20\text{IH}}{4\,(\text{SF} - 2\text{DE})} \right\rceil (\text{CR} + 4), 0\right) \tag{3.3}$$

whereby:

- PL is the payload in bytes.
- SF is the spreading factor.
- CRC is 1 if CRC is enabled, 0 otherwise.
- IH is 1 if implicit header mode is enabled, 0 otherwise.
- DE is 1 when low data rate optimisation is enabled, 0 otherwise. Low data rate optimisation is mandatory for SF11 and SF12 when using BW125.

- CR is the coding rate.

The total size of a packet in symbols is defined as follows:

$$N_{packet} = N_{preamble} + 4.25 + N_{payload} \tag{3.4}$$

whereby $N_{preamble}$ is the programmed preamble length, and $N_{payload}$ as defined in eq. (3.3).

It should be noted that a 32 B packet in one instance is 50 symbols large, while in another it is 48 symbols. Also, it could be that a 8 B packet is 25 symbols large, while a four times larger packet (32 B) only has twice as many symbols (50).

3.3.4 Airtime

Airtime, or how long the transmission time is, is determined by packet size (in symbols) and symbol rate (determined by Spreading Factor (SF) and Bandwidth (BW)). The transmission time for a symbol is defined as follows:

$$T_{sym} = \frac{2^{\mathrm{SF}}}{\mathrm{BW}} \tag{3.5}$$

From eq. (3.4) we know how many symbols a packet is. By multiplying eq. (3.4) with eq. (3.5) we can determine the airtime of a packet, *i.e.*:

$$T_{air} = N_{packet} \times T_{sym} \tag{3.6}$$

3.3.5 Energy Consumption

The energy consumed E in joule by a LoRa radio transmission is defined as follows:

$$E = \mathrm{TP} \times T_{air} \tag{3.7}$$

where TP is the transmit power in Watt, and T_{air} is the airtime in seconds as defined in eq. (3.6).

Table 3.1: Current consumption for various transmit powers and corresponding PA configuration of the Semtech SX1272.

TP (dBm)	PA	Current (mA)	TP (dBm)	PA	Current (mA)
-1	PA0	22	10	PA1	31
0	PA0	22	11	PA1	32
1	PA0	23	12	PA1	34
2	PA1	24	13	PA1	35
3	PA1	24	14	PA1	44
4	PA1	24	15	PA1	82
5	PA1	25	16	PA1	85
6	PA1	25	17	PA1	90
7	PA1	25	18	PA1+PA2	105
8	PA1	25	19	PA1+PA2	115
9	PA1	26	20	PA1+PA2	125

Transmit power is often defined in dBm, and this correlates with the actual power consumption. On the Semtech SX1272 (and many other LoRa radios), this does not scale linearly. This is because of the architecture of the RF front end, shown in fig. 3.2. Exact figures are chip specific, but most LoRa chips have the same architecture.

The RF front end consists of 3 power amplifier (PA) blocks: PA0, PA1 and PA2. PA0 is an unregulated, high efficiency power amplifier, with a range of −1 dBm to 14 dBm, and connected to pin RFO. Since this pin is often not used, this mode of operation is usually not considered.

PA1 and PA2 are regulated power amplifiers that are connected via the RF_BOOST pin. The PAs can be used in either *programmable* or *fixed* configuration. In programmable configuration, PA1 is used to output in a range of 2 dBm to 17 dBm, programmable in 1 dB steps. In fixed configuration, PA1 is combined with PA2 and the output can be set in a range of 5 dBm to 20 dBm, in 1 dB steps. Note that a transmit power of 20 dBm can only be used with a 1% duty cycle, and requires adjustment to the over current protection.

The typical current consumption of a Semtech SX1272 [Sem] is shown in table 3.1.

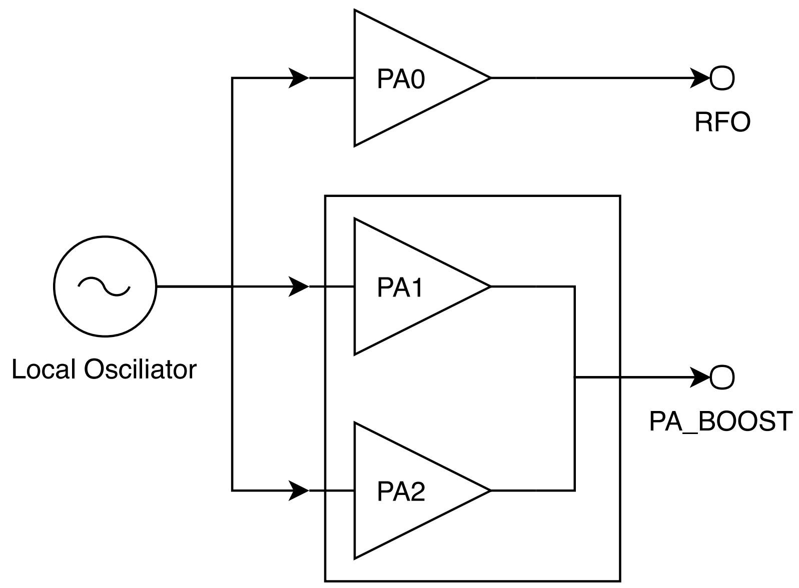


Figure 3.2: RF front-end architecture of the Semtech SX1272 showing the internal PA configuration.

3.4 Carrier Activity Detection

Radio transceivers usually provide a Clear Channel Assessment (CCA) interface to detect an occupied channel. CCA is used in communication protocols in two ways: for transmissions to decide if packets if the channel is occupied, and for reception to detect whether the radio must be kept active to receive an ongoing transmission. In particular for power constrained nodes it is important to have an accurate and fast CCA mechanism as this enables implementation of power-efficient duty cycling. Nodes perform periodic short CCA checks and only power the receiver for longer if a transmission is detected.

LoRa transceivers do not provide a traditional CCA interface based on an Received Signal Strength Indicator (RSSI) threshold to detect an occupied channel. LoRa can receive transmissions with a signal strength that is below the noise floor and, consequently, an RSSI threshold check will not reveal an occupied channel. LoRa radios provide therefore a Carrier Activity Detection (CAD) mode to detect a present preamble.

CAD is not as generic as CCA as it can only detect an occupied channel when a preamble is present. If the CAD is activated during the transmission of the packet payload (after the preamble), it will return a false negative.

The CAD process takes approximately 2 symbol periods, and runs in two phases: receiving and processing. In the receiving phase, the radio is enabled to listen for any preambles for about 1 symbol period. The radio is then switched off, and the received signal is analysed to detect the presence of a preamble. This processing phase requires about half the energy required in receive mode (around 6 mA depending on the SF/BW). When a preamble is detected, a 'CAD detected' interrupt is fired. The microcontroller can then decide to switch the radio in RX mode to receive the ongoing transmission, if required. The exact CAD duration in seconds is calculated as:

$$T_{CAD} = \overbrace{\frac{32}{\mathrm{BW}} + \frac{2^{\mathrm{SF}}}{\mathrm{BW}}}^{\text{receiving}} + \overbrace{\frac{\mathrm{SF} \times 2^{\mathrm{SF}}}{1750 \times 10^3}}^{\text{processing}} \tag{3.8}$$

whereby SF is the spreading factor, and BW is the bandwidth in Hz. The first two terms, as indicated by the brace 'receiving', calculate the duration the receiver is active. The two last terms, as indicated by the brace 'processing', calculate the time spend on processing the received data.

3.5 Feature Evaluation

LoRa has interesting features, aside the increased communication range, that should be taken into account when constructing network protocols. For example, channel separation using different SF is possible, concurrent non-destructive transmissions are possible and carrier detection via CAD is provided. However, from available documentation the performance and ability of these features is not clear. Therefore we carry out a series of experiments to evaluate these provided features.

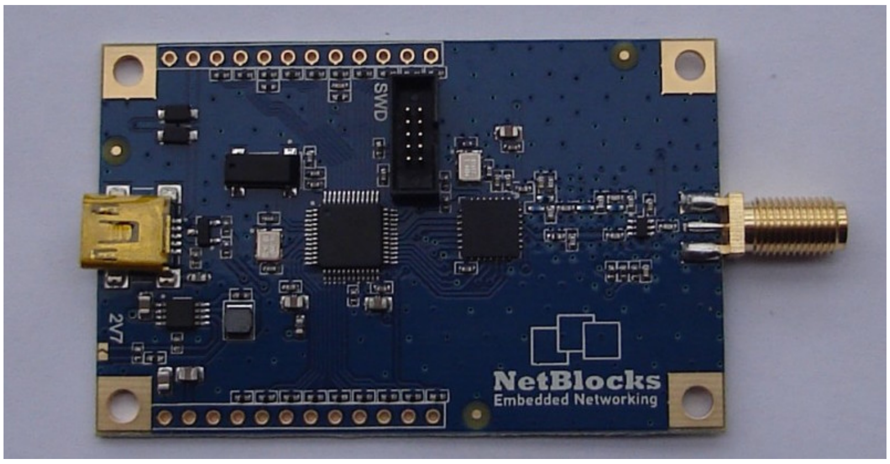


Figure 3.3: NetBlocks XRange SX1272 LoRa RF module.

3.5.1 Experimental Device

For our studies we use the XRange SX1272 LoRa RF module device from NetBlocks[2] as shown in fig. 3.3. The device comprises a Semtech SX1272 LoRa transceiver and a low-power STMicroelectronics STM32L151 ARM Cortex-M3 microcontroller. We use GCC ARM and the LoRa radio driver and runtime environment derived from the IBM LoRaMAC in C (LMiC)[3]. The runtime environment, code used for our experiments is available at http://www.lancaster.ac.uk/scc/sites/lora/.

The energy consumption of the system is for most application cases dominated by energy cost for communications. Energy consumption for transmission, reception, listening, and CAD must be distinguished. The energy consumption in these states depends on selected SF and BW. Also, selected communication parameters will influence transmission times of packets and ultimately energy consumption.

To give an example we assume SF12, BW125, CR1, and TP 17 dBm. This is an energy hungry configuration allowing for very long ranges that was used in our experimental evaluation discussed later. A transmission of a packet with 10 B payload and 12.25 symbols preamble has a transmission duration of 991.23 ms. Transmitting such message

[2]http://www.netblocks.eu/
[3]http://www.research.ibm.com/labs/zurich/ics/lrsc/lmic.html

will cost 214 mJ. Reception of this message will cost 25.7 mJ and performing a CAD will cost 1.23 mJ. This excludes any local processing, which is often negligible.

If we assume a system where the above message is transmitted every 15 min and we assume a battery capacity of 2 typical AA batteries of 5400 mAh the node will have a lifetime of 6.2 years.

If we assume a node only carries out a CAD every 5 s to check for an incoming message the node will have a lifetime of 6.0 years (assuming again 5400 mAh battery capacity).

3.5.2 Orthogonal Spreading Factors

Different spreading factors are claimed to be orthogonal to each other. Thus, construction of virtual channels on the same carrier frequency is possible, alike Code Division Multiple Access (CDMA).

We evaluate how well this separation works using a simple experimental setup. A transmitter is set to continuously send a 40 B packet with a fixed SF. A receiver set to the same SF is used to receive the transmissions. A second transmitter is used to transmit continuously and sequentially using all other SF to the same receiver.

The transmitter is located in an office, while the receiver is placed in a different office on the same floor, about 30 m apart. The second transmitter is placed in the office opposite the first transmitter, approximately 10 meters apart.

Findings

All transmissions where sender and receiver use the same SF are received correctly. None of the transmissions emitted by the second node using a different SF are received. This result suggests that channel separation using SF works perfectly.

However, as we will show in section 3.5.4 this is only partially true. When using CAD to detect an incoming transmission a signal using the wrong SF may be detected as valid transmission even though it cannot be decoded. This is important as the false detection rate has a negative impact on energy efficiency of a protocol.

3.5.3 Concurrent Transmissions

In LoRa concurrent transmissions are claimed to be non-destructive and such feature is very valuable for protocol design. Well-timed cooperative transmissions have been used in Glossy [Fer+11]. In Glossy the same message is transmitted accurately timed by multiple nodes allowing correct reception. A-MAC [Dut+10] and Whitehouse et al. [Whi+05] also make use of the capture effect. Here multiple different messages are transmitted concurrently and depending on power levels and timing one of the concurrently transmitted messages can be received.

We set up an experiment to understand the exact conditions in which this effect is present in LoRa. We use a receiver, one *weak* transmitter set to transmit at 2 dBm, and one *strong* transmitter set to transmit at 3 dBm. We chose the 1 dBm difference to evaluate the worst case.

Like with the experiments with orthogonal spreading factors, the transmitters are placed in offices opposite each other, about 10 m apart. The receiver is placed in an office on the same floor, about 30 m from both transmitters.

Both transmitters send the same 32 B packet with explicit header and CRC. The strong transmitter varied the transmission time offset relative to the weak transmitter. From being one packet (airtime) early to being one packet (airtime) late. For each offset, sixteen 32 B packets were transmitted using first identical packet payloads and subsequently different payloads. We also run the experiment with all combinations of SF and BW.

The experiment results are shown in fig. 3.4 for SF12 and BW125. The Y-axis represents the Packet Reception Rate (PRR). The X-axis represents the transmission offset relative to the weak node in symbol time. The top bar shows packets received from

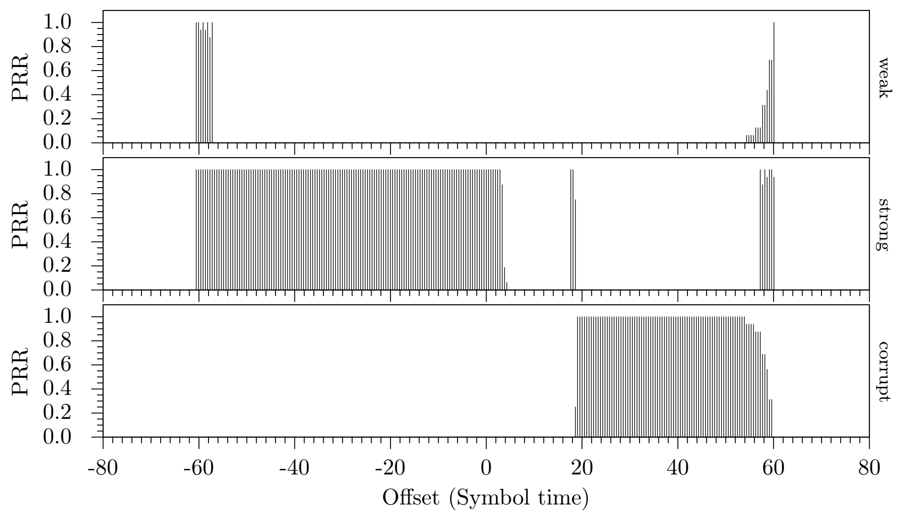


Figure 3.4: Example collision result. Spreading factor 11, bandwidth 125 kHz. X-axis shows the transmission offset relative to the weak node in symbol time, Y-axis shows the Packet Reception Rate (PRR).

the weak transmitter at the receiver; the middle bar shows packets received from the strong transmitter at the receiver. The bottom bar shows when packets were received from either transmitter, but deemed corrupt (CRC failure). We did notice that about 1 in 6000 packets was corrupted, but did not fail the CRC. Often these packets had 1 bit corrupted.

Results for other SF and BW combinations are very similar. Also, transmitting the same packet payload or a different payload does not change the obtained results significantly.

As can be seen, the strong transmitter is successfully decoded if it transmits not later than 3 symbol periods after the weak transmitter started. If the weak transmitter starts later than 3 symbol periods no transmission is received (or corrupted data is received).

Although the packet takes 60.25 symbol periods to transmit, both nodes can be received at an offset of -57 symbol periods or more. The tail of the strong node does destroy the initial preamble of the weak node, but as long as at most 3 symbols are destroyed, the weak packet can also be successfully received. This relationship is not symmetrical, as at an offset of +57 symbol periods, the weak node's tail (CRC) gets destroyed, invalidating a packet that may have been correctly received. It is only that at an offset of +60 symbol periods or more, both packets gets received perfectly.

We also experimented with two transmitters set to the same transmit power. In this case either of the two is perceived as stronger and the above described behaviour applies (although the role of stronger/weaker transmitter may alternate with each transmission making it difficult to conduct experiments and describe results).

Findings

One of two concurrent transmission can be received with very high probability if both transmissions do not have an offset of more than 3 symbol periods. This translates to a duration between 768 μs and 98.3 ms, depending on the SF and BW. Synchronisation of nodes within these bounds is relatively easy to achieve and therefore protocols making use of this feature can easily be implemented with LoRa.

3.5.4 Carrier Activity Detection

We set up an experiment to test the reliability of CAD. A detector node starts the CAD process on a regular interval (every 100 ms) and records whether it has detected a carrier or not. After 300 samples, it switches the SF/BW combination and repeats the process. A transmitter node is programmed to continuously send out preambles at 2 dBm, with a fixed SF/BW combination. The experiment is repeated with different transmitter SF/BW combinations. Both transmitter and receiver are located in separate offices, on the same floor, about 30 m apart.

The results for a transmitter using SF7 and BW250 are shown in fig. 3.5. Results for a transmitter using different SF/BW combinations are similar. When transmitter and receiver use the same SF/BW combination the worst detection rate was meas-

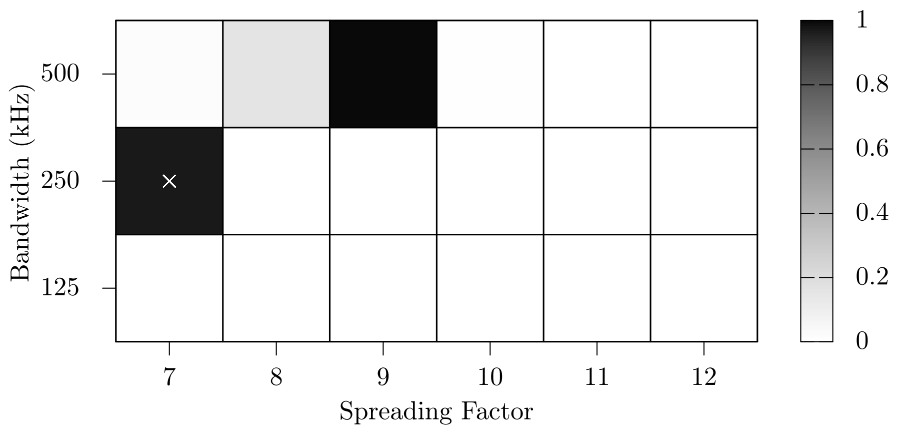


Figure 3.5: Carrier detection ratios for a transmitter sending at spreading factor 7 and bandwidth 250 kHz, indicated by the white cross. Carriers were also detected by adjacent data rates.

ured at 97% (for SF7 and BW250). However, the CAD process also detects carriers in SF/BW combinations different from the combination the transmitter is using (up to 99% detections for SF9 and BW500). This happens for data rates that are adjacent to the current data rate. In those case, the receiver enable receive mode with the wrong settings, and will not be able decode an ongoing transmission (whether it is intended for the receiver or not). With no transmitter active, the false positive rate is 0.092%.

Findings

CAD can only detect channel occupancy while a preamble is transmitted. The detection probability is high (above 97%) and false positives are low (0.092%). However, if multiple LoRa networks are active on different SF/BW combinations false positives can be very high (depending on SF/BW ratios). When using multiple SF/BW combinations in the same network (or when constructing multiple networks separated by SF/BW) the choice of combinations is important when using CAD.

3.6 Medium Access Control

With the ability to send and receive packets, we can address the next layer of the OSI model: the MAC layer. The MAC layer is responsible for scheduling communication between devices, ensuring robust communication, while minimising energy consumption.

The de facto standard MAC for LoRa is LoRaWAN, which is described in detail in section 2.4.5. While LoRaWAN has helped in deploying LoRa networks [Ora12; The19a; KPN19], it is not without its shortcomings. Using the findings from the previous sections, we address these shortcomings by constructing a novel MAC called LoRaBlink.

3.6.1 Limitations of LoRaWAN

Although LoRaWAN provides a lot of features that makes it possible to build a robust IoT application, there are some limitations. Some of these limitations are a consequence of the design of LoRaWAN, and some are not addressed in the standard.

Single-Hop Network LoRaWAN is set up as a single-hop star network. While this greatly simplifies deployments and operation, it also means there should be enough gateways to cover the whole deployment area. In an urban environment this may be achievable, but in more rural areas this can become challenging, as nodes are more nodes are spaced out, and power and a wired, or wireless backbone Internet connection may not be readily available. Being able to support a (shallow) multi-hop network would help in connecting any nodes that are in a hard-to-reach place.

Inefficient Spectrum Usage LoRaWAN is based on pure ALOHA. In pure ALOHA, a network has a theoretical maximum throughput of 18% [TW11]. That is, 82% of all transmitted message collide and are lost. As shown in section 3.5.2, due to the capture effect and orthogonal SF, a LoRa network could perform slightly better.

However, from the literature we already know we can do much better. For example the theoretical maximum throughput of slotted ALOHA is 37%. Especially with the envisioned thousands to millions of nodes, and limited acknowledgement support, this can be quite a problem. Some form of scheduling or time slots could therefore greatly improve the performance of LoRaWAN.

Duty Cycle A LoRaWAN node has a maximum duty cycle of 1% (in Europe), allowing to send at most 3.6 s every hour. With a data rate of 250 bit/s, this means a node can send at most 27 kB per day on average. If required, a node could send more data in a burst, but that would mean it has to stay quiet for a longer time. In Europe, as an alternative to a 1% duty cycle, a transmitter can also use LBT with AFA. The radios used in most LoRa nodes are capable of supporting these features, but presumably for simplicity, LoRaWAN does not use this mechanism.

This regulatory constraint affects gateways even more. Unlike nodes that have to only send data to one gateway, gateways have to support several thousands of nodes. As gateways are also bound to the 1% duty cycle, this severely limits the amount of data (*e.g.* acknowledgements, firmware updates) that can be send to sensor nodes.

Real-time LoRaWAN is not suitable for applications requiring real-time data and control, for two reasons. Firstly, Class A nodes are asynchronous, bound to a 1% duty cycle, and can only receive a message when they send a message first. A gateway has to wait for a node to wake up, before it is able to send any control message. Using Class B would be better, as a node gets scheduled receive slots, but consumes a whole lot more energy.

Payload Size The relative low data rate and duty cycle also limits the kind of data that could be send. Depending on the attainable data rate, the maximum payload per message is between 51 B to 222 B. This makes it suitable for only sending small status messages and sensor readings. Sending larger payloads, like audio, or video, are not practically possible.

Downlink Messages LoRaWAN is a bidirectional protocol, but it is engineered as an upstream protocol. The vast majority of the messages go from the node to the gateway. While gateways can receive up to 8 concurrent message, they can only transmit to one node at a time. Since gateways are half duplex, this also means that while transmitting, a gateway cannot receive any messages. Downlink messages, even as simple as acknowledgements for uplink message, are therefore highly discouraged by LoRaWAN.

The limitations on downlink messages severely hinder support for performing Over-the-Air (OTA) firmware upgrades. For example, doing a OTA firmware upgrade with a 100 kB firmware image, will take about 9 hours in the ideal situation where the maximum data rate can be achieved. Updating a large set of nodes therefore becomes very time consuming. A gateway could speed this process up by sending the firmware fragments as a burst of messages, potentially reducing the transfer time to around 6 minutes. Downside is that after this firmware upgrade, a gateway cannot send any downlink messages for about 10 hours to stay within the 1% duty cycle limit.

Another approach to overcome this limitation, is by sending firmware upgrades using multicast. With multicast, messages are sent to a group of nodes, instead of to each node individually, as it the case with unicast. Since firmware images are rarely unique for each node, supporting multicast is essential. LoRaWAN, however, does not support sending multicast frames.

3.6.2 LoRaBlink

With the aforementioned limitations, and findings during our feature evaluation, we developed a novel MAC protocol tailored for a LoRa-powered IoT network called *LoRaBlink*. LoRaBlink is inspired by Glossy [Fer+11], and is designed to support reliable and energy efficient multi-hop communication. It is also designed to support low-latency bidirectional communication.

Protocol Design

LoRaBlink aims to address a number of aspects necessary for deployment of IoT applications and which are not covered by currently defined LoRaWAN protocol. These are:

- *Multi-Hop:* The protocol should support communication over multiple hops.
- *Low-Energy:* Nodes should be able to duty-cycle to conserve energy and enable battery powered operations over long time spans.
- *Resilience:* The protocol should be resilient and enable high message delivery probability.
- *Low-Latency:* The protocol should enable low-latency communication.

Further to these requirements we also make the assumption that the network has a low density, low traffic volume and contains a limited number of nodes. We also assume that a single sink is used for communication and that communication is between the sink and the nodes.

A vast number of protocols exist to implement these requirements [Bac+10]. However, none of the available options is particularly designed to make use of LoRa specific features such as the ability to receive one message out of a pool of concurrent transmissions.

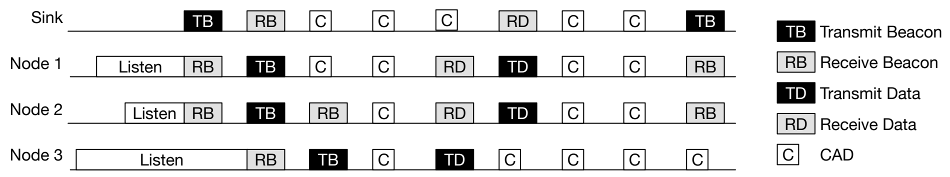


Figure 3.6: LoRaBlink: Protocol example using a 4 node network.

Protocol Operations

The protocol integrates MAC and routing in a single simple protocol. Time synchronisation among nodes is used to define slotted channel access. Nodes transmit concurrently within slots and properties of the LoRa physical layer ensure that one of the concurrent transmissions is received. Messages are distributed from the sink to nodes using flooding. Messages from nodes to the sink use a directed flooding approach. The result is a very simple, but robust protocol that covers the set requirements.

Figure 3.6 shows an operation example of LoRaBlink in a network containing 3 nodes and a sink. Node 1 and 2 are in communication range of the sink. Node 3 cannot be directly reached by the sink but is in range of node 1 and 2.

Each node powering up will remain in listen mode until a beacon is received. Beacons are used for time synchronisation and mark the start of an epoch. Each epoch contains N slots. The first N_B slots of an epoch are used for beacon transmissions. A beacon message contains the hop distance to the sink and upon receiving a beacon a node will transmit its own beacon according to its distance to the sink. A node will aim to select its position based on minimal distance to the sink. In the example in fig. 3.6 the sink transmits a beacon received by node 1 and node 2. Both nodes use the beacon to determine epoch start and their distance to the sink (1 hop). In the next beacon slot node 1 and 2 transmit their beacon concurrently. Due to properties of the LoRa physical layer either one of these (depending on transmission time difference and perceived signal strength at node 3) is received at node 3. Node 3 updates its hop distance to the sink as 2 and transmits its own beacon in the next beacon slot. This beacon is received by node 2 (we assume node 1 would not receive) which discards the message as its hop count is less than 2. The number of beacon slots N_B determines the maximum depth the network can have.

Following the beacon slots are N_D data slots. A node that has data to transmit selects the next available data slot and transmits. After transmission a node listens for an Acknowledgement (ACK) (an optional protocol feature; the ACK is not shown in fig. 3.6). Two nodes may transmit in the same slot with the result that at least one message is decoded by one receiver, with a chance that two different nodes in the network decode one of each transmission. If a node has a lower hop count to the sink than the source node it will relay the message in the next slot. Multiple nodes may forward which introduces redundancy. ACK messages may also collide but a receiver will always be able to decode one of multiple ACK correctly. In fig. 3.6 node 3 generates a data message. The message is received by node 2 and 1 which then forward the message simultaneously in the next slot. The sink will be able to decode one of the two transmissions. Data travelling from the sink to a node will use the same mechanism as used for beacon distribution. If the sink has to send non-delay sensitive information to nodes in the network it can be delayed and included in beacon messages for distribution.

Node Lifetime

To improve energy consumption of the system beacon messages are sent infrequent (a long epoch is used) and the CAD is used within slots to detect incoming transmissions. Infrequent beacon transmission is possible as tight time synchronisation in the network is not necessary.

Epoch length and N_B and N_D determine energy consumption and data transport delay in the network. While not the most energy efficient protocol, the additional range may be a benefit to low-node-density deployments, requiring far fewer

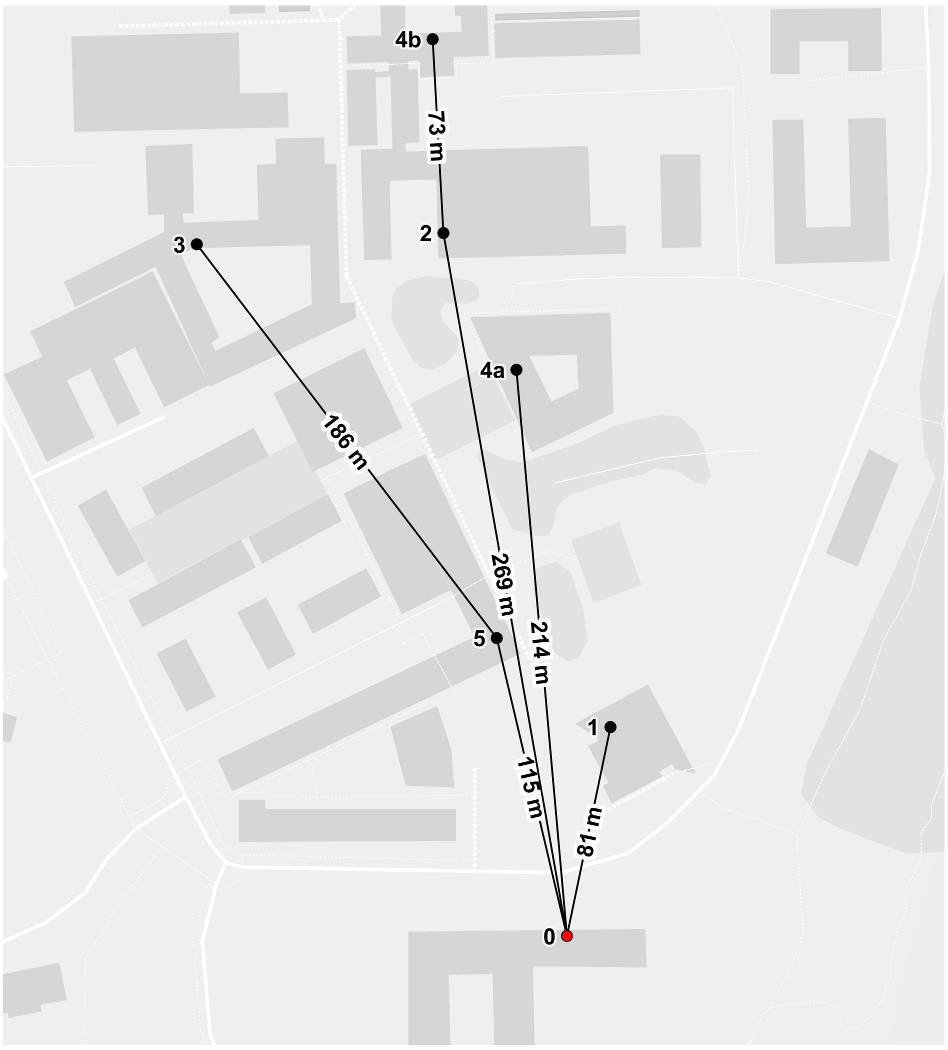


Figure 3.7: LoRaBlink: Map of a small scale deployment. Lines are routes between nodes, with distance in meters.

forwarding nodes to cover the same area. The transceiver configuration described in section 3.5.1 and an epoch length of 15 min with $N_B = 3$ and $N_D = 177$ (5 second slot) we obtain a maximum node lifetime of 2 years with two AA batteries (5400 mAh) – assuming that one beacon is transmitted and two are received and all other slots in the epoch contain one CAD.

3.6.3 Proof of Concept

To demonstrate the feasibility of LoRaBlink, we deployed a 6 node network on the south-end of the Lancaster University campus as shown in fig. 3.7. Nodes are deployed in buildings across campus on the ground floor within buildings approximately 1.5 m above the floor. The sink node is located in the third floor on a windowsill. Node 4 was first deployed at position 4a and was moved to position 4b to create a larger network. Nodes have to communicate through several buildings and structures.

We use in the experiment SF12 and BW125 and a TX power of 17 dBm. The epoch length was set to 5 min with $N_B = 5$ and $N_D = 55$ (slots every 5 seconds). Nodes are set to transmit a data packet (10 B) randomly within one slot of each epoch. In this experiment we did not use CAD and instead implemented a listen period of 50 symbols in each slot. This was done to avoid packet losses due to CAD and to evaluate data delivery of LoRaBlink on its own.

In our evaluation packets from all nodes were delivered with a reliability of 80% over a duration of 2.3 h. Node 4 delivered messages for the first half of the experiment from the position marked 4a and later from position 4b. Node 3 and 4b delivered messages via one hop while all other nodes were able to directly communicate with the sink node. Messages transmitted by node 3 and 4b are relayed by multiple nodes (node 5, node 2 and node 4a for node 3) in the same slot.

The experiment shows that LoRaBlink can deliver messages reliably over large distance in a challenging multi-hop environment (buildings and objects in the communication path). The experiment also shows that using concurrent transmissions is feasible.

3.6.4 Evaluation

An in-depth performance evaluation of LoRaWAN and LoRaBlink would be quite interesting, but requires a clear workload and an experimental setup that is beyond

Table 3.2: LoRaWAN vs LoRaBlink feature comparison

Feature	LoRaWAN	LoRaBlink
Topology	single-hop	multi-hop
Spectrum Usage	low	medium
Real-time	low	high
Energy Consumption	low	medium
Duty Cycle	low	low

the scope of this work. Instead, we evaluate LoRaBlink on five features:

- *Communication Topology:* In what topology nodes can communicate.
- *Spectrum Usage:* How efficient the communication channel is utilised.
- *Real-Time:* How suitable communication is for real-time operation.
- *Energy Consumption:* How energy efficient the protocol is.
- *Duty Cycle:* What the maximum duty cycle is.

Table 3.2 shows a summary of the comparison between LoRaWAN and LoRaBlink. The following sections evaluates each feature in more detail.

Communication Topology LoRaWAN is single-hop star network, which eases deployment as end nodes do not have to maintain any topology information. A downside to this approach is that all the sensor nodes need to be in range of a gateway. Any nodes that are in a ‘bad' spot, cannot communicate with the network.

LoRaBlink is a (shallow) multi-hop. All nodes in the network participate in sending messages from the source to the destination, either upstream from node to sink, or downstream from sink to nodes. Nodes in a ‘bad' spot are able to communicate as long as there is a path to the sink and *vice versa*. As LoRaBlink use designated beacon broadcast slots, the depth of the network (the number of hops from source to sink) are limited to the amount of broadcast slots.

Spectrum Usage Since LoRaWAN is based on pure ALOHA, the lower bound for spectrum usage is 18%. In practice, LoRaWAN will do slightly better, as collision are not destructive for all packets.

LoRaBlink employs TDMA, which greatly improves the spectrum usage. Worst case is equivalent to slotted ALOHA, which has a spectrum usage of 37%. Similar to LoRaWAN, the performance in practice will be slightly higher as collision do not have to be destructive. LoRaBlink does have designated broadcast slots, but not designated data slots. Any node can transmit whenever it has data. For future work, spectrum usage could be improved by assigning slots to particular nodes, at the cost of making the protocol more complex.

Since all transmissions happen in timed slots, However, utilisation is limited by the number of beacon slots and the length of the epoch, as well as the guard times between slots to counter for propagation delays and clock drift. Supporting deeper networks comes at the expense of useful air time.

Real-time With LoRaWAN, the latency is very dependent on the node communication. Communication from end node to gateway can happen whenever the node has data to send (and is not prohibited by regulatory limitations).

With LoRaBlink, the latency is known before hand, as each node listens and communicate on a set interval. While a node deep in the network on a high level, has a high latency, it has a known latency. Therefore for real-time communication we can give guarantees on when a node can communicate.

Energy Consumption LoRaWAN has a low energy consumption, as nodes only need to wake up when they have data to send. They do not need to wake up to maintain the network, although it is encouraged to wake up on regular intervals to listen for gateway beacons to ensure the node is still within range. The downside of this approach is that especially in more dense

LoRaBlink is more energy consuming than LoRaWAN, as nodes have to regularly listen to beacons, and forward beacons and data. By using CAD, the energy consumption is reduced as opposed to just listening for beacons and data.

Duty Cycle Both LoRaWAN and LoRaBlink are limited by a 1% duty cycle as required by regulations. The duty cycle for both protocols is therefore low.

A way to get around this limitation is by using LBT and AFA (see section 2.4.2). LoRaBlink does use a form of LBT in the form of CAD, but this is will only detect LoRa preambles. For LBT, the node has to be able to detect ongoing transmissions from any communication technology that is above the RSS threshold. Semtech gives some guidance on how sample the RSSI register while performing CAD [14]. Although this approach is more designed for supporting antenna diversity, this method could be used to implement LBT.

3.6.5 Future Work

LoRaBlink provides some improvements over the limitations of LoRaWAN by utilising LoRa's unique features. Future work revolves around improving the channel utilisation, and performing an in-depth performance evaluation. For the latter, a proper LoRa testbed is required, and a benchmark workload has to be defined. This work can be useful to evaluate other MAC protocols, such as LoRaWAN and Symphony Link.

3.7 Conclusions

In this chapter, we had a look at LoRa, a novel wide band spread spectrum modulation technique. We looked at the theoretical operation of the physical layer in detail, and described the parameters of LoRa, and the properties and effects it has on the wireless link. LoRa has a range of interesting features, aside from the long communication range. The developers of LoRa claim it provides channel separation via orthogonal

SFs, concurrent transmissions via the capture effect, and Carrier Activity Detection (CAD). To substantiate these claims, we performed a series of experiments. We discovered that the channel separation via different SFs works perfectly. However, we did find that CAD can be falsely triggered with particular SF/BW combinations. The concurrent transmissions did work quite well, and messages could be retrieved with high probability. We did discover that the timing of the concurrent transmissions is quite critical, as they cannot overlap within a particular section of the transmission. Using these findings, we constructed a novel TDMA-style MAC protocol, called LoRaBlink, to address some of the shortcomings of LoRaWAN. We showed the feasibility of LoRaBlink via a proof of concept deployment, and compared its features to LoRaWAN to demonstrate its merits.

Chapter 4

LoRa Scalability

In the previous chapter we have explored LoRa from the physical layer. One of the compelling reasons for using LoRa is to build a large scale star network. Support for thousands of devices is one of the advantages the LoRaWAN application protocol, which is build on top of LoRa, claims. To substantiate these claims, we first develop a model of a LoRa link, based on the experimental data we gathered in chapter 3. From this model, we build a simulator called *LoRaSim* that we use to simulate a large scale LoRa network under various conditions. We test the limitations of LoRa, and specifically LoRaWAN, and determine the bounds of scalability.[1]

4.1 Related Work

There is limited published work discussing scalability of LoRa. Closest to this is the work by Petäjäjärvi et al. [Pet+15], and our work in chapter 3. Petäjäjärvi et al. present an evaluation of LoRa link behaviour in open spaces. In another paper [Pet+16], the same authors evaluate the coverage and reliability of a LoRa node operating close to a human in an indoor area. The authors also analyse the capacity and scalability of LoRa in a more general approach [MPH16]. This work however seems to be based

[1]The chapter is based on the paper 'Do LoRa Low-Power Wide-Area Networks Scale?' [Bor+16].

mostly on the theoretical data rather than real-world calibrated simulations as we do. W evaluate LoRa link behaviour in built-up environments, and build upon the results reported in these papers when constructing our communication models for LoRaSim.

A vast number of wireless simulation tools exist, such as ns-3 [RH10], or OMNet++ [Var01]. There are also simulators designed for WSN and IoT environments, such as Cooja [Öst+06] or TOSSIM [Lev+03] These simulators can be extended by the components designed for our simulator LoRaSim to enable LoRa simulations.

The Semtech LoRa modem calculator [Sem] helps with analysis of LoRa transmission features (airtime of packets, receiver sensitivity) but does not enable network planning.

Siradel provides a simulation tool called *S_IoT* [Sir17]. S_IOT relies on Volcano, a 3D-ray tracing propagation model and a portfolio of 2D and 3D geodata. The tool supports sink deployment decisions based on propagation models. This commercial tool considers the environment to a much greater detail than LoRaSim. However, it does not take into account actual traffic, collisions or details such as capture effect.

4.2 Link Behaviour

In this section we develop a model of LoRa communication behaviour that we use subsequently in our simulation environment LoRaSim. Specifically, we develop a model describing (i) achievable communication range in dependence of communication settings SF and BW, and (ii) capture effect behaviour of LoRa transmissions depending on transmission timings and power. The development process of these models is supported by practical evaluation.

4.2.1 Communication Range

A transmission is successfully received if the received signal power P_{rx} lies above the sensitivity threshold S_{rx} of the receiver. The received signal power P_{rx} depends on the transmit power P_{tx} and all gains and losses along the communication path:

$$P_{rx} = P_{tx} + G_{tx} - L_{tx} - L_{pl} - L_m + G_{rx} - L_{rx} \tag{4.1}$$

P_{rx} is the received power in dB, P_{tx} is transmitted power in dB, G_{tx} is the transmitter antenna gain in dBi, L_{tx} is the transmitter loss (*e.g.* RF switch, non-matching circuit, connectors) in dB, L_{pl} is the path loss in dB, L_m are miscellaneous losses (fading margin, other losses) in dB, G_{rx} is the receiver antenna gain in dBi and L_{rx} are receiver losses.

For the purpose of this study we simplify this general equation to:

$$P_{rx} = P_{tx} + GL - L_{pl} \tag{4.2}$$

Here, GL combines all general gains and losses while L_{pl} represents the path loss, determined by the nature of the communication environment.

On the transmitter side, range can only be changed by changing the transmit power. Other parameters like SF, BW and CR do not influence the radiated power, or any other gains and losses. On the receiver side, the range is limited by the sensitivity threshold S_{rx}, which is influenced by the LoRa parameters SF and BW.

Path Loss

Many models exist to describe path loss in dependence of different environments (built-up area, free space). We use the well known log-distance path loss model [Rap02] which is commonly used to model deployments in built-up and densely populated areas. We choose this model as it matches environments in which we expect LoRa deployments are to be found. Using this model the path loss in dependence of the communication distance d can be described as:

$$L_{pl}(d) = \overline{L_{pl}}(d_0) + 10\gamma \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma \tag{4.3}$$

where $L_{pl}(d)$ is the path loss in dB, $\overline{L_{pl}}(d_0)$ is the mean path loss at the reference distance d_0, γ is the path loss exponent and $X_\sigma \sim N(0, \sigma^2)$, the normal distribution with zero mean and σ^2 variance to account for shadowing.

The advertised communication range of LoRa is more than 15 km for suburban environments. Petäjäjärvi et al. [Pet+15] have reported a range of 15 km to 30 km in a city, where the receiver was located in a 24 m tall tower and the transmitter was on the roof of a car, and in a boat on open water. Our own experiments with the NetBlocks XRange (see section 3.5.1) show a range of 2.6 km in rural areas. From our studies [BVR16] in built-up environments we deduce a range of 100 m. This is significantly less than other reported ranges, probably caused by less than ideal indoor deployment, hardware, and antennas, and as such represents a worst-case deployment. We also performed all the simulations using parameters reported by Petäjäjärvi et al. [Pet+15], and obtained similar results in terms of scalability.

Obviously, the communication range and, hence, the exact path loss model is highly dependant on the environment and a generic figure cannot be given. To determine the parameters for our environment, we combined a NetBlocks XRange node with a MediaTek LinkIt ONE [Med16]. The MediaTek LinkIt ONE is an Arduino compatible development platform, powered by a 260 MHz MT2502A ARM7EJ-S MCU, 16 MB Flash, 4 MB RAM, and with support for Wi-Fi, Bluetooth, GSM (2G), and GPS.

One transmitter node was placed in an office on the third floor near the windowsill, broadcasting packets with a 3 s inter-packet spacing at SF12, BW125, and CR4. The receiver node was walked around campus. On the receiver node, the XRange recorded each received packet together with the RSSI and SNR, and send this information via the serial port to the LinkIt ONE. The LinkIt ONE recorded this data to the internal storage, together with the current time and GPS coordinates. From these empirical measurements, with d_0 set at 40 m, we determined that in the built up environment $\overline{L_{pl}}(d_0)$ is 127.41 dB, γ is 2.08 and σ is 3.57. We use these values in our simulation.

Sensitivity The sensitivity of a radio receiver at room temperature, as found in [13], is given by:

$$S = -174 + 10\log_{10}(\mathrm{BW}) + \mathrm{NF} + \mathrm{SNR} \tag{4.4}$$

The first term describes thermal noise in 1 Hz of bandwidth and can only be influenced by changing the temperature of the receiver. BW is the receiver bandwidth. NF is the receiver noise figure, and fixed for a given hardware implementation. SNR is the Signal-to-Noise Ratio (SNR) required by the underlying modulation scheme, and is determined by the spreading factor SF. The higher the SF, the higher the SNR.

As BW is set in steps of powers of 2, we can derive from eq. (4.4) that increasing the bandwidth decreases the sensitivity by 3 dB and *vice versa*. Similar for SF, increasing the spreading factor doubles the chips per symbol, which increases the sensitivity by 3 dB.

To determine the receiver sensitivity for our experimental platform, we carry out an experiment using two NetBlocks XRange nodes. Both nodes are placed in different rooms on different floors of an office building, to get maximum attenuation. The distance between the nodes is two floors, and approximately 40 m. One node transmits a fixed number of packets, on all combinations of spreading factor (SF7 to SF12), bandwidth (125 kHz, 250 kHz, 500 kHz), coding rates (CR 4/5, CR 4/6, CR 4/7 and CR 4/8) and transmit powers (2 dBm to 17 dBm). We repeat the measurement over several days and of all the correctly received packets we record the minimal RSSI to determine the sensitivity. The results are shown in table 4.1.

As expected, decreasing the bandwidth or increasing the spreading factor does improve sensitivity. The difference between each step, however, is not 3 dB, but more in the range of 0 dB to 4 dB, and 2 dB on average. Presumably this is caused by external interference, and hardware limitations other than the radio chip itself. We use these experimental determined values in our simulations.

Table 4.1: Measured receiver sensitivity in dBm for different bandwidths and spreading factors.

	Bandwidth (kHz)		
SF	125	250	500
7	-126.50	-124.25	-120.75
8	-127.25	-126.75	-124.00
9	-131.25	-128.25	-127.50
10	-132.75	-130.25	-128.75
11	-134.50	-132.75	-128.75
12	-133.25	-132.25	-132.25

Summary

Using eqs. (4.2) to (4.4) we can now estimate if a LoRa transmission will be received or not. The decision regarding transmission reception can be formally described as:

$$R = \begin{cases} 1, & P_{rx} > S_{rx} \\ 0, & else \end{cases} \tag{4.5}$$

To determine P_{rx}, the parameters $\overline{L_{pl}}$, d_0, γ and σ must be set to parametrise the path loss model and the communication distance d must be known. In our simulations we set these parameters to the values previously described to reflect a built up environment. S_{rx} depends on the selected BW and SF. We use the measured sensitivity as shown in table 4.1 in our simulations to determine sensitivity in dependence of BW and SF.

4.2.2 Collision Behaviour

When two LoRa transmissions overlap at the receiver, there are several conditions which determine whether the receiver can decode, one or two packets, or nothing at all. These conditions are Carrier Frequency (CF), Spreading Factor (SF), power, and timing.

Reception Overlap

Packet reception starts at time a and ends at time b. We define reception interval (a_i, b_i) for packet $i \in \mathbb{N}$, that is reception i starts at a_i and ends at b_i. We define the midpoint $m_i = \frac{a_i+b_i}{2}$ and midpoint length $d_i = \frac{b_i-a_i}{2}$. Two packets, x and y, overlap when their reception intervals overlap, that is:

$$O(x, y) = |m_x - m_y| < d_x + d_y \tag{4.6}$$

Carrier Frequency

When two transmissions overlap in time, but not in Carrier Frequency (CF), they do not interfere with each other and can both be decoded (assuming a receiver is listening at both carrier frequencies). The overlap in CF is defined as the absolute difference of these frequencies, and the tolerable frequency offset, which depends on the bandwidth. Therefore, we can define the condition when two transmissions collide on CF C_{freq} as:

$$C_{freq}(x, y) = \begin{cases} 1 & \text{if } |f_x - f_y| < f_{threshold} \\ 0 & \text{else} \end{cases} \tag{4.7}$$

where f_x and f_y are the centre frequencies of transmission x and y, and $f_{threshold}$ is the minimum tolerable frequency offset. The minimum tolerable frequency offset for the Semtech SX1272 is 60 kHz for a bandwidth of 125 kHz, 120 kHz for a bandwidth of 250 kHz and 240 kHz for a bandwidth of 500 kHz.

Spreading Factor

The SF used in LoRa are orthogonal. Transmissions with different SF (and same CF and BW) can thus be successfully decoded (assuming two available receive paths).

Therefore, we define the condition on when two receptions collide on SF C_{sf} as:

$$C_{sf} = \begin{cases} 1 & \text{if } SF_x = SF_y \\ 0 & \text{else} \end{cases} \tag{4.8}$$

where SF_x and SF_y are the SF of transmission x and y.

Power

As LoRa is a form of frequency modulation, it exhibits the *capture effect*, as demonstrated in section 3.5.3. The capture effect occurs when two signals are present at the receiver and the weaker signal is suppressed by the stronger signal. The difference in received signal strength can therefore be relatively small. When the difference is too small, however, the receiver keeps switching between the two signals, effectively not able to decode either transmission. Therefore, we can define the condition on when packet x collides with packet y on received signal strength as:

$$C_{pwr}(x, y) = \begin{cases} 1 & \text{if } (P_x - P_y) < P_{threshold} \\ 0 & \text{else} \end{cases} \tag{4.9}$$

where P_x is the received signal strength of transmission x and P_y is the received signal strength of transmission y and $P_{threshold}$ is the power threshold.

Timing

While the capture effect helps in decoding strong packets, as shown in section 3.5.3 there is a timing component to whether a overlapping transmission can be successfully decoded.

Figure 4.1 shows another result of the experiment performed in section 3.5.3 for SF12 and BW250. From this figure we can see that a strong transmission can be successfully decoded when it arrives one packet time early up to at most 3 symbols late, successfully suppressing the weak transmission. However, with an offset of more than

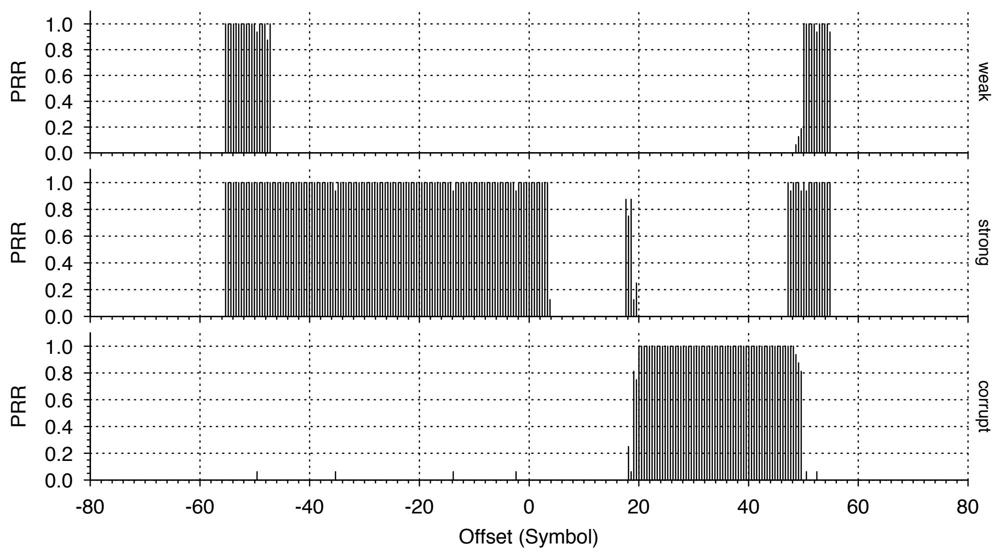


Figure 4.1: Capture effect. SF = 12, BW = 250 kHz, 55.25 symbols packet length. X-axis shows the transmission offset relative to the weak node in symbols, Y-axis shows the packet reception rate.

+3 symbols up to the end of the packet, no transmission gets through. The receiver requires 5 symbols to detect the preamble and synchronise. The transmissions were sent with 8 preamble symbols. Therefore, after 3 symbols, the receiver has locked on to the weak transmission, but its signal is suppressed by the strong transmission and the packet is corrupted.

From this result, we can conclude that packets can overlap, as long there are at least 5 preamble symbols left intact (in case of a *weak* packet). In other words, the critical section of a packet reception starts at the last 5 preamble symbols, so we can redefine the interval for transmission x as $x_{cs} = (a_x + T_{sym} \cdot (N_{pp} - 5), b_x)$, where T_{sym} is the symbol time and N_{pp} is the number of programmed preamble symbols. Therefore, packet x collides with packet y when it overlaps in its critical section x_{cs}:

$$C_{cs}(x, y) = \begin{cases} 1 & \text{if } O(x_{cs}, y) \\ 0 & \text{else} \end{cases} \tag{4.10}$$

Summary

When all conditions as defined in eqs. (4.6) to (4.10) are true, then packet x and y collide:

$$\begin{aligned} C(x,y) = O(x,y) \wedge C_{freq}(x,y) \wedge C_{sf}(x,y) \\ \wedge C_{pwr}(x,y) \wedge C_{cs}(x,y) \end{aligned} \tag{4.11}$$

We use this model of collision behaviour in our simulations.

4.3 LoRaSim

We use a simulator to examine and understand scalability of LoRa networks. It is not feasible to evaluate scalability of large-scale LoRa networks in practice as the deployment of such networks would be prohibitively expensive. Furthermore, a real deployment would not allow us to test a larger number of configurations and topologies as is needed for a general study on scalability. However, to ensure our results are of practical relevance we use the aforementioned practical experiments to calibrate our simulation.

4.3.1 Simulation Framework

For the purpose of this study we developed the simulation tool *LoRaSim*[2], which is a custom-build discrete-event simulator that uses the SimPy [Lün+18] framework. LoRaSim allows us to place N LoRa nodes in a 2-dimensional space (grid layout or random distribution). M LoRa sinks (the data collection points) can also be placed within the space.

Each LoRa node has a specific communication characteristic defined by the transmission parameters TP, CF, SF, BW and CR. For an experiment, each node's transmission behaviour is described by the average packet transmission rate λ and packet

[2] Available at http://www.lancaster.ac.uk/scc/sites/lora/

payload B. We assume a preamble length of 8 symbols, so packet airtime for a packet is given by B, SF, BW and CR. The behaviour of a node n during a simulation run is therefore described by the set $SN_n = \{TP, CF, SF, BW, CR, \lambda, B\}$.

Each LoRa sink is able to receive for a given CF multiple signals with different SF and BW combinations. This mimics the behaviour of LoRa gateway chips such as the Semtech SX1301 that can receive 8 concurrent signals as long as these signals are orthogonal (*i.e.* as they are using different SF or BW settings). Two of such chips can be used in a sink node to ensure that concurrent signals on all orthogonal SF and BW settings can be received simultaneously.

The communication behaviour of LoRa nodes can be modelled using the equations for communication range (eq. (4.5)) and collision behaviour (eq. (4.11)). However, the simulator has the ability to replace both models with a simplified variant. The simple variant assumes infinite communication range and any two transmissions overlapping in time at the receiver with the same CF, SF and BW will collide and none of the two transmissions is received. The simple models allows us to establish a baseline that can be analytically described (see Experiment 1).

4.4 Evaluation

4.4.1 Metrics

To evaluate scalability and performance of LoRa deployments we define two metrics: Data Extraction Rate (DER) and Network Energy Consumption (NEC).

DER

In an effective LoRa deployment all transmitted messages should be received by the backend system. This means that each transmitted message should be received correctly by at least one LoRa sink. We define the Data Extraction Rate (DER) as the ratio of received messages to transmitted messages over a period of time. The achievable DER depends on the position, number, and behaviour of LoRa nodes and sinks which is defined by N, M and SN. DER is a value between 0 and 1; the closer the value is to 1 the more effective the LoRa deployment is. In a perfect deployment one would expect DER $= 1$. The metric does not capture individual node performance and is a metric looking at the network deployment as a whole.

NEC

The energy consumption of a LoRa node will depend in most scenarios mainly on the energy consumption of the transceiver. As nodes will be deployed in many scenarios on batteries it is essential to keep energy consumption for transmissions to a minimum. Transmit energy consumption for each message depends on transmit power TP and transmission duration which is influenced by SF, BW and CR. We define Network Energy Consumption (NEC) as the energy spent by the network to successfully extract a message. The NEC depends on the number of nodes, frequency of transmissions, and transmitter communication parameters. The lower the metric, the more efficient is the deployment as lifetime of nodes is longer. The energy required to extract a message should be independent of the number of nodes deployed in the network. Again, the metric does not capture individual node behaviour and is a metric looking at the network deployment as a whole.

4.4.2 Experimental Evaluation

Experiment Set 1 — Single Sink

In our first set of experiments we evaluate the principle capability of LoRa using a simple setup where N nodes transmit to one sink ($M = 1$). We use in these experiments homogeneous transmitter configurations; for an experiment run all nodes use the same configuration set $SN = \{TP, CF, SF, BW, CR, \lambda, B\}$. Nodes are placed

Table 4.2: Parameter setting for Experiment Set 1.

	Set		
Parameter	SN^1	SN^2	SN^3
TP (dBm)	14	14	14
CF (MHz)	868	868	868
SF	12	6	12
BW (kHz)	125	500	125
CR	$4/8$	$4/5$	$4/5$
λ (ms)	1×10^{-6}	1×10^{-6}	1×10^{-6}
B (byte)	20	20	20

randomly around the sink such that all nodes can reach the sink with the given setting SN.

We compare the three transmitter configurations SN^1 and SN^2 and SN^3 (see table 4.2). In all settings we assume a 20 B packet is sent by each node every 16.7 min representing a realistic application. With SN^1 we choose the most robust LoRa transmitter settings leading to transmissions with the longest possible airtime of 1712.13 ms. With SN^2 we choose the transmission setting that leads to the shortest airtime of 7.07 ms. With SN^3 we choose the setting use by common LoRaWAN deployments as, for example, one trialled in Amsterdam[3]. We use SN^1 with simple channel models and our LoRa channel models to analyse the impact of these more realistic channel representations. For all subsequent experiments we use the LoRa channel representation.

Figure 4.2 shows the result of our first set of experiments. Each data point represents the mean of 30 simulation runs. For each run, the nodes are randomly placed around the sink, and simulated for approximately 58 days. The standard deviation was at most 7.609×10^{-3}, and is therefore not visible in the plot. With an increasing number of nodes the DER drops exponentially in all cases. The difference in DER is significant when comparing the configuration with longest (SN^1) and shortest air-

[3]https://thethingsnetwork.org

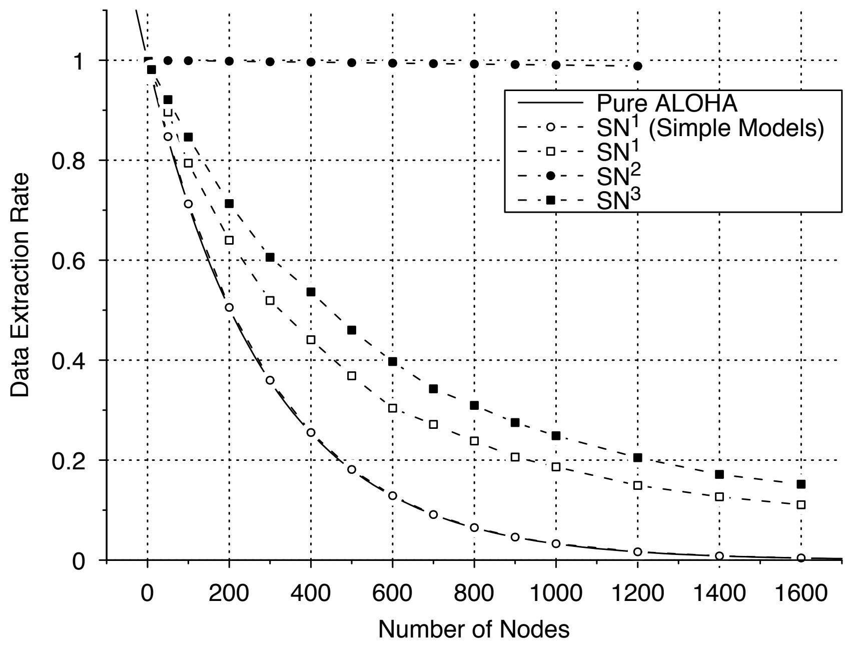


Figure 4.2: Experiment Set 1 – Single Sink: Pure ALOHA and SN^1 (Simple Models) overlap. As the number of nodes increases, the DER decreases exponentially. With typical LoRaWAN settings (SN^3) and a typical DER > 0.9 requirement $N = 120$ nodes can be supported.

time (SN^2). The default LoRaWAN configuration (SN^3) is very close to the configuration with the longest airtime (SN^1). We also observe a significant difference between using simple channel models (SN^1 Simple Models) and the LoRa channel representation (SN^1).

If we would assume that an application requires a DER > 0.9 to provide useful functionality, we would be able to support $N = 64$ nodes with the default LoRaWAN configuration (SN^3). The modelled communication range here is around 100 m (as observed in our experiments in a built up environment) and we can see that many applications (such as building automation) could not be supported by a LoRa system. It is likely that in such scenarios more nodes would have to be supported within the given range of a sink.

Obviously one could use less conservative transmission settings (the extreme represented by SN^2) to accommodate more nodes. However, in this case the transmission range is reduced and little protection against burst interference is provided. For example, the average transmission range for SN^1 is 98 m compared to 37 m for SN^2.

If we assume our deployment is located in Europe the regulator would require that each node can only use the channel for 0.1% of the time (duty-cycle limitation). For our experiment using the default LoRaWAN configuration (SN^3) we would obtain a channel duty-cycle of 0.13% that is above the regulator allowance. To comply we would need to reduce the data transmission rate from one 20 B packet every 16.7 min to every 22 min.

For SN^1 fig. 4.2 shows results using simple channel models and LoRa models. The more realistic channel representation leads to an increase in DER as colliding transmissions may still be received due to the capture effect. For example, for $N = 200$ the DER increases from 0.51 to 0.64. This effect is significant and cannot be neglected when analysing the capacity of a LoRa network.

The setup with simple channel models corresponds to Pure ALOHA [TW11]. The DER for such systems is:

$$\text{DER} = e^{(-2N \cdot T_{packet} \cdot \lambda)} \tag{4.12}$$

where N is the number of transmitters, T_{packet} the packet airtime and λ is the transmission rate of all nodes. Figure 4.2 shows for SN^1 simulation results together with the analytic solution that match closely. This analytic solution can be used to describe the DER worst-case bound. More realistic channel models always result in a performance boost due to the capture effect.

Equation (4.12) implies a lower DER for larger packets and higher transmission rates. We have verified that this is indeed the case, also in the more complex simulations such as those with multiple sinks.

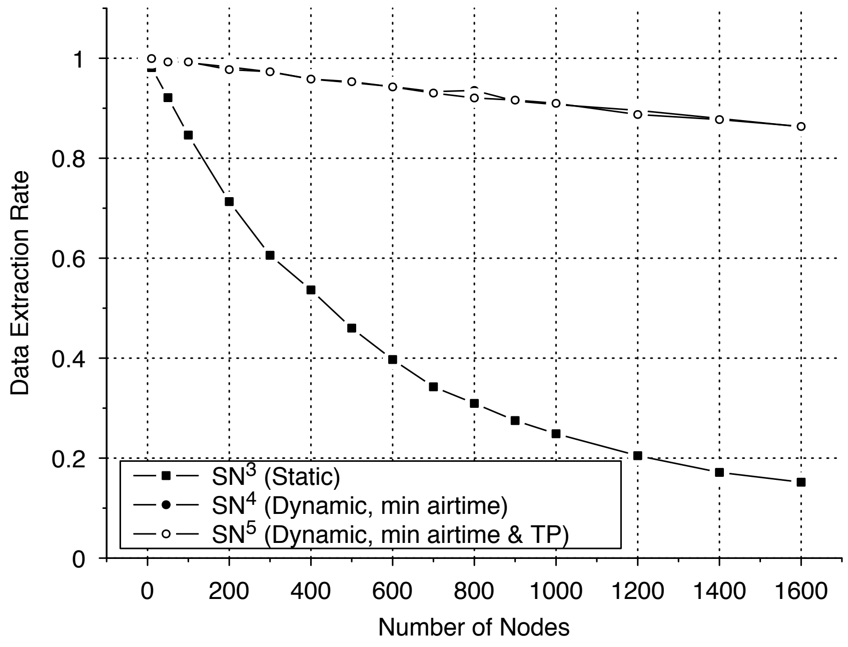


Figure 4.3: Experiment Set 2 — Dynamic Parameters: Lines for SN^4 and SN^5 overlap. When optimising transmission parameters for minimal airtime (or airtime and power) network capacity greatly improves. With minimised airtime (SN^4) and DER > 0.9, well over $N = 1100$ nodes can be supported (compared to $N = 64$ nodes with static settings).

Experiment Set 2 — Dynamic Parameters

In the second set of experiments we evaluate the impact of dynamic communication parameter selection on DER and NEC. We compare three transmitter configurations SN^3, SN^4 and SN^5. SN^3 is the same as in Experiment Set 1 and is used as reference. For all settings we assume again a 20 B packet is sent by each node every 16.7 min and CF is 868 MHz. N nodes transmit to a single sink ($M = 1$). Nodes are placed randomly around the sink within a radius that ensures that all nodes can reach the sink if they use the most robust settings. However, for each node the BW, SF, CR are set such that airtime is minimised (setting SN^4 with constant $TP = 14$ dBm) and then such that first airtime and then TP is minimised (setting SN^5). For all experiments we use the LoRa channel representation. Like in the first experiment sent, simulations were run 30 times, simulated for approximately 58 days. The standard deviation for DER was at most 6.270×10^{-3}, and therefore not visible on the plot.

As shown in fig. 4.3 the optimal allocated settings in terms of airtime (and airtime plus TP) has a huge impact on achievable DER. With minimised airtime (SN^4) and a DER > 0.9 requirement well over $N = 1100$ nodes can be supported. This is a dramatic improvement compared to $N = 64$ nodes achieved with static conservative settings as used in LoRaWAN.

However, it has to be considered that this achievement is not practical and relies on quite optimistic assumptions. First, the simulation does not consider external interference and the minimum airtime setting has a low CR setting which may not provide sufficient protection for this effect. Second, the minimum setting would need to be re-evaluated from time to time due to environmental changes. A protocol would need to be used in the LoRa network to determine and adjust the settings. Although LoRaWAN specifies a *Network Manager* component to specifically deal with this issue the implementation and its protocols are not yet defined. Thus, existing LoRaWAN deployments use static and conservative transmission settings represented by SN^3.

The impact on DER by minimising the TP (SN^5) as compared to only minimising the airtime (SN^4) is minimal. The reason for this is that by minimising the airtime, we already get on close to the receiver sensitivity. There is not enough margin left to also reduce the TP.

Figure 4.4 shows the impact of optimal allocated settings on NEC. Obviously, choosing settings with shorter airtime and less TP will not only help to improve DER but helps to achieve significant energy savings. For example, for $N = 200$ energy consumption in the network is reduced by 90%. This in turn translates to a proportional longer node lifetime if they operate on battery. Again, in practice these savings may only be achieved partially due to a lack of mechanisms for transmission setting adaptation and due to other constraints such as interference.

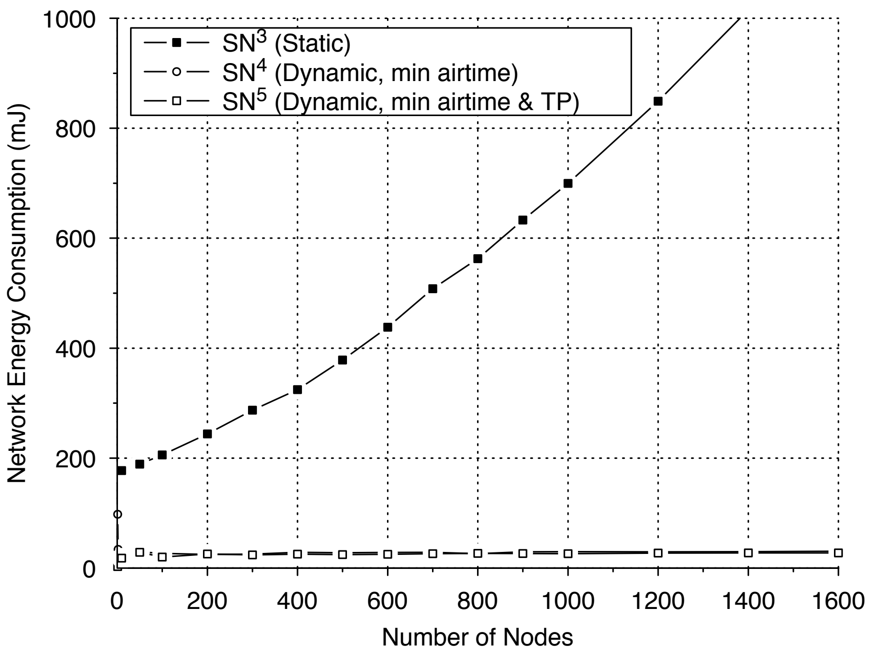


Figure 4.4: Experiment Set 2 – Dynamic Parameters: Lines for SN^4 and SN^5 overlap. Choosing communication parameters of nodes to minimise airtime (or airtime and power) has a significant impact on energy per extracted packet.

As with DER, minimising the TP has minimal impact on NEC. Firstly as there is barely any margin left after reducing airtime. Secondly, lowering the TP by one or two dB reduces the current consumption by a relatively small amount (see table 3.1). Minimising the airtime has a much larger impact, as each step in bitrate (by changing SF or BW) halves or doubles the airtime, and thereby halves or doubles the energy required to send a message.

Experiment 3 — Multiple Sinks

We have seen in the previous experiments that LoRa communication settings have a huge impact on network performance. In this set of experiments we explore the impact of the number of sinks M.

We use the previously described setting SN^1 for each experimental run (a 20 B packet is sent by each node every 16.7 min and CF is 868 MHz). For each run an

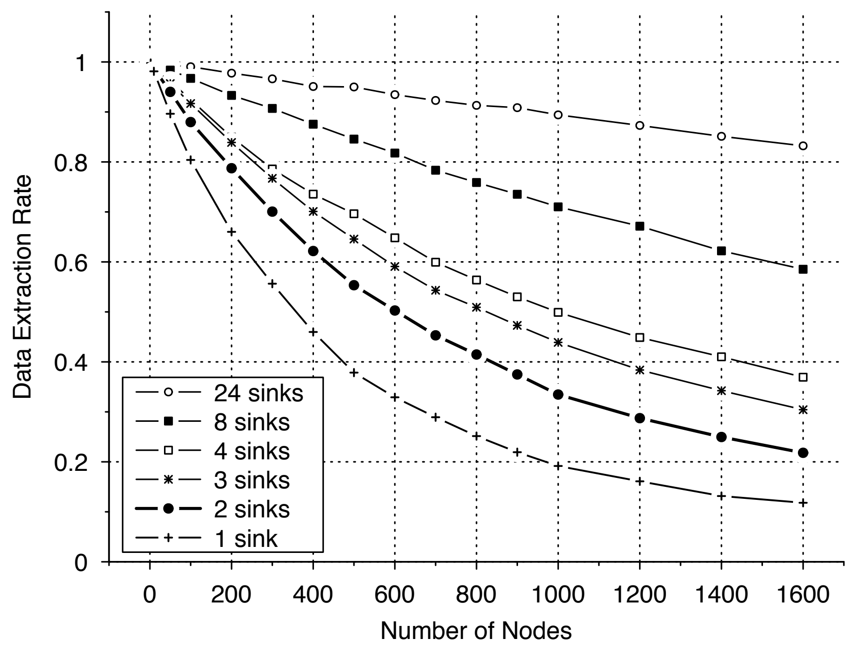


Figure 4.5: Experiment Set 3 — Multiple Sinks: multiple sink can significantly increase the DER.

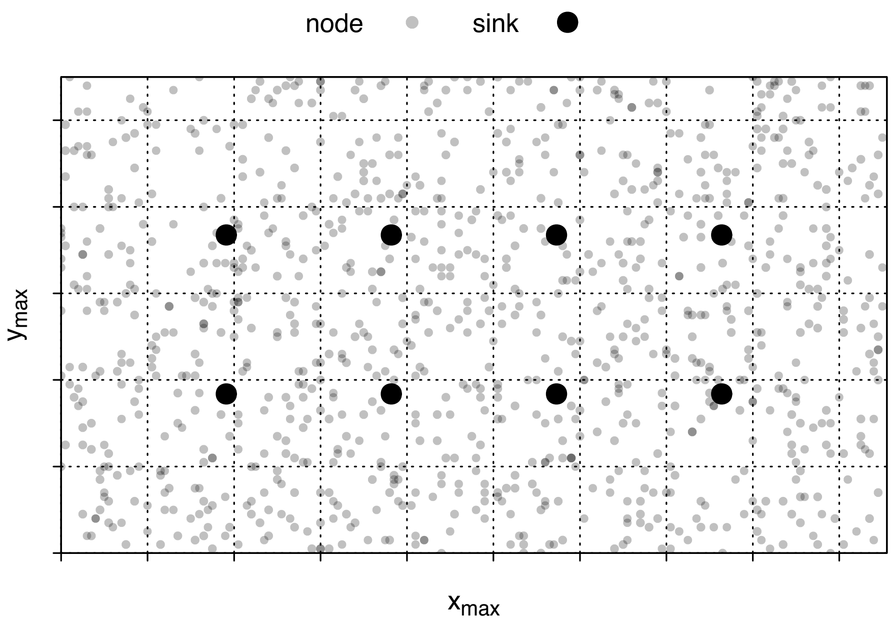


Figure 4.6: Example of a simulated deployment with 1000 nodes and 8 sinks.

increasing number of sinks M is used. The node placement strategy is changed as now multiple sinks are present. Nodes are placed in a rectangle with a diagonal twice the maximum calculated transmission range d_{max} (2.6 km for the most robust setting), and side lengths $x_{max} = \sqrt{3} \cdot d_{max}$ and $y_{max} = d_{max}$. This setup ensures that with communication settings SN^1 nodes within this rectangle can reach at least one sink. With four sinks or less, we space them equally over x_{max} on a straight line with $y = y_{max}/2$. Six or eight sinks are equally spaced over the two straight lines at $y = y_{max}/3$ and $y = 2 \cdot y_{max}/3$.

Twenty-four sinks are equally spaced over three straight lines at $y = y_{max}/4$ and $y = 2 \cdot y_{max}/4$ and $y = 3 \cdot y_{max}/4$.

Figure 4.6 shows an example deployment with 1000 nodes and eight sinks. Here the bottom left point is placed at $(x_{max}/5, y_{max}/3)$ while the top right point is placed at $(x_{max} \cdot 4/5, y_{max} \cdot 2/3)$.

We intentionally chose this sink placement strategy for simplicity rather than optimality. Simulations with different node placements have led to similar results.

Like with the previous experiments, we ran the simulator 30 times, for each set of nodes and sinks. The results are shown in fig. 4.5, whereby each data point represents the mean of those 30 runs. The maximum standard deviation was 7.100×10^{-3}, and is therefore not visible in the plot.

The results in fig. 4.5 depict that increasing the number of sinks significantly increases the DER. For example, with 200 nodes, one sink is not able to support the typical DER > 0.9 requirement while eight sinks achieve this. With 24 sinks, more than 1000 nodes still obey this requirement while with one sink the DER would be as low as 0.19.

Our expectation was that with an increase in the number of sinks the network would get saturated, and the DER would actually decrease. The figure, however, shows that this is not the case. We believe that this is caused by the fact that there only needs to be one sink where the capture effect comes into play in order to ensure that a packet can eventually be received. With more sinks, the chances increase that a packet finds a sink where the capture effect plays for its advantage. With an infinite number of sinks, each node might find such a sink avoiding packet loss.

4.4.3 Findings

Our experiments lead to a number of findings regarding the scalability of LoRa networks:

Lower-Bound on Performance

Pure ALOHA represents a good DER lower-bound in single sink deployments. Equation (4.12) can be used to quickly estimate expected performance of a typical LoRaWAN deployment.

LoRaWAN Scalability

With typical LoRaWAN settings (SF12, 125 kHz bandwidth, CR 4/5), the assumption of a 20 byte packet is sent by each node every 16.7 min and a DER > 0.9 requirement, $N = 64$ nodes can be supported. This is not a sufficient number for applications such as smart city deployments.

Dynamic LoRa Settings

Dynamic allocation of LoRa communication settings has a tremendous impact on network scalability. However, to make use of this potential gain protocols and mechanisms for dynamic parameter selection are required. In LoRaWAN the *Network Manager* is envisioned to fulfil this role but a specification is yet to be given.

Capture Effect

The capture effect has a significant impact on achievable DER. By far not all colliding transmissions are lost, in many situations at least one of the colliding transmissions can be received successfully. As this effect is significant it has to be taken into account when planning LoRa deployments. It also would have to be taken into account in simulation environments.

Multiple Sinks

Adding additional sinks to a deployment improves DER. We have not observed that there is an upper bound below 1 in terms of DER when adding additional sinks.

4.5 Conclusions

In this chapter, we have developed a model of a LoRa link based on real-world experimental results. With this model we build a simulator for a large scale LoRa that used to determine the bounds of scalability for a LoRa enabled network. We have shown that LoRaWAN as it is currently deployed will not scale well. We have shown that dynamic transmission parameter selection and the introduction of more sinks has a dramatic impact on scalability. However, we have not investigated which of the two strategies yields a better return. Deploying multiple sinks is costly and one has to find out where to deploy sinks best. Dynamic parameter selection requires implementation of complex protocols to facilitate this, increasing the risk of creating an unstable network. We explore the latter approach in chapter 5.

Chapter 5

Transmission Parameter Selection

LoRa has a wide range of link parameters that influence the link quality and energy consumption. For the best performance (QoS), a node or network manager needs to select the appropriate configuration. To save energy, we maximise data rate and minimise transmit power. Energy savings of well over 90% can be achieved in this way, while keeping the same good link quality. In addition, this leads to greater scalability of the network. A higher data rate means a shorter airtime, and thus more room for other nodes, as there is less chance of collisions. As a higher data rate is often realised by changing the spreading factor, this also helps with concurrent transmissions, as different spreading factors are orthogonal. Reducing the transmit power also minimises the collision domain.

This chapter analyses the properties of the LoRa link parameters, based on real-world experiments. Based on these results, we develop various algorithms for optimising a link. We analyse these algorithms based on metrics such as stability, energy consumed/saved, convergence speed, and approximate closeness to optimality.[1]

[1]This chapter is based on the paper 'LoRa Transmission Parameter Selection' [BR17].

5.1 Related Work

Transmission parameter selection, or Adaptive Data Rate (ADR), is an active area of research in the more established fields like Wi-Fi and 3GPP technologies like 3G, 4G, and 5G. Here the terms often used are link adaptation, (data) rate control, (transmission) power control or joint (transmission) power control and data rate control.

In Wi-Fi, Minstrel [MS09], and its successors Minstrel-HT [Fie10] and Minstrel-Blues [Hüh13], are widely deployed rate control algorithms on Linux-based computers, routers, and other wireless network equipment. Minstrel uses a *multi-rate retry chain*: an ordered list of four rate/retry pairs. A packet is first transmitted with the first rate for the specified number of retries. If all these attempts are not successful, it proceeds to the next rate/retry pair. When all pairs failed, the packet is discarded. The rate/retry pairs in the chain are picked based on the historic success rate and a throughput estimation. The algorithm adds occasional *probe frames* (10% of the time in Minstrel), to sample and evaluate unused data rates.

This approach is not suitable for our case, as Wi-Fi deals with a much higher traffic rate of several megabytes to gigabytes per second, compared to LoRa that handles traffic rates up to two 37-byte packets per hour. This limits the opportunity of trying many different rates, or probing the channel extensively. In addition, algorithms like those employed by Minstrel are optimising for throughput, and do not consider energy consumption. In our case, the power budget is severely limited, and a long operating lifetime is highly desirable. Therefore ADR should rather optimise for energy consumption, and not throughput.

In cellular communication, the goal of ADR is to minimise resource usage, so to allow as many users as possible, while maintaining the required QoS. Reducing energy consumption for the end nodes, so to maximise operating lifetime, is not an explicit goal [DPS14, chapter 6].

Historically, in CDMA-based networks as well as GSM-based networks, only dynamic power control has been used to compensate for variations in the communication channel. The transmit power is controlled such that the RSSI at the receiver is (near) constant, to minimise the error rate. The result is a constant data rate, which is a desirable property for services like circuit-switched voice.

With the advent of more packet-data traffic, a constant data rate became as less strong requirement, and rather, from a user perspective, the data rate should be "as high as possible". Instead of dynamic power control, the link adaptation focuses more on (only) dynamic rate control. It has been shown that rate control is more efficient than power control [CG01; GV97], so transmissions often happen at full power [DPS14, chapter 6].

Maximising operating lifetime by minimising energy consumption is arguably the most important issue in WSN research. Since the radio is the most power-hungry device, keeping it always on is not acceptable, so research has focused on keeping the radio off as much as possible, only waking up when it is absolutely necessary to receive or transmit data.

IEEE 802.15.4 is a communication standard commonly used in LPWAN networks like WSN. 802.15.4 has a fixed data rate depending on the modulation (*e.g.* 250 kbit/s for DSSS in 2.4 GHz band, 20 kbit/s for BPSK in 868 MHz band). Since there is a limited set of data rates, and commonly available radios often only support one modulation, dynamic rate control is therefore not considered. In contrast, LoRa offers a wide range of data rates, which gives the opportunity to finally control the rate.

With this lack of rate control, WSN research has focused mostly on dynamic power control, mostly in for example topology control [LLV13; San05], intelligent scheduling MAC protocols [YHE04; VL03; Dun11; Duj+14], and routing [Gna+09; PBD01; IGE00]. Since most LoRa networks are single-hop star networks, compared to the more common multi-hop network in WSN, topology control and routing are much less of a concern. In this work we therefore focus on optimising the energy efficiency of a link from node to gateway, instead of optimising the efficiency of the network as whole.

5.2 Transmission Parameters Selection

A LoRa link (see chapter 3) is defined by its Spreading Factor, Bandwidth, Coding Rate, and Transmission Power. The data rate (*i.e.* bit/s) is determined by the SF, BW, and CR. TP sets the transmission power. These settings determine the range, reliability, and energy consumption.

There are two main reasons to use ADR: energy efficiency and scalability. To maximise energy efficiency, an ADR algorithm tries to pick the setting with the lowest energy consumption, while still maintaining a reliable connection. Often this is done by maximising the data rate and minimising the transmit power, while staying within a certain link margin. This ensures we can send as much data as possible, expending as little energy as possible.

To illustrate the impact of different data rates in LoRa, fig. 5.1 shows the energy consumption for the transmission of a 32 B packet at a TP of 14 dBm. Each point in the graph represents a unique transmission parameter configuration (SF, BW and CR), for a total of 72 combinations. Depending on the setting, the energy consumption can vary from 2.20 mJ to 295 mJ, a factor of 134. An important aspect to consider is that by increasing the data rate, the energy consumption reduces exponentially. Also, it can be seen that several configurations lead to either similar or the same energy consumption, while their link quality may be vastly different.

The importance for optimising data rate to improve scalability has been shown in chapter 4. By optimising the data rate, the chance for collisions is reduced, improving the link reliability and capacity of the network. Also, picking a different spreading factor helps to increase the channel capacity, as spreading factors are orthogonal. On the other hand, increasing the data rate and reducing transmit power also reduces the

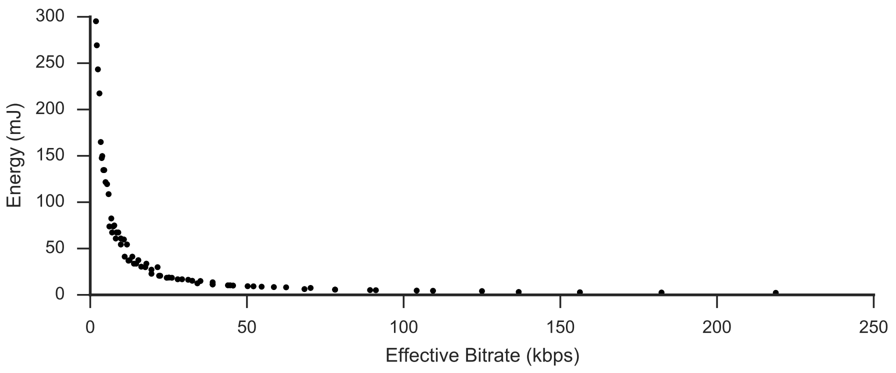


Figure 5.1: Effective data rate in kbit/s vs energy consumption in mJ for a packet with 32 B payload at 14 dBm.

range and link quality, as it becomes harder to overcome any interference. Therefore one should only increase the data rate until the required QoS is met, to overcome any interference.

Similarly, the data rate has an influence on the maximum distance of a transmission. As the distance increases, the receive power decreases exponentially, and it becomes harder to distinguish the transmission from noise. By lowering the data rate, it becomes much easier to decode a message on a noisy channel. Therefore, lower data rates have a higher transmission distance.

5.2.1 ADR Algorithm Outline

An ADR algorithm consists of two parts: a Link Quality Estimator (LQE) and a decision 'module'.

LQE is the process of estimating the quality of a link. Often this estimation is distilled in a single metric called Link Quality Indicator (LQI). The quality of link is influenced by *external interference* and *internal interference*. External interference is the interference caused by the physical environment, such as length of the path between sender and receiver (*path loss*), other radio sources, and physical objects and the environment itself on that path, causing effects like reflection, attenuation, shadowing, and multipath fading. This interference varies over time, by changing environmental circumstance (weather, air pressure, people moving). Therefore it is necessary to be able to determine whether the current settings meet the quality requirements, and how much more room for improvement there is, *i.e.* what the link margin is.

Internal interference is the interference caused by other nodes in the network, by colliding or attenuation with our transmissions. Since these nodes are under our control, the effect of this type of interference can be mitigated. This is one of the responsibilities of the MAC layer, to determine which node can send when on what condition. There is a wide-range of strategies, and many algorithms have been published over the years, each with their own trade-offs in terms of complexity, speed, and autonomous control.

To estimate the quality of a link, one could send a large amount of packets and count how many arrive. As this is quite energy inefficient, the challenge is to estimate the link quality from as little information as possible, minimising the amount of (additional) packets required.

When assessing the link quality based on received packets, it is important to determine whether reception errors come from physical interference, or from other nodes. If a particular link quality is considered to be too low due to physical interference, lower rates and higher transmission powers help in combating the physical interference. However, if the poor performance is caused by a high traffic load, a lower data rate would actually increase congestion, leading to more potential nodes in the collision range. A better strategy in this case would be to, counter-intuitively, increase the data rate, assuming there is enough link margin left. Therefore it is important to determine the source of an error, as changing the data rate may have an opposite effect to the one intended.

Once we have established the quality of a link, a decision module can decide what would be the next step: could there be a better, more energy efficient setting to use, should we stay with our current setting, or should we actually switch to a lower data rate, or increased transmit power. As LoRa offers a wide range of possible settings, whereby the difference between settings may be small, trying out every single setting would not be efficient. Therefore, a decision module has to make a 'smart' choice on what setting to pick. It can change the transmission power or data rate based on different criteria, like ordering the data rates and picking a faster data rate, or a lower transmission power, if there is enough link margin available.

An ADR algorithm generally has the choice of either only controlling the power, only controlling the rate, or do combination of the two. Each option has its own trade-offs in improving energy efficiency, but potentially also reducing the link reliability. An ADR optimisation algorithm therefore has to strike the balance between energy efficiency and reliability.

Power Control

Using only power control is the easiest option to implement, as it does not require any coordination between nodes and gateway. In a LoRa network, both sender and receiver need to be configured for the same data rate (SF and BW) for a transmission to be successfully received. When using a LoRaWAN compatible gateway, this can be mitigated, as it supports receiving a transmission on any SF as long as BW is 125 kHz. Only changing the transmit power does help in making a node more energy efficient, by ensuring it uses the minimal amount of energy required to reach the gateway. Power control does not increase the capacity of the network, compared to using different data rates, unless one considers the collision domain. To help a node in optimising its transmit power, gateways could return the Received Signal Strength (RSS) in ACKs, so nodes can adjust their transmit power based on the desired link margin, although the correlation between RSS and PRR is weak (see section 5.3.3). Gateways can also keep track of the PRR and the RSS of received packets, and instruct nodes to adjust their power, or let the node decide itself what to do. In this scheme, a node would be fixed to use the lowest data rate that guarantees a good link. Depending on the distance of the node to the gateway, some savings can be made in energy consumption, though more savings can be made if we also adjust the data rate.

Rate Control

Only changing the data rate is the most effective way to reduce energy consumption, especially in LoRa, as each step up (in either SF or BW) doubles the data rate, and therefore halves the airtime and energy consumption. It is harder to implement, however, as it requires coordination between the sender and receiver (*i.e.* node and gateway). Both parties need to be configured to use the same data rate (*i.e.* same SF and BW) to be able to receive messages. This can be mitigated by using a receiver with a LoRa baseband chip like the Semtech SX1301 (often used in LoRaWAN gateways), as these chips can receive transmissions on any SF as long as BW is 125 kHz.

Combining of Rate and Power Control

The most gains can be made by combining the data rate and power control, also known as joint power and rate control. Usually, rate control is applied first, and then power control is used if there is enough link margin left.

5.2.2 ADR in LoRaWAN

Parameters selection in LoRaWAN is referred to as ADR [Sor+17, section 4.3.1.1]. The LoRaWAN standard does not specify how exactly ADR is implemented, but only how a node should behave when it enables ADR. This gives the operator of a LoRaWAN network the opportunity to implement its own algorithm and provide it as a 'unique selling point'.

Whenever a LoRaWAN node wants its transmission parameters to be controlled, it sets the `ADR` bit in the frame header. Similar, the Network Server (NS) sets the `ADR` bit in its downlink packets, to indicate it is in a position to send ADR messages. The NS will disable the `ADR` bit in the frame header when it temporarily cannot control the nodes data rate, for example because of rapid changes in the channel. Via the `LinkADRReq` MAC command, the NS can control the data rate, transmission power, and channels a node uses. To ensure a node stays connected to the network, it is required to count its uplink packets (`ADR_ACK_CNT`). Whenever a node receives any downlink message, this counter is reset. If a node does not receive any downlink message within `ADR_ACK_LIMIT` uplink packets, it enables the `ADRACKReq` bit. A NS is then required to send a (potentially empty) downlink message to the node within `ADR_ACK_DELAY` uplink transmissions. When a node does not receive a response within `ADR_ACK_LIMIT` + `ADR_ACK_DELAY` messages, it backs off by first step-wise increasing the transmit power to the maximum permitted, and then decreasing the data rate until the lowest setting is reached. This happens every `ADR_ACK_DELAY`. When a node is at its default (highest) transmit power and default (lowest) data rate, it enables all possible channels which may have been disabled by the ADR. Currently, for all regions, `ADR_ACK_LIMIT` is 64 and `ADR_ACK_DELAY` is 32 [LoR17b].

5.2.3 Limitations of this Study

In this study, we only optimise for a single link, from a node to a gateway. Given the complexity in parameter space, the interactions between parameters on the link quality, and since LoRa is usually single hop, we do not consider optimising for the whole network.

However, we acknowledge that it may well be that if all nodes selected the same best setting, overall performance will be worse because of congestion, collisions, and other internal interference. Selecting a sub-optimal solution (from the point of view of the node) may be better for the network as a whole (and the node in the long term). The interaction of transmission parameters optimisations by individual nodes on other nodes, and the network as a whole, is a most interesting area we want to explore in future work.

5.3 Link Quality Estimation – What is a Good Link?

As described in section 5.2.1, determining the quality of a link is an important part of any parameter selection algorithm. It informs us what the performance of the current link is, what the state of the channel is, and how much room for improvement there may be. So the first step in developing a ADR algorithm is to construct a reliable and accurate LQE. To evaluate and compare links, we define a number of metrics. From the real-world data we gather, we evaluate how links behave and change when the parameters change.

5.3.1 Link Metrics

To evaluate and compare links, we define the following link metrics.

Packet Reception Rate (PRR)

Packet Reception Rate (PRR) is the ratio of transmitted packets vs correctly received packets, *i.e.*:

$$\mathrm{PRR} = \frac{n_{rx}}{n_{tx}} \tag{5.1}$$

whereby n_{rx} is the number of correctly received packets and n_{tx} is the number of transmitted packets. The value varies between 0 and 1, whereby 0 indicates a non-functioning link, and 1 represents perfect reception. PRR is often used as the ground truth for quantifying link quality.

Packet Corruption Rate (PCR)

Packet Corruption Rate (PCR) is the ratio of corrupted packets vs received packets, *i.e.*:

$$\text{PCR} = \frac{n_{err}}{n_{rx}} \tag{5.2}$$

whereby n_{err} is the number of corrupted packets. The values range from 0 to 1, where 0 indicates no received packets were corrupt, while 1 indicates that all received packets were corrupt. PCR may be indicative of a failing link, as it is expected PCR will go up when a link starts to degrade.

Received Signal Strength Indicator (RSSI)

Received Signal Strength Indicator (RSSI) is the received signal strength of a particular packet in dBm. It may be indicative of link performance, as radios have a particular sensitivity threshold. Any packet with a RSSI below the sensitivity threshold cannot be decoded correctly and is lost.

Signal-to-Noise Ratio (SNR)

Signal-to-Noise Ratio (SNR) is the ratio between signal and noise in dB, *i.e.*:

$$\text{SNR}\,[\text{dB}] = \frac{P_{\text{signal}}}{P_{\text{noise}}} \tag{5.3}$$

whereby P_{signal} is the power of the receiving signal in dBm (*e.g.* RSSI) and P_{noise} is the power of the background noise in dB. SNR may be indicative of link performance and environmental circumstances, as lower SNR indicates a more 'noisy' channel, making it harder to decode packets. SNR correlates with RSSI. Note that, in LoRa, packets can still be successfully decoded even with a negative SNR, *i.e.* the packets receiving power is below the noise floor.

Expected Retransmissions (ETX)

Expected Retransmissions (ETX) is the number of expected retransmissions for a packet to be received without error, defined as:

$$\mathrm{ETX} = \frac{1}{\mathrm{PRR}} \tag{5.4}$$

The ETX varies between 1 and infinity, whereby 1 represents a perfect link and infinity a completely failing link.

Effective Bitrate (EBR)

Effective Bitrate (EBR) is the effective data rate in bit/s (as defined by [CBR17]):

$$\mathrm{EBR}\,[\mathrm{bit/s}] = \frac{\mathrm{BR}}{\mathrm{ETX}} = \mathrm{BR} \cdot \mathrm{PRR} \tag{5.5}$$

whereby BR is the net data rate in bit/s.

Effective Energy (EKB)

EKB is the effective energy consumed to send a kilobit of data in J/kbit (as defined by [CBR17]):

$$\mathrm{EKB}\,[\mathrm{J/kbit}] = \frac{P}{\mathrm{EBR}} \tag{5.6}$$

whereby P is the power consumption of the radio in Watts, and EBR is the effective data rate in kbit/s.

5.3.2 Performance Metrics for a Good Link

Typically, the performance of a *good link* is expressed as a lower bound for the PRR, *e.g.* at least 80% of the transmitted packets should arrive. This constraint ensures that a link has a particular delivery ratio.

However, EKB is a better metric to optimise for, as it takes energy consumption into consideration. It better shows how much energy we spend on effectively transferring data. When we minimise EKB, we may get a link with a low PRR, but high data rate. Since most applications are delay tolerant, a configuration with a low PRR is acceptable. EKB is therefore a better performance metric for optimising the life time of a node, as it ensures we get the most bits for our joule.

5.3.3 Experimental Exploration of Link Dynamics

To get a better insight into the link dynamics and the influence of the various parameters of a LoRa transmission, we set up an experiment to gather real-world link data. The goal of this experiment is to test out all the different configurations, and see how much of an effect they have on the performance of a link, and which metrics can provide a good indicator of link quality. The second goal is to determine what the temporal stability of a LoRa link is. This will tell us how often a LQE needs to be run, or for how long results remains valid.

Experimental Setup

The experimental setup consists of two NetBlocks XRange SX1272 LoRa RF nodes [Net] (see fig. 3.3), the same hardware we used for our feature evaluation of LoRa in chapter 3. One node is programmed as sender, the other as receiver. Both nodes are placed in an office building, about 50 m apart, separated by a number of walls and floors. The receiving node is located in an office on the third floor, the transmitting node is located in the basement, in a separate wing of the building. The sender transmits 255 packets for each transmission configuration. The receiver is connected to a computer, listening on the appropriate configuration, recording received packets, and coordinating the experiment by transmitting the configuration for the next round to the transmitter. The experiment cycles through all possible configurations, *i.e.* 1152 combinations of: SF (7 to 12), BW (125 kHz, 250 kHz, and 500 kHz), CR ($4/5$, $4/6$, $4/7$ and $4/8$) and TP (2 dBm to 17 dBm), with a packet size of either 8 B or 32 B, and a known payload of alternating ones and zeroes.

The nodes do not cycle through every single configuration the Semtech SX1272 LoRa radio chip supports, because of hardware limitations of the used platform. Since the antenna is connected to the `PA_BOOST` pin of the chip, the transmit power range is limited between 2 dBm to 17 dBm. The radio chip is capable of a transmit power as low as −1 dBm, but to use this the antenna needs to be connected to the `RFO` pin, either directly or via a RF switch. Similar, the radio chip is capable of boosting the transmit power to 20 dBm, but this requires increasing the limiter value of the over current protection, and limits the radio to a duty cycle of 1%. As we wanted to capture any short term fading, and transmissions in the 868 MHz band are limited to 14 dBm by European regulatory constraints, we limited ourselves to 17 dBm. Finally, we do not use SF6, as it requires changing the detection thresholds of the radio, and this setting is not supported by LoRaWAN.

The experiment is repeated over 12 separate days, during office hours, and out of office hours. An experiment run takes about 34 h to complete. Packets were sent without headers (implicit header mode) and without CRC, as packets with corrupted headers are silently dropped and we wanted to capture as many packets as possible. In total, 1.6 million packets were transmitted, of which 90.5% were received correctly, 6% were corrupt and 3.5% were lost. The reliability of the link is relatively high, as for LoRa a distance of 50 m is small, even without line of sight.

Observations

Figure 5.2 shows a scatter plot of mean RSSI vs PRR for all links on every day. From the scatter plot it seems like the PRR is fairly distributed over the space. When we look at the histogram of the PRR, we can clearly see the PRR of a vast majority of the links is either 1 or 0. The link quality distribution does not show a *grey zone* as is common in other low-power wireless links [ZG03; Sri+06; ZK04]. The plot also shows that there is no correlation between RSSI and PRR. A low RSSI does not mean a low PRR or *vice* *versa*. It can be said that a RSSI of more than −115 dBm will indicate a link with a high PRR, as has been shown previously [Bac+12]. A link with a RSSI below this threshold is not necessarily a bad link, as even some links with a PRR of 1 have an average RSSI of −130 dBm, while some links with a RSSI of −120 dBm have an PRR of 0. This lack of correlation implies that determining the LQI based on the RSSI will not give accurate results.

Figure 5.3 shows a scatter plot of the mean RSSI vs mean SNR for all links on every day. RSSI and SNR are correlated (lower SNR is lower RSSI), partially because the RSSI on the radio chip is corrected by the SNR. The positive linear correlation indicates that the noise fraction (P_{noise}) of the SNR stayed constant.

Figure 5.4 shows three heatmaps with the averaged results of the experiments. The x-axis shows the effective bitrate, the y-axis shows the TP. Each square on the heatmap corresponds to a unique configuration.

The heatmap shown on top depicts the PRR for each configuration. The field in the top-left corner, with SF12, BW125, CR4, and TP 17 dBm, is the most robust transmission configuration with a PRR of 1 (red colour). The configuration becomes less robust towards the right-hand side and the bottom-right corner (with SF7, BW500, CR1, and TP 2 dBm), which represent a configuration with the highest bitrate, lowest transmit power, but least robust transmission configuration. Consequently the PRR in this area is zero. As can be seen, PRR is not homogeneous distributed over the graph. For some configurations a low PRR is observed, which improves again when changing the configuration to a higher effective bitrate and sometimes lower TP. However, as expected, there is an area in the lower-right corner of the heatmap where PRR drops off sharply.

The heatmap shown on the bottom shows the energy consumption for each configuration. The field on the top-left corner (lowest data rate, highest transmit power) has the highest energy consumption per transmission of a 32 B packet (darkest col-

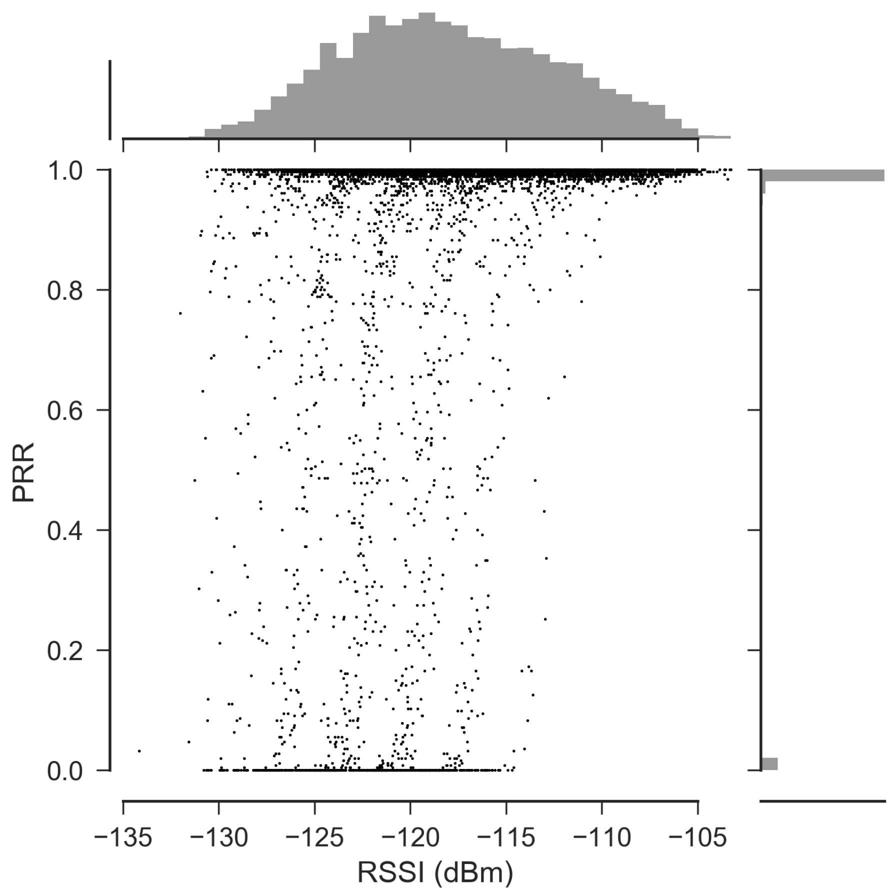


Figure 5.2: Scatter plot of mean RSSI vs PRR for all 1152 links. The plots on the axes shows a histogram of the RSSI on the x-axis and of the PRR on the y-axis.

our). The bottom-right configuration (highest data rate, lowest transmit power) has the lowest energy consumption per packet. It has to be noted that the energy consumption per transmission does not increase linearly from the top-right to the bottom-left corner.

The heatmap in the middle is a combination of the top (PRR) and bottom (energy consumption) graph. It shows the energy consumption of all configurations in which a PRR above a threshold of 0.9 is achieved. If we assume that applications can tolerate a link quality with a PRR of 90% or higher, the heatmap shows all potentially valid configurations that could be used for the link.

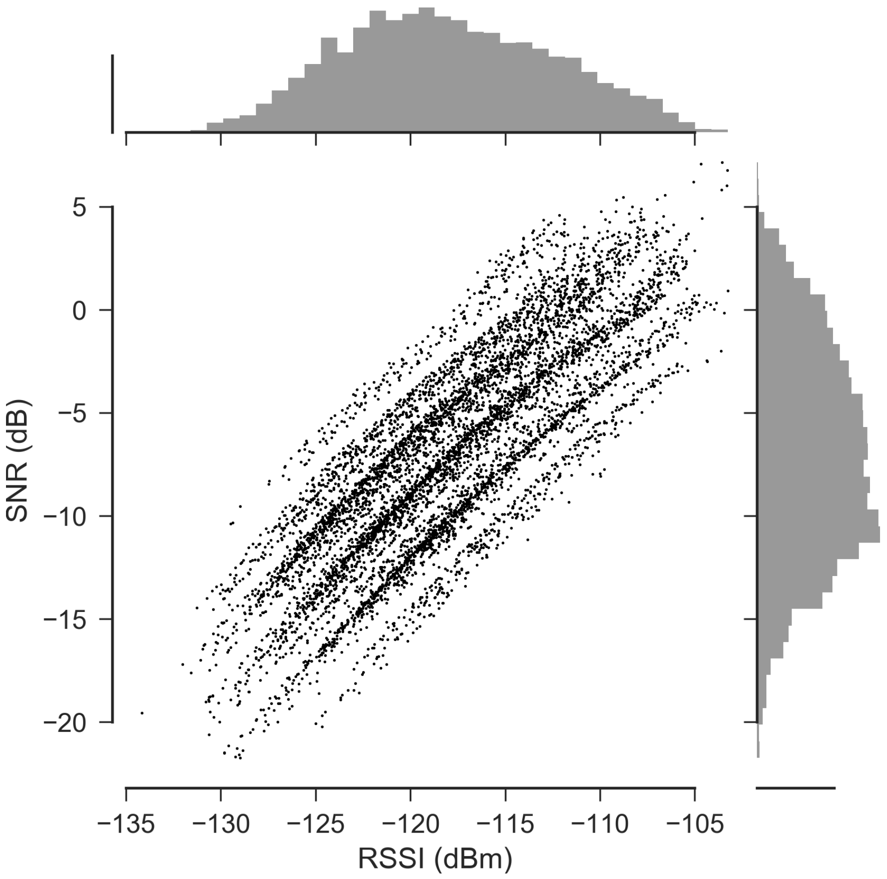


Figure 5.3: Scatter plot of mean RSSI vs mean SNR for all 1152 links. The plots on the axes shows a histogram of the RSSI on the x-axis and of the SNR on the y-axis.

Temporal Effects

As LoRa is wide band and CSS, links should be robust. In our experiments, we have not noticed significant differences over days. LoRa is not immune to temporal fluctuations, as there are fluctuations between time of day (daytime vs nighttime), and between office hours/out-of-office hours (weekdays vs weekends). The quality of the link for the same configuration at different times of the day, however, stays the same. That is, a good link stays a good link, and a bad link stays a bad link. However, because this is a small deployment, with static nodes, it is dangerous to generalise these results. In a real-world deployment, nodes may be mobile, and the environment may

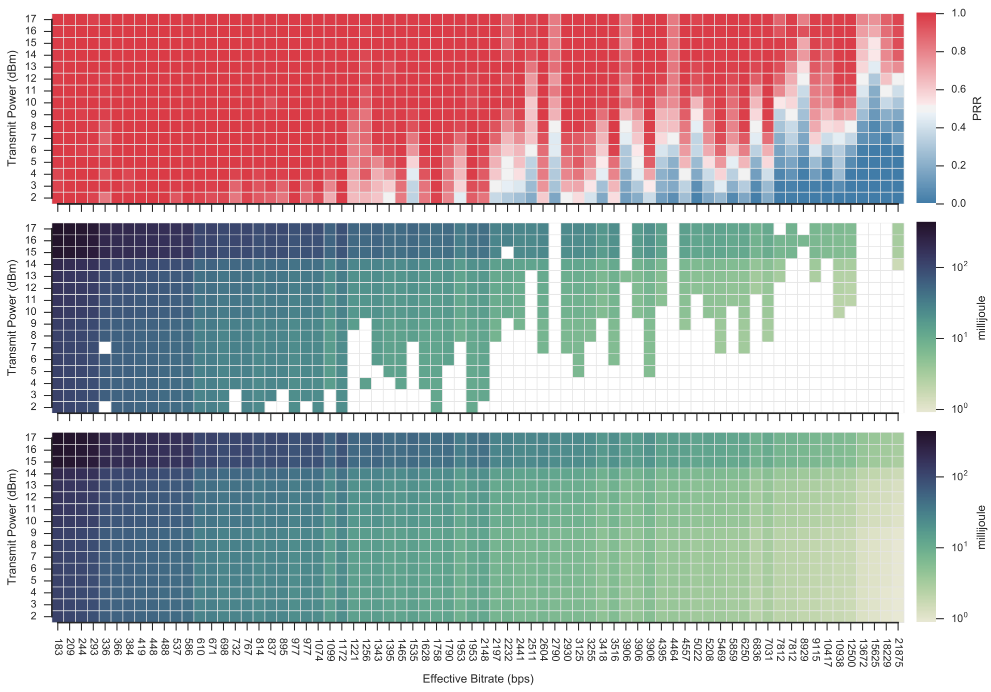


Figure 5.4: Heat map of transmit power and effective bitrate vs the PRR on top, and vs the energy consumption of transmitting a 32 B packet on the bottom. Note the logarithmic scale. The middle figure shows the energy consumption, whereby configurations with a PRR < 0.9 are filtered out.

have significant changes which affects the link quality. Thus it is advisable to regularly evaluate the link performance, and adjust the parameters accordingly.

5.4 Algorithms

In this section we discuss 4 algorithms: static, The Things Network (TTN), Probing and Optimistic Probing. Static is the easiest, but least flexible approach. TTN is the de facto standard, used in the crowdsourced TTN LoRaWAN network. Probing and Optimistic Probing are strategies we derived from our observations in section 5.3.3.

5.4.1 Static

The simplest solution for transmission parameter selection is to use a fixed configuration, for the lifetime of the node. The optimal configuration is the most conservative configuration (*i.e.* SF12 BW125), although it consumes the most energy.

To improve on this approach, one can pick a higher data rate and lower transmit power based on some heuristics, like distance to the gateway, or do some link measurement during deployment.

The static algorithm is the simplest to implement, and guarantees the most reliable link. Using this solution has two major downsides: there is a substantial energy waste, and it is not suitable for mobile nodes, or other fast changing environments. The optimal static configuration measured at deploy time may not be valid any more as the situation has changed dramatically.

5.4.2 TTN

TTN implements an algorithm [VST17] that is described in an unpublished Semtech document titled *Simple Rate Adaptation Recommended Algorithm* [16b]. When a node in a TTN LoRaWAN network enables the `ADR` bit, the NS starts collecting the SNR of the last 20 messages. Based on the maximum SNR (SNR_{max}) of the last 20 messages and the Data Rate (DR) of the last message, the link margin is calculated. The link margin $\text{SNR}_{\text{margin}}$ is defined as

$$\text{SNR}_{\text{margin}} = \text{SNR}_{\text{max}} - \text{SNR}_{\text{required}}(\text{DR}) - m \tag{5.7}$$

whereby $\text{SNR}_{\text{required}}(\text{DR})$ is the required SNR for the specified DR, as shown in table 5.1, and m is the base margin, which is 10 dB by default.

From $\text{SNR}_{\text{margin}}$, the number of steps is calculated as follows:

$$N_{\text{step}} = \left\| \frac{\text{SNR}_{\text{margin}}}{3} \right\| \tag{5.8}$$

If $N_{\text{step}} > 0$, DR is increased by N_{step} until DR6 is reached (see table 5.2 for LoRaWAN data rates as defined for Europe's 868 MHz band).

When, after increasing DR, there are still steps left, TP is decreased by 3 dBm until the minimum transmit power is reached (2 dBm for EU868).

On the other hand, if $N_{\text{step}} < 0$, TP is increased by 3 dBm for each step, until the maximum allowed transmit power is reached (14 dBm for EU868).

If for example $\text{SNR}_{\text{max}} = 5.5$ dB, and the current data rate is DR0 (250 bit/s), then

$$\text{SNR}_{\text{margin}} = 5.5 - (-20) - 10 = 15.5$$

From this, we can calculate N_{step} as

$$N_{\text{step}} = \left\| \frac{15.5}{3} \right\| = 5$$

Since $N_{\text{step}} > 0$, DR can be increased by 5 steps, so the new data rate is DR5 (5.47 kbit/s), and the end node will be instructed to use this new data rate.

The next time the node sets the `ADR` bit, and has for example the same $\text{SNR}_{\text{max}} =$ 5.5 dB, $\text{SNR}_{\text{margin}} = 5.5 - (-7.5) - 10 = 3$, so $N_{\text{step}} = \left\| \frac{3}{3} \right\| = 1$, and therefore the data rate will only be increased by 1, from DR5 to DR6 (11 kbit/s).

If the node has a higher SNR_{max}, for example 11.5 dB, $\text{SNR}_{\text{margin}} = 11.5-(-7.5)-10 = 9$, then $N_{\text{step}} = \left\| \frac{9}{3} \right\| = 3$. Since the maximum DR is DR6, the data rate can only be increased one step from DR5 to DR6. The two remaining steps are used to decrease the TP two times by 3 dBm, for a total of 6 dBm. If the node was transmitting at the default 14 dBm, its new transmit power would be 8 dBm.

It should be noted that the server-side ADR algorithm only increases the data rate. Whenever N_{step} becomes negative, only the TP increased by 3 dBm for each step, up until the maximum TP for the region the node is configured for. Any reduction in data rate is done by the backing-off mechanism specified in the LoRaWAN specification [Sor+17, section 4.3.1.1, page 20]. ADR controlled nodes are required to send packets with ACK every N packets (`ADR_ACK_LIMIT`, 64 for EU868), if this

DR	SNR (dB)
DR0	-20.0
DR1	-17.5
DR2	-15.0
DR3	-12.5
DR4	-10.0
DR5	-7.5
DR6	-4.5

Table 5.1: $\text{SNR}_{\text{required}}$ for each DR as defined for TTN's ADR.

DR	SF	BW (kHz)	bit rate (bit/s)
DR0	12	125	250
DR1	11	125	440
DR2	10	125	980
DR3	9	125	1760
DR4	8	125	3125
DR5	7	125	5470
DR6	7	250	11000

Table 5.2: LoRaWAN transmit data rate for the EU863-870 PHY layer [LoR17b].

fails (no response within `ADR_ACK_CNT` packets, 32 for EU868), the rate is decreased. In TTN's implementation, the NS only sends a `LinkAdrReq` when the node explicitly sets the `ADRACKReq` bit. Since nodes usually only send unconfirmed uplink messages, and the NS rarely sends downlink messages, the ADR is only re-evaluated every `ADR_ACK_LIMIT` uplinks.

With a node sending irregular sensor readings, it can therefore take quite some time before an initial optimisation attempt is made. This process can be sped up by enabling the `ADRACKReq` bit during booting, forcing the NS to send new configurations after 20 successful uplinks. Similar, a node could, at some additional energy cost, send empty uplink packets to speed up the process.

5.4.3 Probing

As noted in the previous section, the TTN algorithm is slow, especially when dealing with poor links. In addition, it only works for the limited set of transmission parameters defined for LoRaWAN, and it relies on calculating the link margin at the NS. From the observations in section 5.3.3 and from fig. 5.4, we can see that when the data rate increases, there is a border where the link quality drops sharply. An intuitive approach would be to try to approach this border, and walk along it, trying to find the optimal configuration.

One could start with the most robust, but most energy costly configuration, and then approach the border by picking the next configuration that is half the energy of the current one. If the new configuration does not meet the quality threshold, one could take a configuration with an energy cost between the last successful configuration and the current failing configuration. This process is basically a binary search over the entire configuration space.

The *Probing* LQE is based on this approach: it sends *probe packets*, to assess the link quality. This can be done either *actively*, whereby packets are sent solely for the purpose of assessing the channel, or *passively* whereby the delivery statistics of regular communication is used as an input. A passive probing approach seems more appealing, as there is no additional energy spent on maintaining the network, and only on useful information. However, as a node may report infrequently, the Probing algorithm may take quite some time to determine a good configuration. Therefore a more active probing approach, to quickly determine the optimal configuration may be more desirable, at some additional energy cost. The algorithm as described here would work with either approach.

Algorithm 1 describes the probing algorithm. A node starts with configuration S_m as the candidate configuration S_c. This can be any configuration, but especially on startup this would be the most robust configuration with highest transmit power

(SF12, TP17). The configuration S_c is evaluated by sending n probe packets, either actively or passively. If the PRR is above the threshold τ, the S_c is marked as 'good' and becomes the new active configuration S_m. A new candidate configuration S_c is selected from all the configurations (S) that have not been tried yet ($S \setminus V$), whereby the energy consumption is at least half that of the current configuration S_m. If, on the other hand, the PRR is below the threshold τ, the S_c configuration is marked as 'bad', and a new configuration is picked from all the possible configurations, which has an energy consumption halfway between the known good configuration S_m and the bad candidate S_c. The idea is that we apparently overstepped the border, and we use a binary search to find the limit. This process is repeated until there are no candidates left ($S \setminus V = \varnothing$).

Algorithm 1 requires a considerable number of probes. Using heuristics, the number of probes could be reduced, without impacting the accuracy of the LQE too much. For example, one way to reduce the number of probes is by realising that the PRR threshold can never be met, when more than $\lceil n \times (1 - \tau) \rceil$ probes are lost. It is therefore not useful to keep on sending probes, and the LQE can terminate early. Especially on bad channels, this heuristic can reduce the probing effort quite significantly. Another way to reduce the probing effort is by considering a channel to be good when the RSSI is higher than a particular threshold. From the results obtained in section 5.3.3, a link has a good PRR if the RSSI is above −105 dBm.

As the algorithm will run until all configurations have been tried, it can be quite costly to run. Therefore, it may be advantageous to set a limit on the number of iterations, or set an energy budget on what one can spend on probing.

5.4.4 Optimistic Probing

Even with the heuristic optimisations, the previous algorithm still requires a considerable amount of packets, around 50 to 100 for good link (we will explore this more

Algorithm 1 Probing Algorithm

```
 1: procedure PROBE($\tau$, $S_m$)          ▷ PRR threshold $\tau$ and initial configuration $S_m$
 2:     $V \leftarrow \emptyset$                          ▷ All tried configurations
 3:     $S_t \leftarrow S_m$                        ▷ Current candidate configuration
 4:     loop
 5:         $V \leftarrow V \cup \{S_t\}$
 6:         if LQE($S_t$) $> \tau$ then
 7:             $S_m \leftarrow S_t$                ▷ Candidate meets PRR threshold
 8:             $e \leftarrow \mathrm{E}(S_m)/2$       ▷ New threshold is 1/2 of current configuration
 9:         else
10:             $e \leftarrow (\mathrm{E}(S_m) + \mathrm{E}(S_t))/2$            ▷ PRR is below threshold
11:         end if
12:         $C \leftarrow \{x \mid x \in S, x \notin V, E(x) < e\}$
13:         if $C = \emptyset$ then
14:             return $S_m$
15:         else
16:             $S_t \leftarrow \max(\{\mathrm{E}(t) : t \in C\})$
17:         end if
18:     end loop
19: end procedure

20: procedure LQE($n$, $S_t$)                   ▷ $n$ probes and $S_t$ configuration
21:     $s \leftarrow 0$                           ▷ Successful transmit counter
22:     for $i \leftarrow 0, n$ do
23:         if TRANSMIT($S_t$) $= success$ then
24:             $s \leftarrow s + 1$
25:         end if
26:     end for
27:     return $s/n$
28: end procedure
```

in-depth in section 5.7.2), to determine the optimal configuration. In section 5.3.3 we concluded that a link is either ‘good’ or ‘bad’, and that there is no grey zone. Therefore, instead of trying to get as close as possible to a particular PRR threshold, we could also try to pick a ‘good’ link. We modify the probing LQE to a more optimistic variant. This optimistic probing algorithm is defined in algorithm 2. Instead of trying to estimate the PRR precisely, we assume a link is ‘good’ whenever a packet gets through. In the case we are ‘unlucky’, and the packet does not get through (the link is not perfect), we can send at most n more packets, whereby n can be fairly small (*e.g.* 3). This approach would improve both the energy consumption and speed of the algorithm, as we do not have to send a lot of probes, and the good configurations are selected quickly. The risk of this approach is the amount of false positives, *i.e.* bad links that are perceived as being good. In other words, the LQE is biased towards good links.

Algorithm 2 Optimistic Probing

```
 1: procedure PROBE($n$, $S_m$)      ▷ $n$ maximum probes and initial configuration $S_m$
 2:     $V \leftarrow \emptyset$                                        ▷ All tried configurations
 3:     $S_t \leftarrow S_m$                                    ▷ Current candidate configuration
 4:     loop
 5:         $V \leftarrow V \cup \{S_t\}$
 6:         if LQE($n, S_t$) then
 7:             $S_m \leftarrow S_t$                                ▷ Candidate is a good link
 8:             $e \leftarrow \mathrm{E}(S_m)/2$            ▷ New threshold is 1/2 of current configuration
 9:         else
10:             $e \leftarrow (\mathrm{E}(S_m) + \mathrm{E}(S_t))/2$                ▷ Candidate is a bad link
11:         end if
12:         $C \leftarrow \{x \mid x \in S, x \notin V, E(x) < e\}$
13:         if $C = \emptyset$ then
14:             return $S_m$
15:         else
16:             $S_t \leftarrow \max(\{\mathrm{E}(t) : t \in C\})$
17:         end if
18:     end loop
19: end procedure

20: procedure LQE($n$, $S_t$)                 ▷ $n$ maximum probes and $S_t$ configuration
21:     for $i \leftarrow 0, n$ do
22:         if TRANSMIT($S_t$) $= success$ then
23:             return $true$
24:         end if
25:     end for
26:     return $false$
27: end procedure
```

5.5 ADR Performance Metrics

To assess the performance of the ADR algorithms, we define the following four metrics: optimality, energy consumption, convergence speed, and stability.

5.5.1 Optimality

An ADR algorithm should optimise a link, which is the configuration that minimises energy consumption per data transfer, given the channel state. Ideally, to assess the performance of an ADR algorithm, we would want to see how close it gets to the optimal configuration. However, determining the optimal setting is hard, as this requires global knowledge of the channel state that varies over time. With the packet traces collected in section 5.6, however, we do have this global knowledge, and can calculate the optimal setting. The optimal setting we can determine from our data set is the lower bound for the ADR algorithms.

In case we do not have this information, we can compare it to the static ADR algorithm instead. The static ADR algorithm gives the upper bound, and how much the other ADR algorithms improve on this, but it does not give us an idea on how close we are the to the optimal, and how much more room there is for improvement.

To be able to compare the different configurations, we use EKB, which is the joule per kilobit of effective data transferred. We do need the PRR to calculate EKB. In a live system, where we do not have the PRR for every single setting, we cannot use EKB to evaluate the performance of an ADR algorithm. We could calculate the joule per kilobit for every setting, without taking the PRR, or effective bitrate in consideration. This would however bias the optimality towards the highest data rate and lowest transmit power, which could have a fairly low PRR. For example, in our experimental data, a setting of SF7, BW500, CR1, TP10, would give an energy consumption of 4.25 mJ/kbit, while it has a PRR of 0.85, and therefore an EKB of 5.00 mJ/kbit. A setting of SF7, BW500, CR1, TP13 would be much better, as it has a PRR of 0.98 and an

EKB of 4.90 mJ/kbit, although its energy consumption would be 4.80 mJ/kbit. Purely based on the energy consumption, the first configuration would be better than the second one. However, taking in consideration the actual performance, the second configuration is more energy efficient.

Using EKB does assume that the MAC protocol does take some effort in delivering packets, like the usage of ACKs or repeated transmissions. A MAC protocol that does not do this, but does want some form of guaranteed delivery would be better off in using the PRR as optimality metric. Or alternatively, add an extra constraint for the minimal PRR desired.

5.5.2 Energy Consumed

To get to the optimal setting, an ADR algorithm needs to send packets to assess the channel state and make a decision on what setting to move to. These *channel assessing packets*, or *probe packets*, can be either passive or active. *Passive probing* is where the channel state is assessed from the regular traffic a node produces. This means there is no additional cost in running the ADR algorithm. In the case of *active probing*, whereby a node actively sends packets to assess the channel state, running the ADR algorithm does come with additional energy cost.

The other way in which energy consumption of an ADR algorithm is calculated is from how quickly it approaches the optimal setting. Ideally an ADR algorithm should quickly move to the lower energy consuming configurations, instead of lingering on a high power setting for a long time. Similar to the optimality metric, this favours high date rate configurations with potentially low PRR, in this particular case we also favour aggressively moving in to the high data rate, low transmit power configurations, even though it may be too much.

To compare the energy consumption, we calculate the energy consumed for transmitting packets in joules from the start of the algorithm up until a stable state has been reached. For some algorithms (Probing and Optimistic Probing) this will be when the search space has been exhausted, for other algorithms (TTN) this will be the last time when the algorithm changed configurations after running for an extended period of time.

This metric favours algorithms that use the minimal number of (extra) packets and are more aggressive to move towards the optimal setting.

5.5.3 Convergence Speed

Similar to how much energy an ADR algorithm uses, it is also important to consider its convergence speed. That is, how long it takes before it approaches the optimal or stable setting. Convergence speed is related to energy consumed, in that a high convergence speed requires less transmitted packets, and thus less energy is consumed.

We define convergence speed as the total airtime of the packets transmitted from the start of the algorithm up until it reaches a stable state. The definition of *stable state* is the same as with the energy consumed; either the last time the algorithm changed configurations when running for an extended time, or when the search space has been exhausted.

This metric favours aggressive algorithms, that use the minimal amount of packets and quickly move towards high data rate (but not necessarily low transmit power).

5.5.4 Stability

Over time, the channel state changes, and the optimal configuration as found by the ADR algorithm may not be the best any more. Either our current configuration performs much worse than a potential better one, or there is more room for picking a better configuration.

An ADR algorithm should be able to detect the channel state has changed, and choose the appropriate course of action. It can either be that an ADR algorithm picks a slightly more conservative configuration, one that is not the best configuration at this moment, but that one is ideal for over a long period of time. In other words, once the ADR picks a configuration, there is no need to start probing again, or change the configuration, thereby wasting any energy saved from choosing this.

The more aggressive algorithms will get to a lower energy configuration fast, but may have to occasionally reprobe or change the configuration, wasting all the energy that was saved. How often this has to happen depends on the temporal stability of a LoRa link. From our experiments (see section 5.3.3), we did not notice any significant changes changes over time, and therefore it seems that the algorithms may not have to reprobe very often.

5.6 Gathering Real-World Link Data

To evaluate the different ADR algorithms, we will use a trace-based simulation. Instead of using the packet traces we gathered for the experiment described in section 5.3, we gathered more extensive real-world link data on the Campus Area Testbed Framework (CATF). Having an extensive packet trace allows us to evaluate ADR algorithms under the a wide range of circumstances, and fairly compare them.

5.6.1 Campus Area Testbed Framework

Campus Area Testbed Framework (CATF) is a small-scale 8 node temporary LoRa testbed, located at the south-end of the campus of Lancaster University, UK. The testbed consists of a number of boxes, called *CSI nodes*. A schematic overview of a CSI node is shown in fig. 5.5. Each box comprises of:

- A COTS x86-64 mini-PC in the form of an Intel NUC, which is connected to the development board and the wired campus network. It is used for access, logging, flashing, and control.

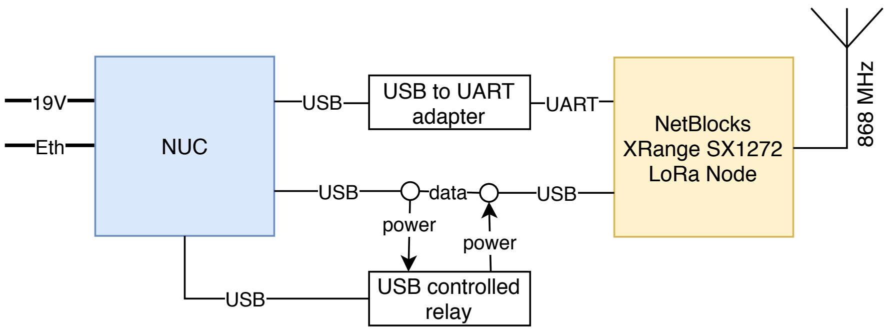


Figure 5.5: Schematic overview of a CSI node.

- A NetBlocks XRange SX1272 LoRa development board, which is connected via USB to the Intel NUC, and it is used for power and programming via Device Firmware Update (DFU). The `BOOT0` pin is tied to 3V pin, so on every power cycle, the node boots into the DFU boot loader for programming.
- A USB-to-serial adapter, wired to the hardware UART of the XRange and connected to the Intel NUC. The adapter is used for logging and controlling the firmware.
- A software-controlled USB relay, which is spliced with the USB connection that powers the XRange, so the relay can be used to power cycle the XRange.

The 8 CSI nodes are distributed through the south-end of the Lancaster University campus, as shown in fig. 5.6. All the nodes were placed indoors (as the boxes were not waterproof), near an electrical and Ethernet socket, for power and network connectivity. The nodes were placed preferably near window sills so they had a bigger chance of reaching another node in the network. Their exact location and description are shown in table 5.3.

Each CSI node sets up a reverse SSH tunnel to a central server, from which each node can be controlled and programmed. The LoRa nodes are programmed with a 'modem' firmware, which provides an easy-to-use AT-like interface via the USB serial port. This makes it easy to change parameters and transmissions on the fly, without

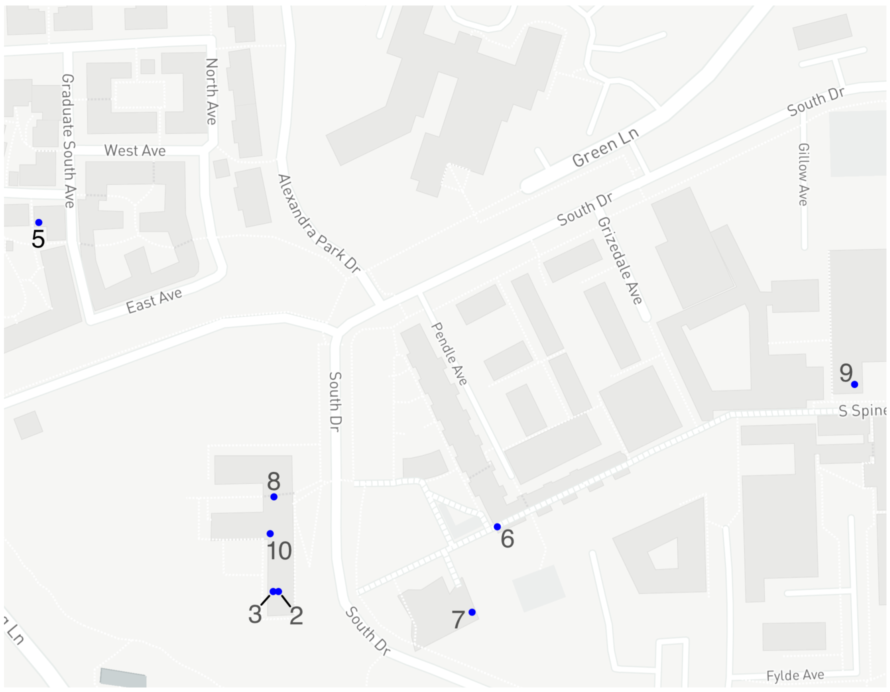


Figure 5.6: Map of the location of the CSI nodes, as indicated by the blue circles.

Table 5.3: Positions of the CSI nodes.

Node ID	Latitude	Longitude	Description
2	54°0′19.9″N	2°47′2.8″W	InfoLab21, D29, desk
3	54°0′19.8″N	2°47′2.8″W	InfoLab21, D29, cupboard
5	54°0′15.5″N	2°47′14.2″W	Grad college office
6	54°0′23.8″N	2°47′4.8″W	Pendle porters
7	54°0′23.4″N	2°47′2.2″W	ISS meeting room
8	54°0′19.8″N	2°47′5.7″W	InfoLab21 reception
9	54°0′30.3″N	2°47′9.2″W	Engineering, C09, Prof. M. Joyce's office
10	54°0′19.7″N	2°47′4.6″W	InfoLab21, D36, Prof. U. Roedig's office

the need for reprogramming the LoRa node every time.

5.6.2 Experimental Setup

We used CATF to collect detailed link information. Each node sends a salvo of three 64-byte packets back-to-back with a PRBS-9 sequence as payload, to mimic real LoRaWAN network data. The packets were send with explicit headers and CRC (see section 3.1.2).

After each salvo, the nodes in turns go through all the 1152 different configurations (SF7-12 BW125-250 CR1-4 TP2-17). For each transmitted packet and each received packet, the LoRa node sends a log message via the hardware UART, which is logged and timestamped by the NUC. For transmitted packets, the log message details the date and time of the start and end of the transmission (as an UNIX timestamp with microsecond precision), the configuration used (SF, BW, CR, and TP), the packet length, and the packet payload. For receptions, the log message details the date and time of the reception (as an UNIX timestamp with microsecond precision), configuration used (SF, BW, CR and TP), RSSI, SNR, CRC status (success or fail), and the packet payload. For each CSI node, these log messages are aggregated and stored in a SQLite database, whereby the receptions are correlated with the transmissions based on the timestamp.

5.6.3 Experiment Results

The experiment was repeated several times over the course of five weeks. In total 375 550 packets were transmitted and 741 689 packets were received. Of the 741 689 received packets, 47 515 (6.4 %) passed the CRC check, but were flagged as corrupt based on the payload.

The network topology is show in fig. 5.7. A directional edge is drawn between two nodes if at least one packet transmitted by node a is received by node b. From the

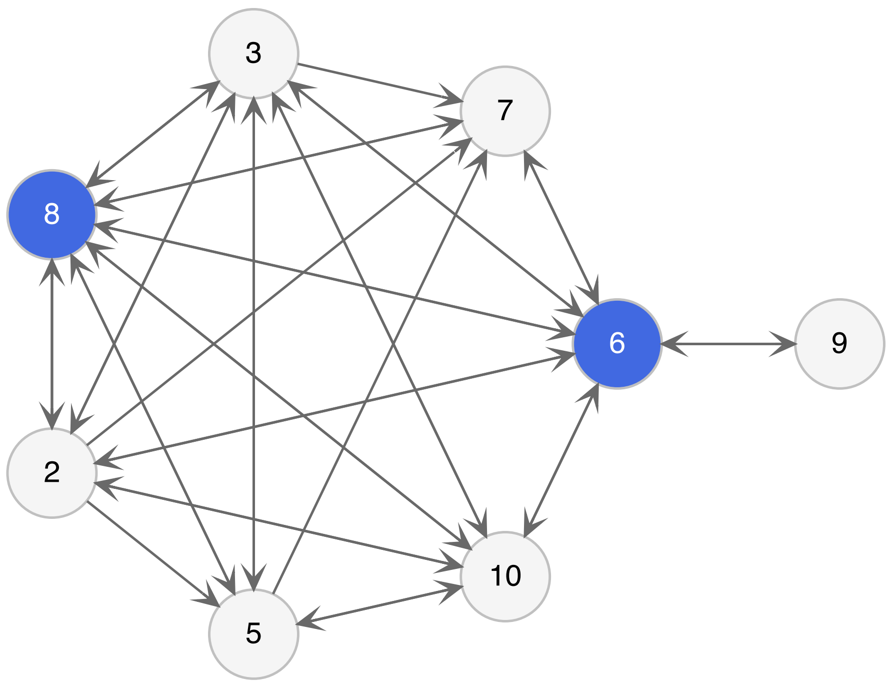


Figure 5.7: Network topology of the CATF testbed. A link indicates at least one packet was received. Blue nodes have the largest indegree centrality.

topology we can see links are asymmetrical. For example, node 7 received a packet from node 5, but node 5 did not receive a packet from node 7. We can also see there is not one single node that connects all other nodes, *i.e.* there is no obvious gateway node. The nodes with the highest indegree centrality of 1.71 are node 6 and node 8. Since this makes them the most central nodes in the topology, they would be the best candidates for the role of gateway.

Figure 5.8 shows a distribution plot of the RSSI for every received packet. The distribution has 3 modes, with peaks around −110 dBm (*mediocre link*), −60 dBm (*good link*), and −20 dBm (*excellent link*). Each mode corresponds with a group of nodes. The peak around −20 dBm corresponds with links between nodes that are in the same room (node 2 and 3) The peak around −60 dBm corresponds with links between nodes that are in the same building (node 2, 3, and 10) The peak around −110 dBm corresponds with links between nodes that are in different buildings.

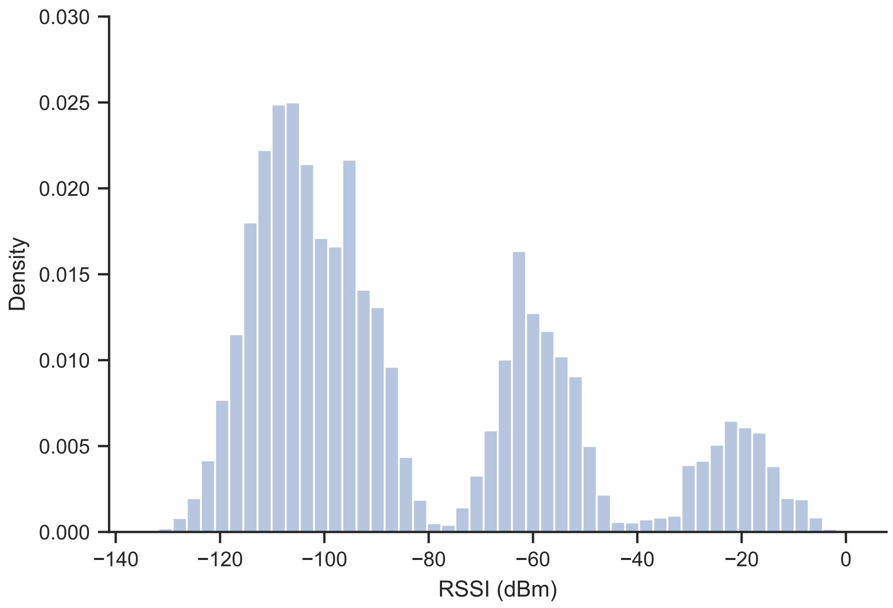


Figure 5.8: RSSI distribution of all received packets.

Heat maps for each link for RSSI, SNR and PRR are shown in fig. 5.9, fig. 5.10, and fig. 5.11, respectively. For the RSSI heatmap (fig. 5.9) and SNR heatmap (fig. 5.10), the values are the mean of the respective metric over the course of the entire experiment. For the PRR heatmap (fig. 5.11), a value of 0.00 is a link that did receive a packet, but less than 1% of packets arrived, *e.g.* the PRR of (5, 3) is 1.7×10^{-4} (0.017%). A blank space is a link that did not receive any packet at all.

These heatmaps give a detailed insight in the quality of each individual link during the experiment. As was shown in fig. 5.8, the links can be roughly divided in 3 different groups. The RSSI heatmap (fig. 5.9) shows the excellent links as yellow, the good links as green, and the mediocre links as blue/purple. When we take the corresponding SNR heatmap (fig. 5.10), we can see that the excellent and good links have a positive SNR, meaning the received signal was above the noise floor. The mediocre links, however, have a negative SNR, indicating the received signal was below the noise floor.

When comparing the RSSI heatmap (fig. 5.9) with the PRR heatmap (fig. 5.11), one can see that a high RSSI (> −60 dBm) results in a high PRR (> 0.8). However, there

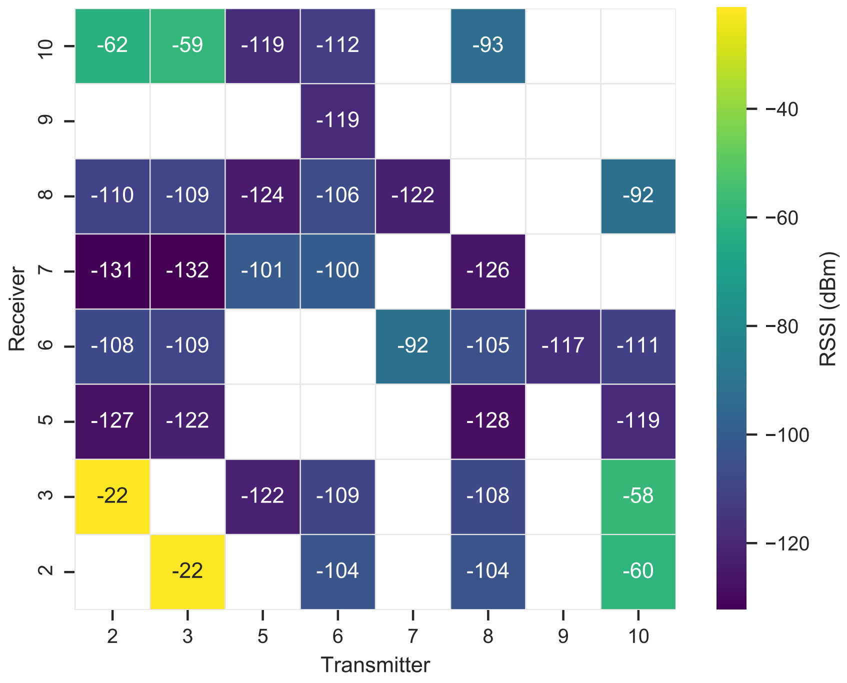


Figure 5.9: Mean RSSI for each link.

are links, *e.g.* node 6 to node 7, which have a much lower RSSI (< -100 dBm), but still have a good PRR (> 0.8). One has to note that the PRR figures are the *mean* for the entire experiment. These values give an indication of the quality of the link, though the link can attain a much higher PRR, depending on the configuration used. Figure 5.12 shows the attainable (maximum) PRR for each link. The heatmap shows that most links can be set up to provide an excellent connection. For the majority of links, a PRR > 0.8 is achievable, though this could require a consuming a lot of energy, using a configuration with the maximum transmit power and lowest data rate. As explained before, a faster, lower transmit power configuration with a worse PRR may be more energy efficient in transferring data.

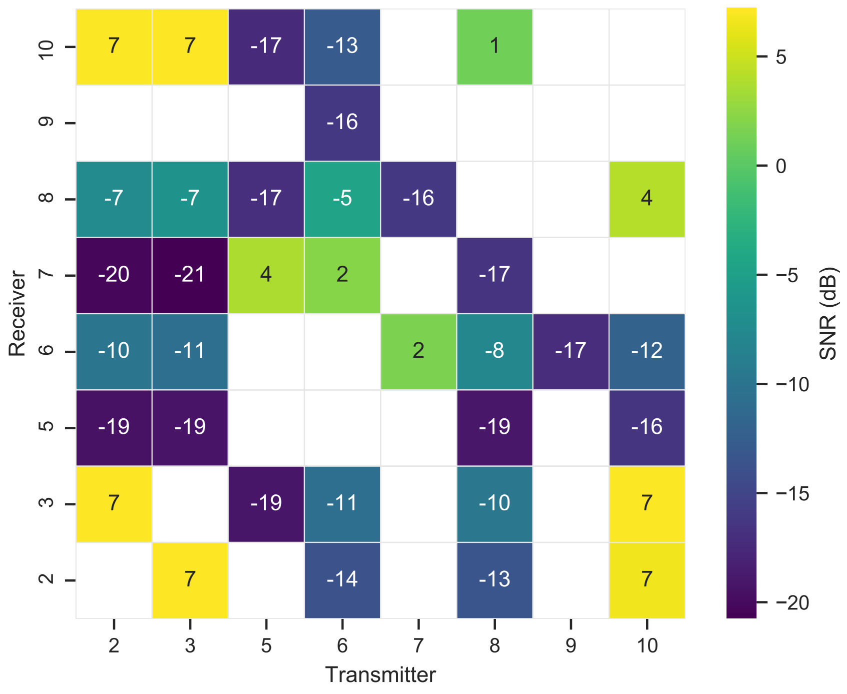


Figure 5.10: Mean SNR for each link.

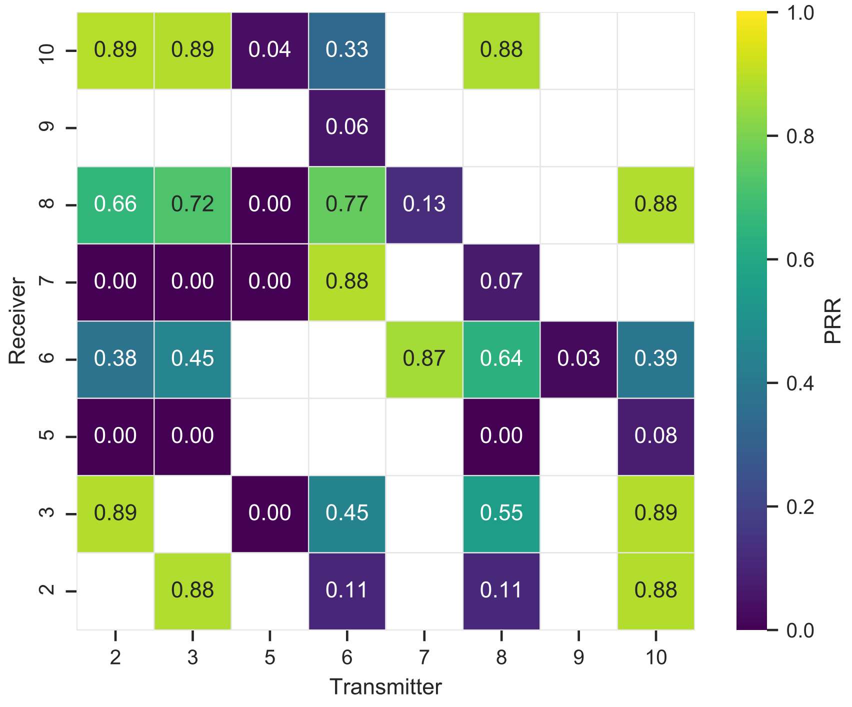


Figure 5.11: Mean PRR for each link.

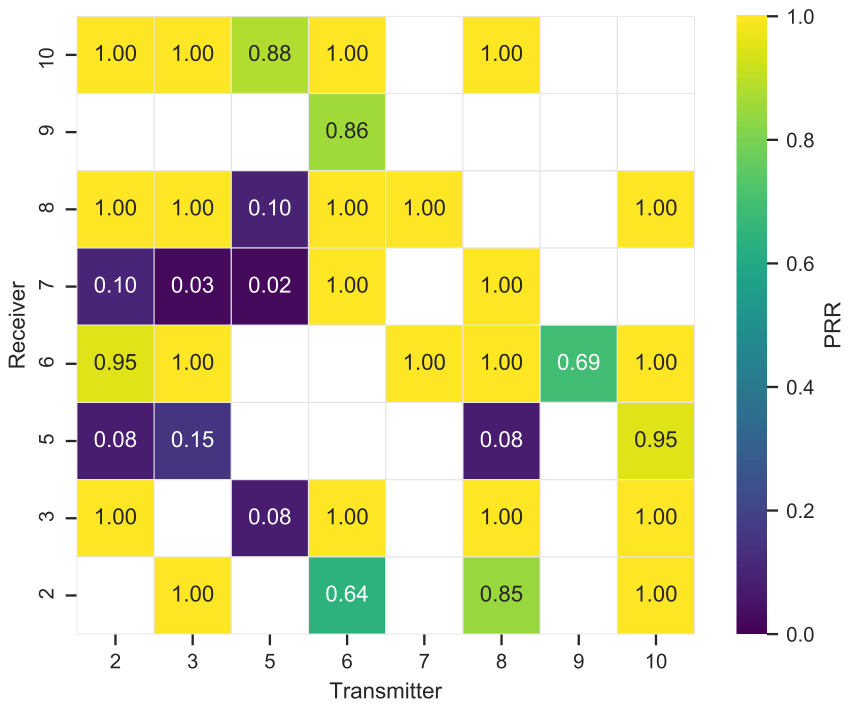


Figure 5.12: Attainable PRR for each link.

5.7 Evaluating ADR Algorithms

Using the packet traces gathered in the previous section, we simulate the ADR algorithms TTN, Probing, and Optimistic Probing. We evaluate the algorithms along the four metrics we defined in section 5.5: optimality, energy consumption, convergence speed, and stability.

5.7.1 Methodology

The ADR simulator is a purpose build discrete-event simulator based on SimPy [Lün+18], and distinct from LoRaSim as used in chapter 3. The simulator consists of two parts: a LoRa radio, and the implementation of each ADR algorithm with their corresponding LQE. The simulations are run on 4 different classes of links.

LoRa Radio

The LoRa radio can be set to all the different configurations, and keeps track of all packets sent, the energy consumed, and the time spent transmitting. There are two variants of the simulated radio: a regular LoRa radio, and a LoRaWAN radio. The LoRa radio is able to use the entire parameter space, with a SF from 7 to 12, BW of 125 kHz, 250 kHz, and 500 kHz, CR from 1 to 4, and TP from 2 dBm to 17 dBm. The LoRaWAN radio is limited to the parameter space as defined in table 5.2: SF from 7 to 12 with a BW 125 kHz, and SF7 with BW 125 kHz, TP from 2 dBm to 14 dBm, and CR fixed to 1.

For simulating the physical layer, we use the packet traces gathered during the experiments described in section 5.6. Transmitting and receiving packets is simulated by playing back the packet traces randomly., thereby emulating the topology shown in fig. 5.7. When node a transmits a packet, node b looks up the id of this packet to determine whether the packet was received successfully, and if so, what the RSSI, SNR, and CRC for this packet are.

The MAC layer is modelled according to LoRaWAN (see section 2.4.5). When a node sends a packet, it opens two receive windows, RX1 and RX2. The first receive window, RX1, is configured with the same data rate as the last transmitted packet. The second receive window, RX2, is configured for the lowest (DR0) data rate. When the gateway receives a packet, and it has something to send back (*e.g.*, an acknowledgement, or a new configuration), it will transmit during both windows, at the maximum transmit power (14 dBm). This increases the chances for packets to be received, but is no guarantee that the packet will actually arrive. This approach mimics a real-world deployment.

ADR Algorithms

With the general simulation setup described in the previous section, this section describes the specific implementation details and parameters for each ADR algorithm as used in the simulation.

TTN For the TTN ADR algorithm, one node is assigned the role of end-device, while the uplink node is assigned the role of gateway and NS.

We assume the NS is always able to control the data rate of the end-device, and therefore has the `ADR` bit set on every downlink frame header. The end-device does not request (regular) acknowledgements on any uplink frame, except for ADR. The NS does not send any downlink frames, unless explicitly requested by the end-device via the `ADRACKReq` bit in the uplink frame header. The parameter `ADR_ACK_LIMIT` is set to 62, and `ADR_ACK_DELAY` is set to 32.

At startup, the end-device starts transmitting packets on a regular interval to the gateway with the `ADRACKReq` bit set. After receiving 20 packets, the gateway calculates the new settings and sends back the result (which may get lost). After this startup phase, the node keeps on transmitting packets on a regular interval. As the gateway does not send back any downlink, the node will enable the `ADRACKReq` bit after transmitting `ADR_ACK_LIMIT` packets. The gateway will then recalculate the settings, and send a new suggestion. On successful receiving a downlink frame, the node will change its configuration as suggested by the gateway. If it does not receive a response in time, it will gradually increase transmit power, until the maximum of 14 dBm is reached, after which it gradually decreases the data rate.

Since the link margins that TTN uses are only defined for the 6 different data rates allowed by LoRaWAN (see table 5.2), we only simulate this algorithm with the LoRaWAN radio.

Probing The Probing algorithm uses the same communication model as TTN. Similar to TTN, the end-device starts with transmitting packets with the `ADRACKReq` bit set. Unlike with TTN, the gateway responds to every packet with this bit set, by sending a downlink frame containing the RSSI and SNR of the last received packet. The end-device unsets the `ADRACKReq` bit when it has found the optimal configuration. After this, it will keep transmitting packets using this configuration.

Probing is configured with a PRR threshold τ of 0.8, a RSSI threshold of –105 dBm, and 20 probes. Since Probing is independent of the parameter space, it can either use the LoRaWAN radio or the LoRa radio.

Optimistic Probing The implementation of Optimistic Probing works the same as Probing. Like Probing, Optimistic Probing can either use the limited LoRaWAN radio, or to full LoRa radio. Optimistic Probing is configured to use 3 probes at most.

Links

The simulations are run for 4 different links based on the classification as defined in section 5.6.3: excellent ($\overline{\text{RSSI}} \approx -20$ dBm), good ($\overline{\text{RSSI}} \approx -60$ dBm), mediocre ($\overline{\text{RSSI}} \approx -110$ dBm), and no link. By picking these 4 different kinds of links, we can see how well the ADR algorithms perform under different circumstances.

Table 5.4 shows the 4 links we use in the simulation. This table also shows the optimal setting, and the corresponding RSSI, SNR, EKB, and PRR when considering a LoRaWAN radio. Table 5.5 shows the same links, but for a LoRa radio. This also demonstrates that a further 20% to 50% reduction in energy consumption is possible when considering the full potential of a LoRa radio.

Table 5.4: LoRaWAN simulation links. From end-device (Nd) to gateway (Gw). Optimal setting is the most energy efficient setting (lowest EKB). RSSI and SNR are the mean for this optimal setting. PRR is over the whole experiment run. Node IDs refer to fig. 5.7 and table 5.3

Link		Optimal								
Nd	Gw	SF	BW	CR	TP	RSSI	SNR	EKB (J/kbit)	PRR	Quality
3	2	7	250	1	4	-22	6.7	7.47×10^{-6}	0.88	*excellent*
6	8	7	250	1	11	-106	-4.9	11.8×10^{-6}	0.77	*good*
3	6	7	125	1	13	-109	-11	39.3×10^{-6}	0.45	*mediocre*
5	6	–	–	–	–	–	–	–	0.00	*no link*

Table 5.5: LoRa simulation links. From end-device (Nd) to gateway (Gw). Optimal is the most energy efficient setting (lowest EKB). RSSI and SNR are the mean for this optimal setting. PRR is over the whole experiment run. Node IDs refer to fig. 5.7 and table 5.3

Link		Optimal								
Nd	Gw	SF	BW	CR	TP	RSSI	SNR	EKB (J/kbit)	PRR	Quality
3	2	7	500	1	2	-27	5.8	3.78×10^{-6}	0.87	*excellent*
6	8	7	500	1	11	-104	-6.1	7.13×10^{-6}	0.62	*good*
3	6	9	500	1	12	-106	-12	31.4×10^{-6}	0.46	*mediocre*
5	6	–	–	–	–	–	–	–	0.00	*no link*

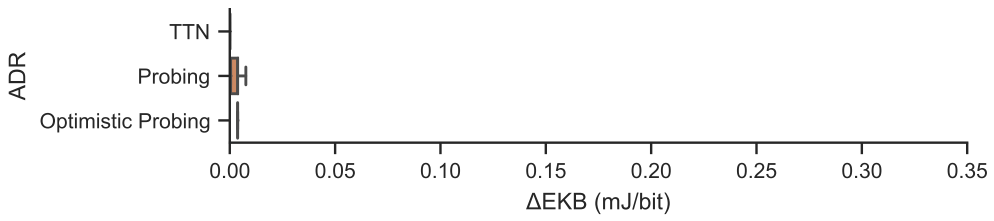
5.7.2 Results

Using the approach described in the previous section, we ran the simulator a 100 times, for every ADR algorithm. The starting condition for all algorithms was DR0 (SF12, BW125, CR1) at 14 dBm, unless otherwise specified.

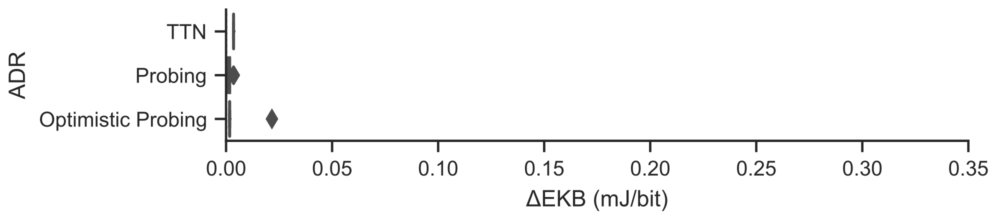
Optimality

Figure 5.13 and fig. 5.14 show the optimality performance of each ADR algorithm, using a LoRaWAN and LoRa radio respectively. We omitted the non-existing link, as it has no optimal setting.

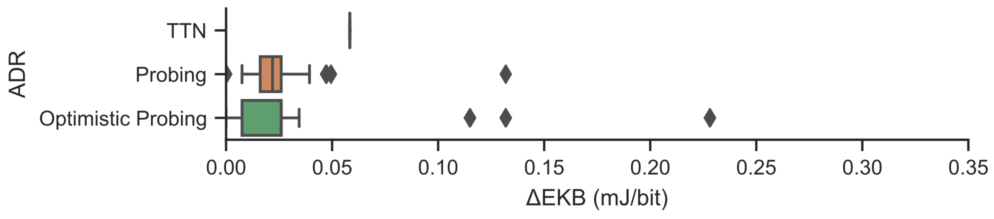
The box plots in the figures show the difference (ΔEKB) between the EKB of the



(a) Excellent Link (3, 2)



(b) Good Link (6, 8)



(c) Mediocre Link (3, 6)

Figure 5.13: Optimality of each ADR algorithm for 3 types of links for a LoRaWAN radio. ΔEKB is the difference between the optimal EKB and the chosen configuration.

optimal configuration and the configuration the algorithm deems optimal. Ideally, this should be zero, meaning the chosen configuration is equivalent to the optimal configuration. This chosen configuration may not be the same as shown in table 5.5, but performs the same. The further away from zero, the less optimal the chosen configuration is.

LoRaWAN We first have a look at how the ADR algorithms perform when using a LoRaWAN radio. For the excellent link, shown in fig. 5.13a, all the ADR algorithms reach a configuration that is (close to) optimal. TTN always reaches the optimal configuration, while Probing has a slightly worse optimality, though the difference is small. Optimistic Probing performs similar to Probing, although its optimality is slightly worse than Probing. It does consistently reach the same (sub optimal) configuration.

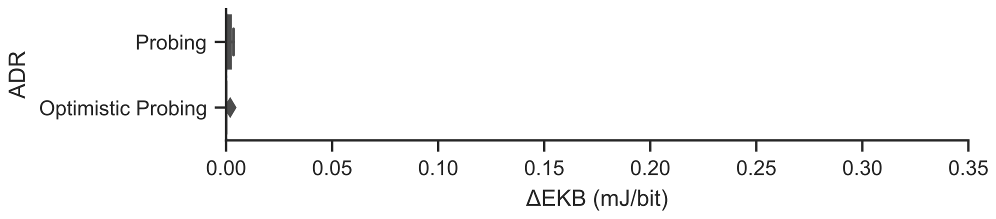
For the good link, shown in fig. 5.13b, the ADR algorithms perform similar for the excellent. TTN performs slightly worse than Probing and Optimistic Probing, but the difference is minimal. Optimistic Probing does have one outlier whereby the chosen configuration is significantly less optimal.

For the mediocre link, show in fig. 5.13c, the ADR algorithms perform quite differently, compared to the previous links. TTN consistently reaches the same configuration, though it is less optimal than Probing and Optimistic Probing.

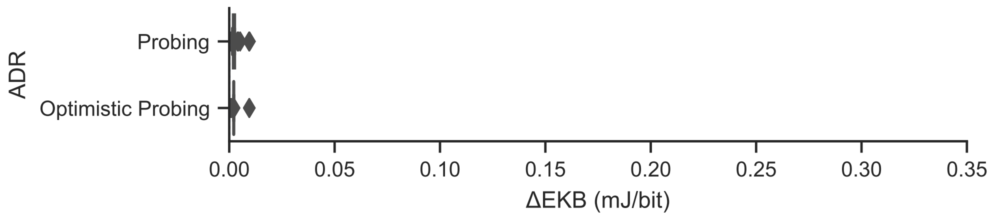
On the other hand, Probing and Optimistic Probing have a larger spread, as they are more sensitive to poor performing links. Since they rely on sampling the channel for a short amount of time, and the chance of losing a packet on the mediocre link is higher, they may incorrectly mark a configuration as 'bad', even though it may perform well over a longer time frame. TTN is less sensitive to packet loss, as it only considers the maximum link margin over the last 20 received packets, and does not take any packet loss into account. Only with excessive packet loss, the end-device may increase the transmit power or lower the data rate. Since TTN starts at the lowest data rate and highest transmit power, and takes small steps when increasing the data rate, it is much less likely to overshoot and end up with a bad configuration, unlike Probing and Optimistic Probing.

LoRa Figure 5.14 shows the optimality performance when using a LoRa radio. We have omitted TTN, as it is not defined for use with a LoRa radio, and the non-existing link for the same reason as in the previous section.

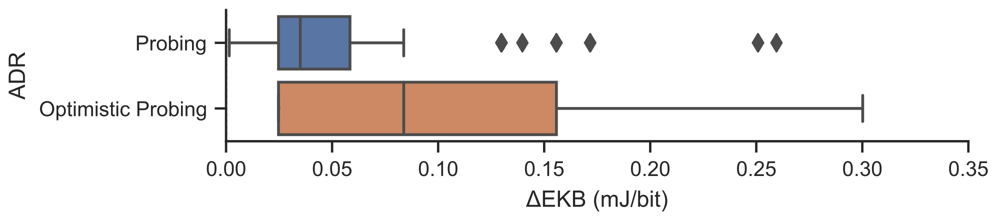
For the excellent link (fig. 5.14a) and the good link (fig. 5.14b), the optimality performance of both algorithms is the same as was shown with a LoRaWAN radio (fig. 5.13).



(a) Excellent Link (3, 2)



(b) Good Link (6, 8)



(c) Mediocre Link (3, 6)

Figure 5.14: Optimality of each ADR algorithm for 3 types of links for a LoRa radio. ΔEKB is the difference between the optimal EKB and the chosen configuration.

For the mediocre link (fig. 5.14c), the difference is much larger. Probing has much larger spread, with its median EKB above the optimal configuration. Optimistic Probing does not even reach the optimal configuration.

Compared to using a LoRaWAN radio (see fig. 5.13c), the ADR algorithms perform much worse. The reason is that the LoRa offers a much larger parameter space. With a mediocre link, this parameter space is much less continuous, with holes in between. This increases the chance of getting into a local optimum.

Another reason is that being at the edge of sensitivity, external interference has much higher influence on the performance of a radio. Different configurations with the same energy expenditure, may therefore perform differently. For example, a higher

CR is beneficial in the light of narrow-band interference [13, section 3.1, page 6]. As both Probing and Optimistic Probing only look at energy consumption when picking the next configuration, it may be that the configuration with a low CR is picked and discarded, while the configuration with a high CR and the same energy could have performed better.

Energy Consumption

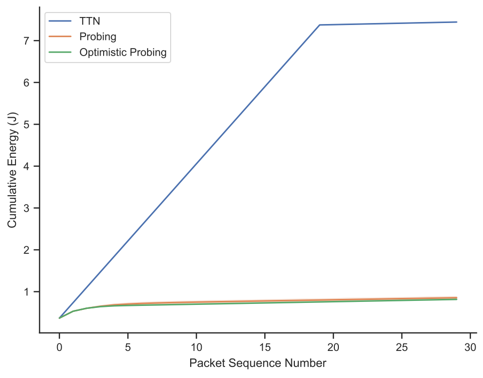
We assess the energy consumption of each ADR algorithm by looking at the cumulative energy consumption over time. The lower the cumulative energy consumption for a ADR algorithm, the longer the operating life time of a sensor node. To make things comparable, we use the packet sequence number as the 'time' measure.

LoRaWAN Figure 5.15 shows the cumulative energy consumption for the ADR algorithms using the LoRaWAN radio for each link type. The shaded area indicates the 95% confidence interval. Each graph is drawn from the initial transmission until shortly after the last algorithm has reached a stable state.

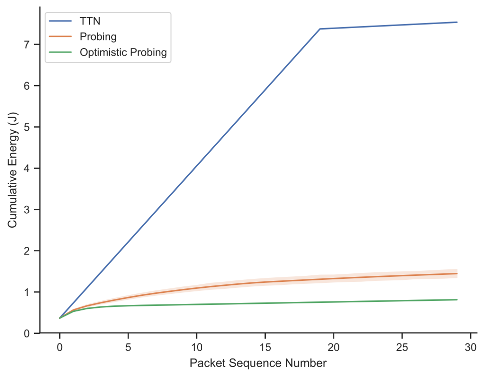
Figure 5.15a clearly shows that every algorithm has some initial probing phase. For TTN, this is after 20 packets, after which the slope of the line radically changes. For Probing and Optimistic Probing, this happens much sooner, on average after 11 and 9 packets respectively.

Even though all the algorithms reach a similar configuration, since TTN spends so much longer in a high power state, this has a significant impact on the operating life time of the node.

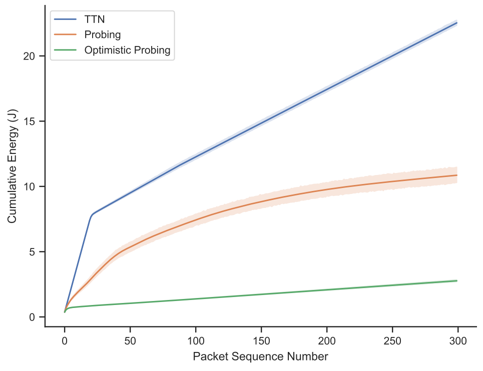
For the good link (fig. 5.15b) and the mediocre link (fig. 5.15c), the situation is similar. The main difference is that since there is more space to explore, Probing and Optimistic Probing take longer to reach a stable state, therefore spending more time in high energy state. Over the life time of the end-device, however, this is still signi-



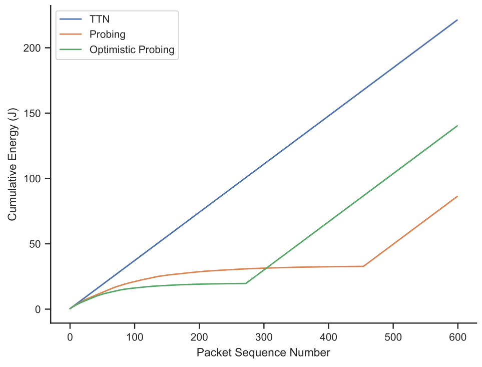
(a) Excellent Link (3, 2)



(b) Good Link (6, 8)



(c) Mediocre Link (3, 6)



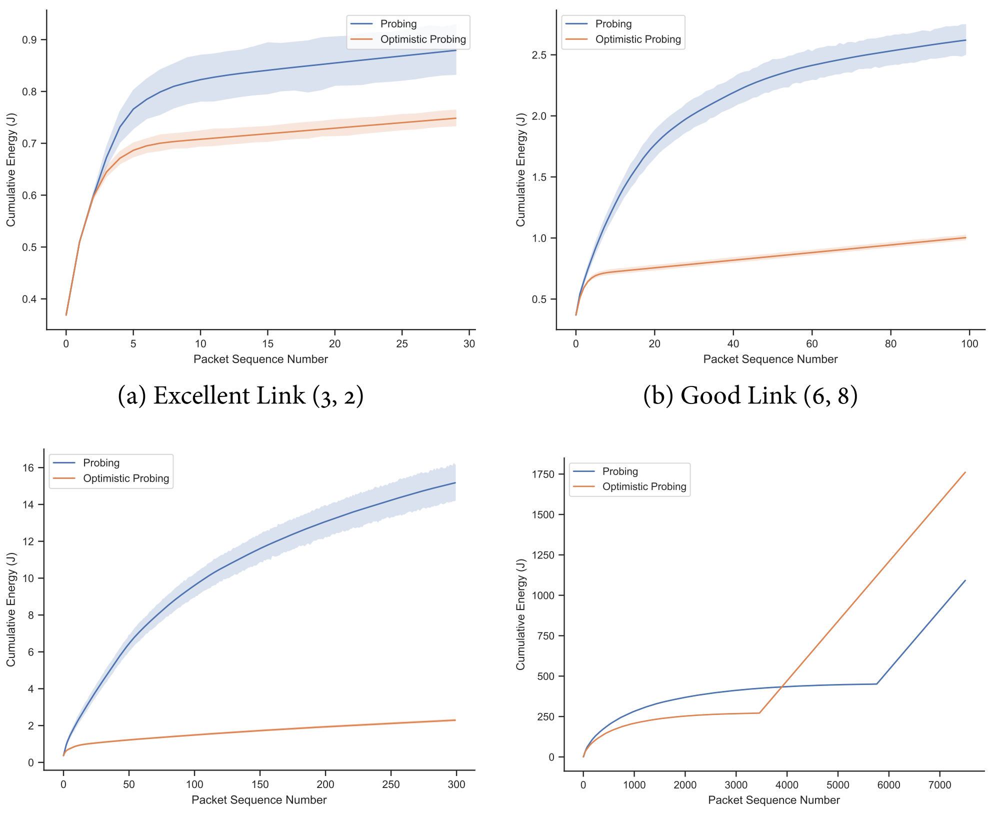
(d) No Link (5, 6)

Figure 5.15: Cumulative energy consumption for LoRaWAN ADR Algorithms. Shaded area indicates 95% confidence interval.

ficantly less than TTN.

Figure 5.15c shows the energy consumption when there is no link. Since TTN does not get any response, it will stay in the initial (high) energy state. Probing and Optimistic Probing, however do try to find a working setting by exploring the entire parameter space. Even though that seems wasteful, it actually extends the life time of the sensor node, since the other data rates are so much more energy efficient. For Probing this is even more so, as it spends more time probing each setting.

On the other hand, a proper sensor node would detect if there is no link. It would be better of just going to sleep and periodically check for any beacons, instead of desperately trying to find an optimal configuration.



(a) Excellent Link (3, 2)

(b) Good Link (6, 8)

(c) Mediocre Link (3, 6)

(d) No Link (5, 6)

Figure 5.16: Cumulative energy consumption for LoRa ADR algorithms. Shaded area indicates 95% confidence interval.

LoRa Figure 5.16 shows the cumulative energy consumption for the ADR algorithms using the LoRa radio for each link type. The shaded area indicates the 95% confidence interval. Each graph is drawn from the initial transmission until shortly after the last algorithm has reached a stable state.

For a LoRa radio, the story is very similar as it was for LoRaWAN radio. The main difference being that since there is a much larger parameter space, the algorithms spend a lot more time in the probing phase. This is especially noticeable with the no-existing link, as shown in fig. 5.16d. Probing spends almost 5000 packets in trying to come up with an optimal configuration. Like with the LoRaWAN radio, this is in the end is better for the operating life time of the node (although realising there is no link would be far better).

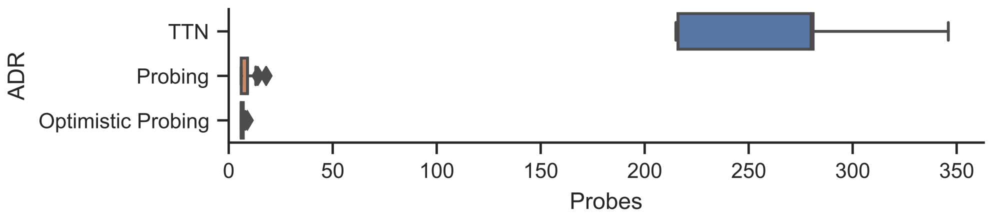
Being able to probe a larger parameter space does come at a cost. The algorithms spend more time in probing phase, exploring the the different parameters, before reaching a stable state. The actual impact on the sensor node operating life time is, however, not that much. The extra parameters are higher data rates, compared to LoRaWAN, that have significantly lesser energy consumption. Especially for the better links, this extra effort results in configurations that are much more efficient. The energy spend in probing these extra parameters is therefore well worth the effort.

Convergence Speed

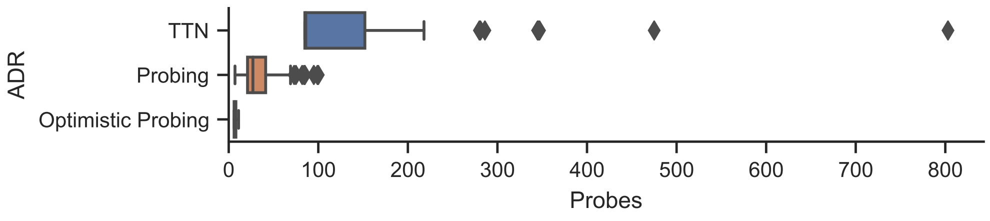
The impact of convergence speeds has already come up in the previous section when we looked at energy consumption. Here we look in more detail on how many packets are required before reaching a stable state.

LoRaWAN Figure 5.17 shows the convergence speed for each ADR algorithm when using a LoRaWAN radio. For the excellent link (fig. 5.17a), Probing and Optimistic Probing require significantly less packets to reach a stable state.

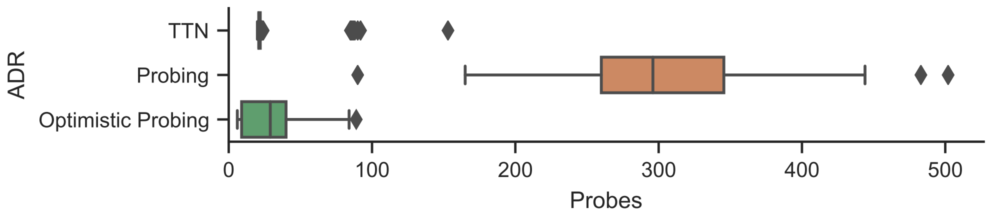
As the quality of the link worsens, this lead quickly disappears. For the good link (fig. 5.17b), both Probing and Optimistic Probing still require less probes to reach a stable state. For the mediocre link (fig. 5.17c), and especially the non-existing link (fig. 5.17d), however, both Probing and Optimistic Probing require significantly more probe packets. TTN consistently requires around 256 packets before reaching a stable state, because it only updates its data rate every 64 packets (`ADR_ACK_LIMIT`), and requires 4 steps to reach the fastest data rate DR6 from DR0. For the mediocre link, the number of probes is even 0, as the node does not get any response to adjust its data rate.



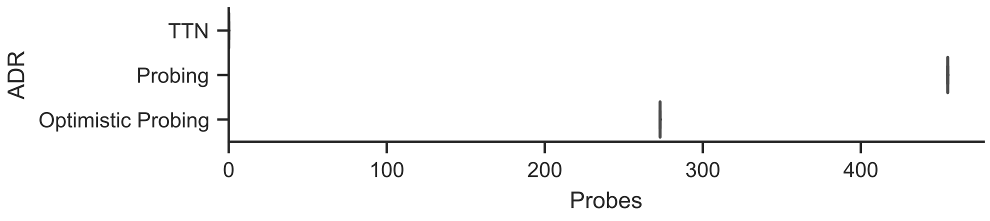
(a) Excellent Link (3, 2)



(b) Good Link (6, 8)

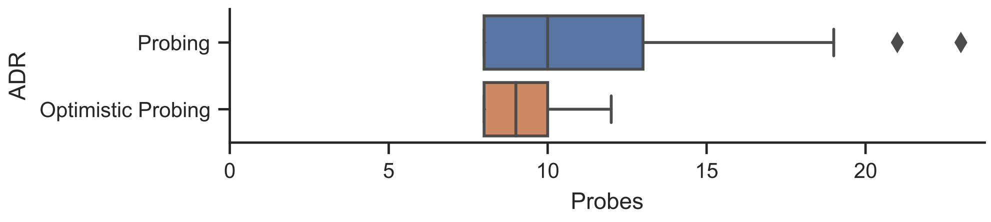


(c) Mediocre Link (3, 6)

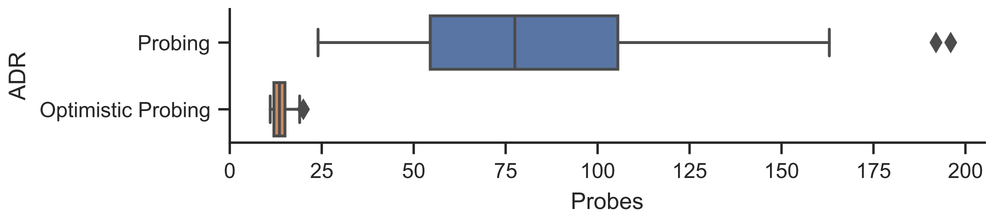


(d) No Link (5, 6)

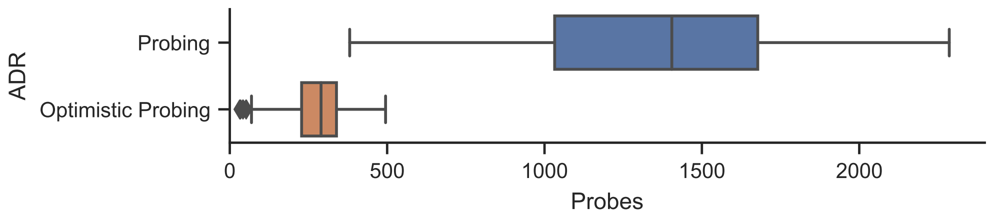
Figure 5.17: Probes required for each ADR algorithm to reach a stable state, for 4 types of links for a LoRaWAN radio.



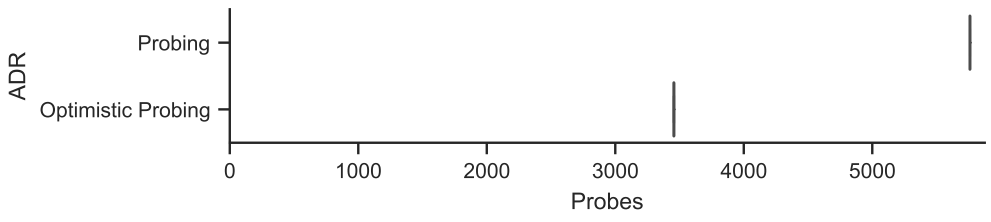
(a) Excellent Link (3, 2)



(b) Good Link (6, 8)



(c) Mediocre Link (3, 6)



(d) No Link (5, 6)

Figure 5.18: Probes required for each ADR algorithm to reach a stable state, for 4 types of links for a LoRa radio.

LoRa The convergence speeds when using a LoRa radio are shown in fig. 5.18. The optimistic probing approach of Optimistic Probing clearly shows it pays off, as it requires quite a lot less probes to reach a stable state, compared to Probing. In doing so it also has a much lower spread than Probing, potentially pointing to a more reliable operation.

Compared to a LoRaWAN radio, there are not that much more probes required to reach a stable state when the link is excellent, or good. When the link becomes mediocre, or even does not exist, the number of probes required shoots up.

Stability

The stability of the 3 ADR algorithms can be derived from looking at the outliers in optimality (fig. 5.13 and fig. 5.14) and convergence speed (fig. 5.17 and fig. 5.18), and in the confidence interval of the energy consumption (fig. 5.15 and fig. 5.16).

All algorithms have a stable performance when the link itself is excellent, or good. As the link quality becomes worse, the algorithms become more unstable.

TTN is the most stable one, having the least amount of outliers and the smallest confidence interval. The algorithm is designed to take small steps, over a long period of time. The downside of this approach is that it takes a long time to reach the optimal state, wasting a lot of energy. Also, this makes TTN not as responsive to quickly changing link dynamics.

Probing and Optimistic Probing, are more responsive than TTN, at the cost of being less stable. Although one would expect the more aggressive approach of Optimistic Probing would make it more unstable, the confidence interval on fig. 5.16 and the spread and outliers in fig. 5.17 shows it performs much better than Probing. This comes at the cost of reaching a less optimal state compared to Probing.

5.8 Conclusions

In this chapter, we had a look at transmission parameter selection, or Adaptive Data Rate (ADR) for LoRa. As demonstrated in chapter 4, adapting the data rate dynamically is essential for the scalability of LoRa. We first described an outline of an ADR algorithm, its properties, and requirements, and discussed how ADR is supported in LoRaWAN. To get a better insight into the link dynamics, and what constitutes a good link, we deployed a small-scale experiment, whereby we exchanged packets between two LoRa nodes on a wide range of parameters, and examined the influence of Spreading Factor (SF), Bandwidth (BW), Coding Rate (CR), and Transmission Power (TP) on the link performance. Based on these results, we developed two on-line ADR algorithms: Probing and Optimistic Probing. To evaluate ADR algorithms we defined four performance metrics: optimality, energy consumed, convergence speed, and stability. We evaluated and compared our two algorithms with TTN, the de facto standard for ADR in LoRaWAN, using simulation based on packet traces collected on a real-world testbed deployed on the Lancaster University campus over a period of five weeks. We demonstrated that Probing and Optimistic Probing outperform TTN in optimality, convergence speed, and energy consumption. Probing and Optimistic Probing are much more aggressive than TTN, making them more responsive to quickly changing link, and able to reach an efficient configuration quicker. In case of a poor link quality, however, TTN performs better. Probing and Optimistic Probing reach a less optimal configuration, though this is offset with the savings made by switching to a more energy-efficient configuration quicker than TTN. Therefore, Probing and Optimistic Probing give the best trade-off between optimality and operating lifetime in most situations, and therefore is the safe (albeit suboptimal) choice.

Chapter 6

Conclusions & Future Work

A Wireless Sensor Network (WSN) is a network of spatially distributed autonomous sensor nodes, that monitor physical or environmental conditions. They form a wireless ad hoc network, whereby sensor data is transported, potentially via other sensor nodes, to a central location. WSNs form an active research area, with many practical applications in academia (*e.g.* remote monitoring of permafrost [Tal+07]), in industry (*e.g.* wine production [Ana+09]), and for consumers (*e.g.* smart homes [Gre15, page 93]).

To enable high-volume deployment of WSNs, the research focus is on reducing cost and energy per sensor node, while simplifying development and maintenance. This has an effect on the used hardware, wireless communication bands, and energy sources. Sensor nodes are build from inexpensive COTS hardware. While cheap, they are severely resource-constrained, having processing power measured in megahertz, and memory and storage measured in kilobytes. This restricts the use of compute heavy algorithms, for data processing or security, as there is no space, time, or energy to run them.

Over the last couple of decades, many algorithms and solutions have been proposed to cope with these limitations and challenges. Most of the previous research has chosen to focus on a specific set of hardware platforms, ignoring many advances in the field of communication technology. These new technologies have grown into mature alternatives to the former state-of-the-art platforms. Even more so, they challenge existing assumptions, offer solutions to long standing problems, but also introduce a new set of challenges.

6.1 Overview

In this work we explored the recent advances in communication techniques that are of interest for WSNs. More specifically, we investigated LoRa, a novel long-range, low-power communication technique, which gained quite some traction in the last couple of years [LoR17a], opening up new opportunities for WSNs.

To better assess the usability of LoRa for WSNs, a thorough investigation is required to assess both its feasibility and the new challenges it introduces. The core research question we answered in this thesis is:

How feasible is LoRa for large-scale IoT deployments?

To focus our research towards answering this core research question, we devised the following sub-questions:

R1) How do the advertised features of LoRa work in practice?
R2) Is it possible to build a multi-hop LoRa network protocol?
R3) How many devices can a LoRa network really support?
R4) How can LoRa transmissions be dynamically optimised?

In the following sections, we summarise the answers to these questions, as presented in the thesis.

6.1.1 LoRa Features in Practice (R1)

Our first step was to evaluate the features of LoRa with practical experiments. The features we focused on are the orthogonality of Spreading Factor (SF), non-destructive concurrent transmissions, and Carrier Activity Detection (CAD).

Orthogonal spreading factors are useful in constructing subchannels on the same carrier frequency, allowing nodes to transmit at the same time without interfering with each other. In our experiments we found that this feature worked perfectly, and channel separation using different SF is possible.

The non-destructive transmissions is a feature that relies on the *capture effect*, whereby with two concurrent transmissions, a stronger signal suppresses the weaker signal, allowing the stronger signal to be received properly, instead of both transmissions being lost due to collision. This particular effect can be used whereby multiple gateways send beacons at the same time, without having to worry whether one beacon corrupts the other beacon. We found that this effect is present, and transmissions can be received with very high probability. There is however a critical section during transmission, in a strong packet cannot be received properly if a weak packet was being received a particular time offset before or after the strong packet.

CAD is a way that enables LoRa to perform CCA. The traditional approach to determine whether a transmission is ongoing, is by sampling the RSSI. When the RSSI is above a particular threshold, the channel is deemed 'occupied'. This approach does not work with LoRa, as transmissions up to −19.5 dB below the noise floor can still be decoded.

CAD works by sampling the channel for a short time, about 2 symbol periods, and analysing the results in a lower power mode for about 1 symbol periods, trying to detect LoRa preambles. This approach should be more accurate than RSSI, as it can detect an actual LoRa transmission. In addition, it is more energy efficient, as the sampling period is short, and the processing happens at a lower power mode. We found that CAD worked well, most of the time. We did find there were false positives depending on the combination of SF/BW, presumably because the preamble signals look quite similar.

6.1.2 Multi-Hop LoRa (R2)

Once the features and inner workings of LoRa have been uncovered, we focused on one of LoRaWAN's limitations: being a pure single hop network. One of the downside of being a single-hop network, is that all nodes need to be in range of a gateway. Nodes that are in a 'bad' spot, cannot be part of network. To this end, we pursued the analysis and development of a shallow multi-hop protocol to help resolve this issue.

We developed LoRaBlink, a TDMA-style multi-hop protocol, which makes extensive use of the CAD features of the LoRa radio. We demonstrated its feasibility with a proof of concept implementation on a small scale testbed. To demonstrate its merits, we evaluated its features with LoRaWAN, a single-hop MAC protocol that is the de facto standard for LoRa.

6.1.3 LoRa Scalability (R3)

When building a single-hop network, one expects to have a large number of nodes. LoRaWAN, the de facto standard MAC and application layer for LoRa, also claims to support thousands of devices. To verify this claim, we analysed the scalability of LoRa. To enable this analysis, we built a link model from experimental data gathered with the LoRa feature evaluation. This model was then turned into a simulator that we used to simulate a wide range of scenarios. Our results indicate that LoRa can support a large number of devices, but it does require changes to how networks are deployed at the moment, for example deploying multiple sinks, or using ADR.

6.1.4 Dynamically Optimising Transmit Parameters (R4)

We explored the dynamic adjustment of transmit parameters in chapter 5. We developed two methods of controlling the transmit parameters, and compared these with the de facto standard solution for LoRaWAN, currently deployed in the crowd-sourced The Things Network [The19b]. We showed that our methods are more responsive to changes in the channel, though may be a bit too aggressive.

6.1.5 Feasibility of LoRa for Large-Scale IoT Deployments (core RQ)

With the answer to R3, we can say that LoRa, as it is currently deployed, is not feasible for large-scale IoT deployments. However, within LoRa itself there is space to make it feasible for large-scale deployments. The answer to R2 shows how developing a new MAC protocol can help. To answer R4, we explored dynamically optimising transmit parameters. We have shown that with ADR, LoRa can be made much more scalable. The algorithms we proposed can be implemented in an existing LoRaWAN network, without requiring any changes to the standard. However, they are more powerful when deployed in a network without the technical limitations of LoRaWAN, therefore making LoRa truly feasible for large-scale IoT.

6.2 Contributions

This thesis makes several contributions to the field of WSN, aiming to advance research by investigating the merits and challenges of LoRa. As such, we contribute methods, models, and experimental results that change long-held assumptions. Specifically, the contributions of this thesis are:

C1) *An in-depth investigation of the physical layer of LoRa (chapter 3).*

LoRa is a novel LPWAN communication technique, with some unique features. The technical details of LoRa, however, are not well documented, as well there is little empirical analysis of LoRa. We are one of the first to perform this analysis, and set a benchmark for future research.

C1.1) *Novel experiment design for three key LoRa features.*

We developed novel experiment design for three key LoRa features: Orthogonal spreading factors, non-destructive concurrent transmissions, and Carrier Activity Detection (CAD). We discovered orthogonal spreading factors can provide virtual channels, though with some caveats. CAD can in making a energy efficient CCA, though it will only work for LoRa communication. The non-destructive concurrent transmissions proved to be useful in, though there are some constraints on how transmissions overlap. These findings are useful for future design and development of *e.g.* MAC protocols.

C1.2) *Improved documentation on inner workings of LoRa.*

We provided a compilation of existing reverse engineering work and patents analysis, supplemented with our own empirical studies.

We argue that our experiments are truly fundamental for understanding the unique features and limitations of LoRa, like long-range communication, virtual channels, and the constraints on the effective use of non-destructive concurrent transmissions. Moreover, the experiment design and results can be used to further explore LoRa's key features, for example in other deployment environments than urban environment we used in our study, *e.g.* rural, city, industrial, and off-shore.

C2) *Methods for analysing the scalability of LoRa-based WSNs (chapter 4).*

LoRa is an interesting new radio technology that promises interconnecting millions of embedded devices, all within a single hop. However, whether this is actually feasible in practice requires thorough scenario-based analysis.

C2.1) *Design and implementation of a LoRa simulator.*

We are the first to develop a dedicated LoRa simulator that is based on exhaustive empirical measurements. With this simulator we can quickly iterate over various scenarios and configurations, gaining a fundamental understanding of the scalability of LoRa.

C2.2) *Defined performance metrics for collection and energy.*

We defined Data Extraction Rate (DER) and Network Energy Consumption (NEC) as scalability-relevant metrics.

C2.3) *Derived LoRa scalability limitations.*

With the simulator and our defined metrics, we determined that a LoRa network, as it is currently deployed in LoRaWAN, can only scale up to 64 nodes realistically, which is not enough for a large-scale deployment.

C2.4) *Proposed deployment changes to improve scalability.*

To address the scalability limitation, we proposed two improvements to the current deployment. Firstly is the introduction of more sinks, or gateways, so nodes have a larger chance of reaching a gateway. Secondly is the use dynamic transmission parameter selection, or Adaptive Data Rate (ADR). Using our simulator, we demonstrated that these suggestions can dramatically improve scalability.

The LoRa simulator we build proved essential for LoRa research in this work, and is used by many other researchers (*e.g.* [ACP18; SPF18; RGS18; Zai+18; Hoe+18; CBR17; KIA17]), and is extended by other research groups for their own explorations and analysis of LoRa and LoRaWAN (*e.g.* [Ikh+18; CRO18; FP18; Pop+17]).

C3) *A dynamic on-line ADR algorithm for optimising energy consumption (chapter 5)*

One approach to improve the scalability of LoRa networks, is the use of dynamic transmission parameter selection, or ADR.

C3.1) *Behaviour analysis based on empirical link measurements.*

We collected a large dataset of LoRa link performance traces for multiple configurations, spanning several weeks. We analysed these extensively to demonstrate the influence of various parameters on communication performance.

C3.2) *Design and implementation of an accurate ADR LoRa simulator.*

Building upon the extensive link performance traces, we designed and implemented the first trace-driven LoRa transmission simulator for research in ADR algorithms.

C3.3) *Design and detailed evaluation of two novel ADR algorithms.*

We develop two on-line algorithms, *Probing* and *Optimistic Probing*, for tuning the transmission parameters of a LoRa radio for optimal communication performance and energy efficiency. To evaluate these ADR algorithms, we defined novel metrics for performance evaluation, namely optimality, energy consumption, convergence speed, and stability. We evaluated these two new algorithms, together with the de facto standard LoRaWAN ADR as implemented by TTN, and discuss the trade-offs of these algorithms, in terms of optimality, energy consumption, convergence speed, and stability.

We previously demonstrated that dynamic transmission parameter selection can dramatically improve the scalability of a LoRa network. With the proposed algorithms we showed that they improve on the current state-of-the-art, making the deployment of a large-scale LoRaWAN IoT network feasible. With the traces collected, and the metrics defined, we set a benchmark for further research on ADR for LoRa, to compare and improve upon.

6.3 Future Work

Although this work has proven that new communication technologies are very promising for improving WSN state-of-the-art, more analysis and implementation work is still needed to push these technologies to a main-stream status. We suggest the following future work directions that spawn from our results.

6.3.1 Long-term Temporal Effects on LoRa Links

In chapters 3 and 5 we focused on the real-world behaviour of a LoRa links with data gathered from a small number nodes, in an urban environment over several weeks. This is a relatively short period of time, and only one use case. To build more accurate link models, larger scale and longer term experiments in different environments (*e.g.* rural, high rise, open) are required.

Setting up a large-scale LoRa testbed introduces some unique practical challenges. Due to the long communication ranges, sensor nodes should be dispersed over a much larger area, requiring the support of many more parties.

A potential approach would be to design a public testbed, usable by other researchers. This model has been proven to be quite useful in the space WSN research, with testbeds like the Kansei testbed [Ert+06], WISEBED [Cha+10], and the FIT IoT-LAB [Adj+15].

Some early work in this area is for example Pervasive Nation [Per19], offering a LPWAN with LoRa nodes in Ireland, and ETH Zürich expanding its FlockLab with LoRa nodes [Trü+19].

6.3.2 Transmission Parameter Selection

The results in chapter 4 showed the importance of using (dynamic) transmission parameter selection, or ADR, for scalable LoRa-based LR-WPAN-networks. More investigation in ADR is required, as there are many options for improving the performance. For example, the sampling of channels could perhaps be done more efficiently. One approach would be to apply statistical approaches, like bandit strategies for solving the multi-armed bandit problem. The problem of ADR maps well on the the multi-armed bandit problem, where there is a trade-off between exploring and exploiting choices, with limited resources and maximising gain.

Another avenue worth exploring is machine learning, for example reinforcement learning. Training a machine learning model on a sensor node is probably not feasible, unless one builds a sensor node with dedicated machine learning chips. On the other hand, the use of trained machine learning models on microcontroller platforms has been demonstrated. This also ties back to the future work of studying of temporal effects. The data gathered during this approach can be used to train models.

6.3.3 LPWAN MACs

More research in MAC for LoRa, and other LPWAN technologies is required. LoRaWAN may be the standard for LoRa, but is far from ideal. Having a single-hop instead of a multi-hop network, covering a large area and therefore having a lot of active devices, introduces new challenges that we have not seen before. The subject of MACs has been extensively studied within WSNs, and the lessons and experiences form a large corpus of MAC WSN algorithms [Bac+10; Hua+13]. Future work would be required to research to what sense these algorithms could be applied, and what new algorithms can be used to exploit the unique properties of (for example) LoRa.

6.3.4 Transmission Parameter Selection in Other Low-Power, Low-Data Rate Technologies

With our work in chapter 5, and the proposed future work, we focus on LoRa. The problem of transmission parameter selection, however, can also be applied to other

LPWAN communication technologies, like Sigfox, Bluetooth LE, and NB-IoT. All these technologies have adjustable data rates that need to be dynamically configured. The algorithms we have developed may be applicable to these other communication technologies as well.

References

[09] *Cyber-Physical Systems (CPS)*. NSF 08-611. US National Science Foundation, 2009. URL: https://www.nsf.gov/pubs/2008/nsf08611/nsf08611.htm.

[12] *Electromagnetic compatibility and Radio spectrum Matters (ERM); Short Range Devices (SRD); Radio equipment to be used in the 25 MHz to 1000 MHz frequency range with power levels ranging up to 500 mW; Part 1: Technical characteristics and test methods*. EN 300 220-1 V2.4.1. May 2012.

[13] *SX1272/3/6/7/8: LoRa Modem Designer's Guide*. AN1200.13. Version 1. Semtech Corporation. July 2013.

[14] *Reading channel RSSI during a CAD*. AN1200.21. Version 1. Semtech Corporation. Oct. 2014.

[15] *ERC Recommendation 70-03: Relating to the use of Short Range Devices (SRD)*. CEPT/ERC/REC 70-03. Sept. 2015.

[16a] 'IEEE Standard for Low-Rate Wireless Networks'. In: *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)* (Apr. 2016), pp. 1–709. DOI: 10.1109/IEEESTD.2016.7460875.

[16b] *Simple Rate Adaptation Recommended Algorithm*. Version 1.0. Unpublished. Semtech. 2016.

[18] *XE216-512-TQ128 Datasheet*. X006991. XMOS. Sept. 2018. URL: https://www.xmos.com/developer/download/private/XE216-512-TQ128-Datasheet%281.16%29.pdf.

[ACP18] K. Q. Abdelfadeel, V. Cionca and D. Pesch. 'A Fair Adaptive Data Rate Algorithm for LoRaWAN'. In: *EWSN*. 2018.

[Adj+15] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele and T. Watteyne. 'FIT IoT-LAB: A large scale open experimental IoT testbed'. In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. Dec. 2015, pp. 459–464. DOI: 10.1109/WF-IoT.2015.7389098.

[Aer16] J. Aerts. 'Integrating Long Range Technology into the Contiki Operating System Framework'. MA thesis. Vrije Universiteit Brussel, June 2016.

[Ana+09] G. Anastasi, O. Farruggia, G. L. Re and M. Ortolani. 'Monitoring High-Quality Wine Production using Wireless Sensor Networks'. In: *2009 42nd Hawaii International Conference on System Sciences*. Jan. 2009, pp. 1–7. DOI: 10.1109/HICSS.2009.313.

[ARM] ARM Limited. *big.LITTLE*. URL: https://developer.arm.com/technologies/big-little.

[Bac+10] A. Bachir, M. Dohler, T. Watteyne and K. K. Leung. 'MAC essentials for wireless sensor networks'. In: *Communications Surveys & Tutorials, IEEE* 12.2 (2010), pp. 222–248.

[Bac+12] N. Baccour, A. Koubâa, L. Mottola, M. A. Zúñiga, H. Youssef, C. A. Boano and M. Alves. 'Radio Link Quality Estimation in Wireless Sensor Networks: A Survey'. In: *ACM Trans. Sen. Netw.* 8.4 (Sept. 2012), 34:1–34:33. ISSN: 1550-4859. DOI: 10.1145/2240116.2240123.

[Bea] BeagleBoard.org Foundation. *BeagleBoard.org - Community supported open hardware computers for making*. URL: https://beagleboard.org.

[BKR15] M. Bor, A. King and U. Roedig. 'Lifetime bounds of Wi-Fi enabled sensor nodes'. In: *Procedia Computer Science* 52 (June 2015): *The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015)*. Ed. by E. Shakshuki, pp. 1108–1113. ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.05.127.

[Bor+16] M. Bor, U. Roedig, T. Voigt and J. M. Alonso. 'Do LoRa Low-Power Wide-Area Networks Scale?' In: *MSWIM '16*. Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. MSWIM '16 (Malta, Malta, 13th–17th Nov. 2016). ACM SIGSIM. New York, NY, USA: ACM Press, Nov. 2016, pp. 59–67. ISBN: 978-1-4503-4502-6. DOI: 10.1145/2988287.2989163.

[Bor15] M. Bor. *LoRaBlink Kit*. Version 20151214. 14th Dec. 2015. URL: https://www.lancaster.ac.uk/scc/sites/lora/lorablinkkit.html.

[BR14] M. Bor and U. Roedig. 'OpenCL as Wireless Sensor Network programming abstraction'. In: *IWOCL '14*. Proceedings of the International Workshop on OpenCL 2014. IWOCL 2014 (Bristol, United Kingdom, 12th–13th May 2014). Ed. by S. McIntosh-Smith and B. Bergen. AMD et al. New York, NY, USA: ACM, May 2014. ISBN: 978-1-4503-3007-7.

[BR17] M. Bor and U. Roedig. 'LoRa Transmission Parameter Selection'. In: *2017 International Conference on Distributed Computing in Sensor Systems (DCOSS)*. DCOSS 2017 (Ottawa, Canada, 5th–7th June 2017). IEEE, June 2017. DOI: 10.1109/DCOSS.2017.10.

[BV17] M. Bor and T. Voigt. *LoRaSim*. Version 0.2.1. 10th June 2017. URL: https://www.lancaster.ac.uk/scc/sites/lora/lorasim.html.

[BVR16] M. Bor, J. E. Vidler and U. Roedig. 'LoRa for the Internet of Things'. In: *International Conference on Embedded Wireless Systems and Networks (EWSN) 2016*. Proceedings. MadCOM 2016 (Graz, Austria, 15th–17th Feb. 2016). Ed. by K. Römer, K. Langendoen and T. Voigt. Canada: Junction Publishing, Feb. 2016, pp. 361–366. ISBN: 978-0-9949886-0-7. URL: http://dl.acm.org/citation.cfm?id=2893711.2893802.

[CBR17] M. Cattani, C. A. Boano and K. Römer. 'An Experimental Evaluation of the Reliability of LoRa Long-Range Low-Power Wireless Communication'. In: *Journal of Sensor and Actuator Networks* 6.2 (2017). ISSN: 2224-2708. DOI: 10.3390/jsan6020007. URL: http://www.mdpi.com/2224-2708/6/2/7.

[CG01] S. T. Chung and A. J. Goldsmith. 'Degrees of freedom in adaptive modulation: a unified view'. In: *IEEE Trans. Communications* 49 (2001), pp. 1561–1571.

[Cha+10] I. Chatzigiannakis, S. Fischer, C. Koninis, G. Mylonas and D. Pfisterer. 'WISEBED: An Open Large-Scale Wireless Sensor Network Testbed'. In: *Sensor Applications, Experimentation, and Logistics*. Ed. by N. Komninos. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 68–87. ISBN: 978-3-642-11870-8.

[CRO18] M. Cesana, A. Redondi and J. Ortìn. 'A Framework for Planning LoRaWan Networks'. In: *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. Sept. 2018, pp. 1–7. DOI: 10.1109/PIMRC.2018.8580875.

[Doh+08] M. Dohler, T. Watteyne, F. Valois and J.-L. Lu. 'Kumar's, Zipf's and Other Laws: How to Structure a Large-Scale Wireless Network?' In: *Annals of Telecommunications* 63.5 (June 2008), pp. 239–251. ISSN: 1958-9395. DOI: 10.1007/s12243-008-0026-5. URL: https://doi.org/10.1007/s12243-008-0026-5.

[DPS14] E. Dahlman, S. Parkvall and J. Sköld. *4G - LTE/LTE-Advanced for Mobile Broadband (2nd Edition)*. Elsevier, 2014. ISBN: 978-0-12-419985-9.

[Duj+14] D. Dujovne, T. Watteyne, X. Vilajosana and P. Thubert. '6TiSCH: deterministic IP-enabled industrial internet (of things)'. In: *IEEE Communications Magazine* 52.12 (2014), pp. 36–41.

[Dun11] A. Dunkels. *The ContikiMAC Radio Duty Cycling Protocol*. Tech. rep. T2011:13. SICS, Dec. 2011. URL: http://dunkels.com/adam/dunkels11contikimac.pdf.

[Dut+10] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang and A. Terzis. 'Design and Evaluation of a Versatile and Efficient Receiver-initiated Link Layer for Low-power Wireless'. In: *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. SenSys '10. Zürich, Switzerland: ACM, 2010, pp. 1–14. ISBN: 978-1-4503-0344-6. DOI: 10.1145/1869983.1869985. URL: http://doi.acm.org/10.1145/1869983.1869985.

[Ert+06] E. Ertin, A. Arora, R. Ramnath, V. Naik, S. Bapat, V. Kulathumani, M. Sridharan, H. Zhang, H. Cao and M. Nesterenko. 'Kansei: A Testbed for Sensing at Scale'. In: *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*. IPSN '06. Nashville, Tennessee, USA: ACM, 2006, pp. 399–406. ISBN: 1-59593-334-4. DOI: 10.1145/1127777.1127838. URL: http://doi.acm.org/10.1145/1127777.1127838.

[Fer+11] F. Ferrari, M. Zimmerling, L. Thiele and O. Saukh. ‘Efficient network flooding and time synchronization with Glossy’. In: *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*. Apr. 2011, pp. 73–84.

[Fie10] F. Fietkau. *minstrel_ht: new rate control module for 802.11n*. 1st Mar. 2010. URL: https://lwn.net/Articles/376765/.

[FP18] M. O. Farooq and D. Pesch. ‘Poster: Extended LoRaSim to Simulate Multiple IoT Applications in a LoRaWAN.’ In: *EWSN*. 2018, pp. 175–176.

[Ger99] N. A. Gershenfeld. *When Things Start to Think*. Henry Holt, 1999. ISBN: 0-8050-5874-5.

[Gna+09] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss and P. Levis. ‘Collection tree protocol’. In: *SenSys ’09*. 2009.

[Gre15] S. Greengard. *The Internet of Things*. MIT Press Essential Knowledge series. MIT Press, Mar. 2015. 232 pp. ISBN: 9780262527736.

[GS96] L. Goudge and S. Segars. ‘Thumb: reducing the cost of 32-bit RISC performance in portable and consumer applications’. In: *COMPCON ’96. Technologies for the Information Superhighway Digest of Papers*. Feb. 1996, pp. 176–181. DOI: 10.1109/CMPCON.1996.501765.

[GT16] D. Guinard and V. Trifa. *Building the Web of Things. with examples in Node.js and Raspberry Pi*. Shelter Island, NY, 2016. ISBN: 9781617292682.

[GV97] A. J. Goldsmith and P. P. Varaiya. ‘Capacity of Fading Channels with Channel Side Information’. In: *IEEE Trans. Inform. Theory* 43 (1997), pp. 1986–1992.

[Her+11] J. M. Hernández-Muñoz, J. B. Vercher, L. Muñoz, J. A. Galache, M. Presser, L. A. Hernández Gómez and J. Pettersson. 'Smart Cities at the Forefront of the Future Internet'. In: *The Future Internet*. Ed. by J. Domingue, A. Galis, A. Gavras, T. Zahariadis, D. Lambert, F. Cleary, P. Daras, S. Krco, H. Müller, M.-S. Li, H. Schaffers, V. Lotz, F. Alvarez, B. Stiller, S. Karnouskos, S. Avessta and M. Nilsson. Vol. 6656. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 447–462. ISBN: 978-3-642-20897-3. DOI: 10.1007/978-3-642-20898-0_32. URL: http://dx.doi.org/10.1007/978-3-642-20898-0_32.

[Hoe+18] A. Hoeller, R. D. Souza, O. L. A. López, H. Alves, M. de Noronha-Neto and G. Brante. 'Exploiting Time Diversity of LoRa Networks Through Optimum Message Replication'. In: *2018 15th International Symposium on Wireless Communication Systems (ISWCS)* (2018), pp. 1–5.

[Hor07] C. A. Hornbuckle. 'Fractional-N synthesized chirp generator'. U.S. pat. US7791415B2. Semtech Corporation. 18th May 2007.

[Hua+13] P. Huang, L. Xiao, S. Soltani, M. W. Mutka and N. Xi. 'The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey'. In: *IEEE Communications Surveys & Tutorials* 15 (2013), pp. 101–120.

[Hüh13] T. Hühn. 'A Measurement-Based Joint Power and Rate Controller for IEEE 802.11 Networks'. PhD thesis. Technische Universität Berlin, FG INET Prof. Anja Feldmann, June 2013. URL: http://opus4.kobv.de/opus4-tuberlin/frontdoor/index/index/docId/3939.

[IGE00] C. Intanagonwiwat, R. Govindan and D. Estrin. 'Directed diffusion: a scalable and robust communication paradigm for sensor networks'. In: *MobiCom '00*. 2000.

[Ikh+18] M. G. Ikhsan, M. Y. A. Saputro, D. A. Arji, R. Harwahyu and R. F. Sari. 'Mobile LoRa Gateway for Smart Livestock Monitoring System'. In: *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*. Nov. 2018, pp. 46–51. DOI: 10.1109/IOTAIS.2018.8600842.

[Joh14] I. Johnson. *Code Size – A Comprehensive Comparision of microMIPS32 and Thumb Code Size Using Many Megabytes of User Code*. 28th Apr. 2014. URL: https://community.arm.com/processors/b/blog/posts/code-size-a-comprehensive-comparison-of-micromips32-and-thumb-code-size-using-many-megabytes-of-customer-code.

[KIA17] O. Khutsoane, B. Isong and A. M. Abu-Mahfouz. 'IoT devices and applications based on LoRa/LoRaWAN'. In: *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society* (2017), pp. 6107–6112.

[Kin+00] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra and M. Spasojevic. 'People, places, things: Web presence for the real world'. In: *Proceedings Third IEEE Workshop on Mobile Computing Systems and Applications*. Dec. 2000, pp. 19–28. DOI: 10.1109/MCSA.2000.895378.

[KKP99] J. M. Kahn, R. H. Katz and K. S. J. Pister. 'Next Century Challenges: Mobile Networking for "Smart Dust"'. In: *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking - MobiCom '99*. ACM Press, 1999. ISBN: 1581131429. DOI: 10.1145/313451.313558. URL: http://dx.doi.org/10.1145/313451.313558.

[Kni16] M. Knight. 'Reverse Engineering the LoRa PHY'. In: *PoC||GTFO. PASTOR LAPHROAIG'S MERCY SHIP HOLDS STONES FROM THE IVORY TOWER, BUT ONLY AS BALLAST!* 13 (Oct. 2016). Ed. by M. Laphroaig, Melilot, E. Sultanik, J. Torrey, A. Albertini, P. Teuwen and sundry others. URL: https://archive.org/details/pocorgtf013.

[KPN19] KPN. *LoRa connectivity for your connected devices*. 2019. URL: https://www.kpn.com/zakelijk/internet-of-things/en/lora-connectivity.htm.

[KS16] M. Knight and B. Seeber. 'Decoding LoRa: Realizing a Modern LPWAN with SDR'. In: *Proceedings of the 6th GNU Radio Conference*. Vol. 1. 1. CU Boulder, Boulder, CO, Sept. 2016. URL: https://pubs.gnuradio.org/index.php/grcon/article/view/8.

[LBV06] K. Langendoen, A. Baggio and O. Visser. 'Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture'. In: *Proceedings 20th IEEE international parallel & distributed processing symposium*. IEEE. 2006, 8–pp.

[Lev+03] P. Levis, N. Lee, M. Welsh and D. Culler. 'TOSSIM: Accurate and Scalable Simulation of entire tinyOS applications'. In: *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. Los Angeles, California, USA, 2003.

[Lib] Libellium. *Waspmote*. URL: http://www.libelium.com/products/waspmote/.

[Lib+18] O. Liberg, M. Sundberg, Y.-P. E. Wang, J. Bergman and J. Sachs. *Cellular Internet of Things*. Technologies, Standards and Performance. Academic Press, 2018. ISBN: 978-0-12-812458-1. DOI: https://doi.org/10.1016/C2016-0-01868-5.

[Lin] LinkLabs. *Symphony Link™ — Internet of Things LWPA*. URL: https://www.link-labs.com/symphony.

[LLV13] M. Li, Z. Li and A. V. Vasilakos. 'A Survey on Topology Control in Wireless Sensor Networks: Taxonomy, Comparative Study, and Open Issues'. In: *Proceedings of the IEEE* 101 (2013), pp. 2538–2557.

[LoR17a] LoRa Alliance. *LoRa Alliance™ Surpasses 500 Member Mark and Drives Strong LoRaWAN™ Protocol Deployments*. July 2017. URL: https://lora-alliance.org/in-the-news/lora-alliancetm-surpasses-500-member-mark-and-drives-strong-lorawantm-protocol.

[LoR17b] LoRa Alliance Technical Committee. *LoRaWAN 1.1 Regional Parameters*. Ed. by N. Sornin. Version 1.1A. Oct. 2017.

[Lün+18] O. Lünsdorf, S. Scherfke, K. Turner, K. G. Müller, T. Vignaux, J. Koomer, S. Kennedy, M. Grogan, S. Reed, C. Körner, A. Beham, L. Reis, P. Grayson and C. Klein. *SimPy: Discrete event simulation for Python*. Comp. software. 2018. URL: https://simpy.readthedocs.io/.

[Mai+02] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk and J. Anderson. 'Wireless sensor networks for habitat monitoring'. In: *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. WSNA '02. Atlanta, Georgia, USA: ACM, 2002, pp. 88–97. ISBN: 1-58113-589-0. DOI: 10.1145/570738.570751. URL: http://doi.acm.org/10.1145/570738.570751.

[Med16] MediaTek Labs. *LinkIt™ ONE*. 2016. URL: http://labs.mediatek.com/en/platform/linkit-one.

[Mic] Microchip. *Microchip AVR® MCUs*. URL: https://www.microchip.com/design-centers/8-bit/avr-mcus.

[MPH16] K. Mikhaylov, J. Petäjäjärvi and T. Haenninen. 'Analysis of Capacity and Scalability of the LoRa Low Power Wide Area Network Technology'. In: *European Wireless 2016; 22th European Wireless Conference*. May 2016, pp. 1–6. ISBN: 978-3-8007-4221-9.

[MS09] A. Mcgregor and D. Smithies. 'Rate Adaptation for 802.11 Wireless Networks: Minstrel'. 2009. URL: http://blog.cerowrt.org/papers/minstrel-sigcomm-final.pdf. unpublished paper.

[Net] NetBlocks. *NetBlocks XRange SX1272 LoRa RF Node*. URL: https://www.netblocks.eu/xrange-sx1272-lora-datasheet/.

[Nor15] J. P. Norair. *LoRa as a PHY for DASH7*. 29th Mar. 2015. URL: http://www.indigresso.com/_blog/?p=162.

[Ora12] Orange. *Orange continues the deployment of its LoRa® network to achieve national coverage in France*. 2012. URL: https://www.orange.com/en/Press-Room/press-releases/press-releases-2017/Orange-continues-the-deployment-of-its-LoRa-R-network-to-achieve-national-coverage-in-France.

[Öst+06] F. Österlind, A. Dunkels, J. Eriksson, N. Finne and T. Voigt. 'Cross-Level Sensor Network Simulation with COOJA'. In: *First IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp 2006)*. Nov. 2006.

[Ott+05] C. Otto, A. Milenković, C. Sanders and E. Jovanov. 'System architecture of a wireless body area sensor network for ubiquitous health monitoring'. In: *J. Mob. Multimed.* 1.4 (Jan. 2005), pp. 307–326. ISSN: 1550-4646. URL: http://dl.acm.org/citation.cfm?id=2010498.2010502.

[Par] Parallax. *Propeller*. URL: http://www.parallax.com/microcontrollers/propeller.

[PBD01] C. E. Perkins, E. M. Belding-Royer and S. R. Das. 'Ad hoc On-Demand Distance Vector (AODV) Routing'. In: *RFC* 3561 (2001), pp. 1–37.

[Per19] Pervasive Nation. 2019. URL: https://connectcentre.ie/pervasive-nation/.

[Pet+15] J. Petäjäjärvi, K. Mikhaylov, A. Roivainen, T. Hänninen and M. Pettissalo. 'On the coverage of LPWANs: range evaluation and channel attenuation model for LoRa technology'. In: *ITS Telecommunications (ITST), 2015 14th International Conference on*. Dec. 2015, pp. 55–59. DOI: 10.1109/ITST.2015.7377400.

[Pet+16] J. Petäjäjärvi, K. Mikhaylov, M. Hämäläinen and J. Iinatti. 'Evaluation of LoRa LPWAN technology for remote health and wellbeing monitoring'. In: *2016 10th International Symposium on Medical Information and Communication Technology (ISMICT)*. IEEE. Mar. 2016, pp. 1–5. DOI: 10.1109/ISMICT.2016.7498898.

[Pop+17] A.-I. Pop, U. Raza, P. Kulkarni and M. Sooriyabandara. 'Does Bidirectional Traffic Do More Harm Than Good in LoRaWAN Based LPWA Networks?' In: *GLOBECOM 2017*. 2017 IEEE Global Communications Conference (Singapore, 4th–8th Dec. 2017). 2017, pp. 1–6. ISBN: 978-1-5090-5019-2. DOI: 10.1109/GLOCOM.2017.8254509.

[PSC05] J. Polastre, R. Szewczyk and D. Culler. 'Telos: enabling ultra-low power wireless research'. In: *Proceedings of the 4th international symposium on Information processing in sensor networks*. IPSN '05. Los Angeles, California: IEEE Press, 2005. ISBN: 0-7803-9202-7. URL: http://dl.acm.org/citation.cfm?id=1147685.1147744.

[Rap02] T. S. Rappaport. *Wireless Communications: Principles and Practice*. 2nd ed. Prentice Hall, 2002. ISBN: 978-0-13-042232-3.

[RGS18] H. Rahim, C. Ghazel and L. A. Saïdane. 'An Alternative Data Gathering of the Air Pollutants In the Urban Environment using LoRa and LoRaWAN'. In: *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)* (2018), pp. 1237–1242.

[RH10] G. F. Riley and T. R. Henderson. 'Modeling and Tools for Network Simulation'. In: ed. by K. Wehrle, M. Güneş and J. Gross. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. Chap. The ns-3 Network Simulator, pp. 15–34. ISBN: 978-3-642-12331-3. DOI: 10.1007/978-3-642-12331-3_2.

[San05] P. Santi. 'Topology control in wireless ad hoc and sensor networks'. In: *ACM Comput. Surv.* 37 (2005), pp. 164–194.

[Sem] Semtech Corporation. *LoRa Calculator: fast evaluation of link budget, time on air and energy consumption.* URL: http://www.semtech.com/apps/filedown/down.php?file=SX1272LoRaCalculatorSetup1'1.zip.

[Sem12] Semtech Corporation. *Semtech Acquires Wireless Long Range IP Provider Cycleo.* 2012. URL: http://investors.semtech.com/releasedetail.cfm?ReleaseID=655335.

[SH16] R. Smith and J. Ho. *A9's CPU: Twister - The Apple iPhone 6s and iPhone 6s Plus Review.* AnandTech. Nov. 2016. URL: https://www.anandtech.com/show/9686/the-apple-iphone-6s-and-iphone-6s-plus-review/4.

[Sik+07] P. Sikka, P. Corke, L. Overs, P. Valencia and T. Wark. 'Fleck - A platform for real-world outdoor sensor networks'. In: *2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information.* Dec. 2007, pp. 709–714. DOI: 10.1109/ISSNIP.2007.4496930.

[Sir17] Siradel. *Siradel S_IoT.* 2017. URL: http://www.siradel.com/portfolio-item/alliance-lora/.

[Son+08] J. Song, S. Han, A. Mok, D. Chen, M. Lucas, M. Nixon and W. Pratt. 'WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control'. In: *Proceedings of the 2008 IEEE Real-Time and Embedded Technology and Applications Symposium.* RTAS '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 377–386. ISBN: 978-0-7695-3146-

5. DOI: 10.1109/RTAS.2008.15. URL: http://dx.doi.org/10.1109/RTAS.2008.15.

[Sor+17] N. Sornin, A. Yegin, A. Bertolaud, J. Delcef, V. Delport, P. Duffy, F. Dyduch, T. Eirich, L. Ferreira, S. Gharout, O. Hersent, A. Kasttet, D. Kjendal, V. Kleban, J. Knapp, T. Kramp, M. Kuyper, P. Kwok, M. Legourierec, C. Levasseur, M. Luis, M. Pauliac, P. Pietri, D. Smith, R. Soss, T. Tashiro and P. Thomsen. *LoRaWAN 1.1 Specification*. Ed. by N. Sornin. Version 1.1. Oct. 2017.

[SPF18] M. Slabicki, G. Premsankar and M. D. Francesco. 'Adaptive configuration of lora networks for dense IoT deployments'. In: *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium* (2018), pp. 1–9.

[Sri+06] K. Srinivasan, P. Dutta, A. Tavakoli and P. Levis. 'Understanding the Causes of Packet Delivery Success and Failure in Dense Wireless Sensor Networks'. In: *SenSys '06*. Proceedings of the Fourth International Conference on Embedded Networked Sensor Systems. SenSys '06 (Boulder, Colorado, USA, 31st Oct.–3rd Nov. 2006). New York, NY, USA: ACM, 2006, pp. 419–420. ISBN: 1-59593-343-3. DOI: 10.1145/1182807.1182885.

[SS13] O. B. A. Seller and N. Sornin. 'Low power long range transmitter'. European pat. EP2763321A1. Semtech Corporation. 5th Feb. 2013.

[Tal+07] I. Talzi, A. Hasler, S. Gruber and C. Tschudin. 'PermaSense: Investigating Permafrost with a WSN in the Swiss Alps'. In: *Proceedings of the 4th Workshop on Embedded Networked Sensors*. EmNets '07. Cork, Ireland: ACM, 2007, pp. 8–12. ISBN: 978-1-59593-694-3. DOI: 10.1145/1278972.1278974. URL: http://doi.acm.org/10.1145/1278972.1278974.

[Tex] Texas Instruments. *MSP430™ ultra-low-power MCUs*. URL: http://www.ti.com/microcontrollers/msp430-ultra-low-power-mcus/overview.html.

[The19a] The Things Network. *10k Connected Gateways*. 2019. URL: https://www.thethingsnetwork.org/article/10k-connected-gateways.

[The19b] The Things Network. *The Things Network: Building a global open LoRaWAN™ network*. 2019. URL: https://www.thethingsnetwork.org/.

[Trü+19] R. Trüb, R. D. Forno, T. Gsell, J. Beutel and L. Thiele. 'A Testbed for Long-Range LoRa Communication: Demo Abstract'. In: *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*. IPSN '19. Montreal, Quebec, Canada: Association for Computing Machinery, 2019, pp. 342–343. ISBN: 9781450362849. DOI: 10.1145/3302506.3312484. URL: https://doi.org/10.1145/3302506.3312484.

[TW11] A. S. Tanenbaum and D. Wetherall. *Computer Networks*. 5th ed. Pearson, 2011. ISBN: 0132553171.

[Var01] A. Varga. 'The OMNeT++ discrete event simulation system'. In: *Proceedings of the European simulation multiconference (ESM'2001)*. Vol. 9. S 185. 2001, p. 65.

[VL03] T. Van Dam and K. Langendoen. 'An adaptive energy-efficient MAC protocol for wireless sensor networks'. In: *Proceedings of the 1st international conference on Embedded networked sensor systems*. 2003, pp. 171–180.

[Voi+17] T. Voigt, M. Bor, U. Roedig and J. M. Alonso. 'Mitigating Inter-Network Interference in LoRa Networks'. In: *International Conference on Embedded Wireless Systems and Networks (EWSN) 2016*. Proceedings. MadCOM 2017 (Uppsala, Sweden, 20th–22nd Feb. 2017). Ed. by P. Gunningberg, T. Voigt, L. Mottola and C. Lu. Canada: Junction Publishing, Feb. 2017. ISBN: 978-0-9949886-1-4.

[VST17] H. Visser, J. Stokking and T. Telkamp. *Implement Adaptive Data Rate #430*. 2017. URL: https://github.com/TheThingsNetwork/ttn/pull/430.

[Whi+05] K. Whitehouse, A. Woo, F. Jiang, J. Polastre and D. Culler. 'Exploiting the Capture Effect for Collision Detection and Recovery'. In: *Embedded Networked Sensors, 2005. EmNetS-II. The Second IEEE Workshop on*. May 2005, pp. 45–52. DOI: 10.1109/EMNETS.2005.1469098.

[Win62] M. Winkler. 'Chirp signals for communications'. In: *IEEE WESCON Convention Record*. Vol. 7. 1962.

[YHE04] W. Ye, J. Heidemann and D. Estrin. 'Medium access control with coordinated adaptive sleeping for wireless sensor networks'. In: *IEEE/ACM Transactions on networking* 12.3 (2004), pp. 493–506.

[Zai+18] N. A. B. Zainal, M. H. Habaebi, I. J. Chowdhury and M. R. Islam. 'Cluttered traffic distribution in LoRa LPWAN'. In: 2018.

[ZG03] J. Zhao and R. Govindan. 'Understanding Packet Delivery Performance in Dense Wireless Sensor Networks'. In: *SenSys '03*. Proceedings of the First International Conference on Embedded Networked Sensor Systems. SenSys '03 (Los Angeles, California, USA, 5th–7th Nov. 2003). New York, NY, USA: ACM, 2003, pp. 1–13. ISBN: 1-58113-707-9. DOI: 10.1145/958491.958493.

[ZK04] M. Zuniga and B. Krishnamachari. 'Analyzing the Transitional Region in Low Power Wireless Links'. In: *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004*. IEEE SECON 2004. First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004 (Santa Clara, CA, USA, 4th–7th Oct. 2004). Oct. 2004, pp. 517–526. DOI: 10.1109/SAHCN.2004.1381954.

[Zol] Zolertia. *Zolertia - Hardware Solutions for Internet of Things Applications*. URL: https://zolertia.io.

Appendix A

Publications & Tools

Some of the material presented in this thesis has previously appeared in a different form in published papers, articles, and tools. These publications and my contributions are detailed in this chapter.

OpenCL as Wireless Sensor Network Programming Abstraction

Contributions I proposed and worked out the idea, and did most of the evaluation. I am the main author of the paper.

Full Citation M. Bor and U. Roedig. 'OpenCL as Wireless Sensor Network programming abstraction'. In: *IWOCL '14*. Proceedings of the International Workshop on OpenCL 2014. IWOCL 2014 (Bristol, United Kingdom, 12th–13th May 2014). Ed. by S. McIntosh-Smith and B. Bergen. AMD et al. New York, NY, USA: ACM, May 2014. ISBN: 978-1-4503-3007-7

Lifetime Bounds of Wi-Fi Enabled Sensor Nodes

Contributions I contributed to the idea of the paper; implemented, performed all the experiments and analysed the results. I am the main author of the paper.

Full Citation M. Bor, A. King and U. Roedig. 'Lifetime bounds of Wi-Fi enabled sensor nodes'. In: *Procedia Computer Science* 52 (June 2015): *The 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015)*. Ed. by E. Shakshuki, pp. 1108–1113. ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.05.127

LoRa for the Internet of Things

Contributions I contributed to the idea of the paper; designed, implemented, and performed the experiments. Provided major parts of the implementation and evaluation of LoRaBlink. I am the main author of the paper.

Full Citation M. Bor, J. E. Vidler and U. Roedig. 'LoRa for the Internet of Things'. In: *International Conference on Embedded Wireless Systems and Networks (EWSN) 2016*. Proceedings. MadCOM 2016 (Graz, Austria, 15th–17th Feb. 2016). Ed. by K. Römer, K. Langendoen and T. Voigt. Canada: Junction Publishing, Feb. 2016, pp. 361–366. ISBN: 978-0-9949886-0-7. URL: http://dl.acm.org/citation.cfm?id=2893711.2893802

Do LoRa Low-Power Wide-Area Networks Scale?

Contributions I contributed to the idea of the paper; I performed the measurements and determined the model. Implemented major parts of LoRaSim, and performed most of the simulations and data analysis. I am the main author of the paper.

Full Citation M. Bor et al. 'Do LoRa Low-Power Wide-Area Networks Scale?' In: *MSWIM '16*. Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. MSWIM '16 (Malta, Malta, 13th–17th Nov. 2016). ACM SIGSIM. New York, NY, USA: ACM Press, Nov. 2016, pp. 59–67. ISBN: 978-1-4503-4502-6. DOI: 10.1145/2988287.2989163

Mitigating Inter-Network Interference in LoRa Networks

Contributions I contributed to the idea of the paper and implemented parts of the improved model into LoRaSim.

Full Citation T. Voigt et al. 'Mitigating Inter-Network Interference in LoRa Networks'. In: *International Conference on Embedded Wireless Systems and Networks (EWSN) 2016*. Proceedings. MadCOM 2017 (Uppsala, Sweden, 20th–22nd Feb. 2017). Ed. by P. Gunningberg et al. Canada: Junction Publishing, Feb. 2017. ISBN: 978-0-9949886-1-4

LoRa Transmission Parameter Selection

Contributions I contributed to the idea of the paper. Designed, implemented, and performed the experiments. Designed, implemented, and analysed the model. I am the main author of the paper.

Full Citation M. Bor and U. Roedig. 'LoRa Transmission Parameter Selection'. In: *2017 International Conference on Distributed Computing in Sensor Systems (DCOSS)*. DCOSS 2017 (Ottawa, Canada, 5th–7th June 2017). IEEE, June 2017. DOI: 10.1109/DCOSS.2017.10

LoRaSim

Contribution I contributed to the idea, and implemented the majority of the software. I am the main author of the software.

Full Citation M. Bor and T. Voigt. *LoRaSim*. Version 0.2.1. 10th June 2017. URL: https://www.lancaster.ac.uk/scc/sites/lora/lorasim.html

LoRaBlink Kit

Contributions I proposed the work, and did the implementation. Some parts of the software were implemented by IBM Corporation and distributed under licence the Eclipse Public License v1.0. I am the main author of the software.

Full Citation M. Bor. *LoRaBlink Kit*. Version 20151214. 14th Dec. 2015. URL: https://www.lancaster.ac.uk/scc/sites/lora/lorablinkkit.html

Acronyms

3GPP	3rd Generation Partnership Project
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
ACK	Acknowledgement
ADR	Adaptive Data Rate
AFA	Adaptive Frequency Agility
AP	access point
ASK	Amplitude-Shift Keying
BLE	Bluetooth Low Energy
BPSK	Binary Phase-Shift Keying
BSS	Basic Service Set
BW	Bandwidth
CAD	Carrier Activity Detection
CATF	Campus Area Testbed Framework

CCA	Clear Channel Assessment
CDMA	Code Division Multiple Access
CF	Carrier Frequency
CFR	Code of Federal Regulations
COTS	Commercial Off-The-Shelf
CPS	Cyber-physical System
CPU	Central Processing Unit
CR	Coding Rate
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier-sense Multiple Access with Collision Avoidance
CSS	Chirp Spread Spectrum
DER	Data Extraction Rate
DFU	Device Firmware Update
DPSK	Differential Quadrature Phase Shift Keying
DR	Data Rate
DS	Distribution System
DS-UWB	Direct-Sequence Ultra Wide Band
DSP	Digital Signal Processor
DSSS	Direct-Sequence Spread Spectrum
EBR	Effective Bitrate
EC-GSM-IoT	Extended Coverage GSM IoT
eDRX	Extended Discontinuous Reception
EEPROM	Electrically Erasable Programmable ROM
EIRP	Effective Isotropic Radiated Power

eMTC	Enhanced Machine-Type Communication
ESS	Extended Service Set
ETSI	European Telecommunications Standards Institute
ETX	Expected Retransmissions
FCC	Federal Communications Commission
FEC	Forward Error Correction
FHSS	Frequency Hopping Spread Spectrum
FPGA	Field-Programmable Gate Array
GFSK	Gaussian Frequency-Shift Keying
GPIO	General-Purpose Input/Output
HSPA	High Speed Packet Access
HTTP	Hypertext Transfer Protocol
I^2C	Inter-Integrated Circuit
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
ISM	Industrial, Science, and Medical
ITU	International Telecommunication Union
JSON	JavaScript Object Notation
LBT	Listen Before Talk
LMiC	LoRaMAC in C

LoRa	Long Range
LoRaWAN	Long-Range Wide-Area Network
LPWAN	Low-Power Wide Area Network
LQE	Link Quality Estimator
LQI	Link Quality Indicator
LR-WPAN	Low-Rate Wireless Personal Area Network
LTE	Long-Term Evolution
LTE-MCM	LTE Machine Type Communication
M2M	Machine to Machine
MAC	Medium Access Control
MCU	Microcontroller Unit
MIMO	Multiple-Input Multiple-Output
NB-CIoT	Narrowband Cellular IoT
NB-IoT	Narrowband IoT
NB-LTE	Narrowband LTE
NEC	Network Energy Consumption
NFC	Near Field Communication
NS	Network Server
OFDM	Orthogonal Frequency-Division Multiplexing
OOK	On-off Keying
OQPSK	Offset-Quadrature Phase-Shift Keying
OTA	Over-the-Air
PA	power amplifier

PCR	Packet Corruption Rate
ppm	parts per million
PRR	Packet Reception Rate
PRU	Programmable Real-time Unit
PSM	Power Saving Mode
PSSS	Parallel-Sequence Spread Spectrum
QoS	Quality of Service
RAM	Random Access Memory
REST	Representational State Transfer
RISC	Reduced Instruction Set Computer
ROM	Read-Only Memory
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
SC-FDMA	Single Channel FDMA
SF	Spreading Factor
SNR	Signal-to-Noise Ratio
SoC	System on Chip
SPI	Serial Peripheral Interface
SRD	Short Range Device
STA	station
TCXO	temperature compensated crystal oscillator
TDMA	Time-Division Multiple Access
TP	Transmission Power

TTN	The Things Network
UART	Universal Asynchronous Receiver-Transmitter
UMTS	Universal Mobile Telecommunications System
USP	unique selling point
WEP	Wireless Equivalent Privacy
WirelessHART	Wireless Highway Addressable Remote Transducer Protocol
WLAN	Wireless Local Area Network
WoT	Web of Things
WPA	Wi-Fi Protected Access
WSN	Wireless Sensor Network

List of Figures

List of Tables