Lightweight learning algorithms for massive IoT and analysis of their performance

Ghina Dandachi, Yassine Hadjadj-Aoul, Patrick Maillé, Renzo Efrain Navas

Lightweight learning algorithms for massive IoT and analysis of their performance

Project Deliverable D.3.1

INTELLIGENTSIA project

Executive summary We address in this deliverable parameters optimization as well as the automation of devices configuration in massive IoT LoRaWAN scenarios. The utilization of automation techniques for devices configuration is a crucial evolution in IoT LoRa radio access in the way for network virtualization and automation. The challenges in LoRa radio access networks virtualization consists on partitioning the resources between different services and devices that are connecting in an ALOHA-like access.
We will investigate how to perform an automatic orchestration of radio resources between different devices. In particular, we will focus on a reducing the overhead required to ensure a good functioning of the automated devices configuration. We intend to (i) develop strategies enabling IoT devices automated configuration (ii) explore possible strategies enabling to follow a certain goal, such as maximize the energy efficiency, or the reliability, represented here by the Packet delivery ratio, and (iii) prepare a platform for service differentiation of different IoT slices.

Authors: Ghina Dandachi, Yassine Hadjadj-Aoul, Patrick Maille, Renzo Navas (INRIA Rennes - Bretagne Atlantique and University of Rennes 1, France)

Revised by: Tatiana Aubonnet (CNAM Paris), Alessandro Aimi (Orange Innovation, Châtillon)

Approved by:

Date: November 8, 2022

Contents

List of Figures

Chapter 1

General Introduction

This deliverable presents the work done in the task T3.1 entitled Automating IoT device configuration. This task focuses on having multiple IoT nodes learning which parameters to use (power, spreading factors, etc.) for uploading data, in order to optimize their global performance, i.e., taking into account the interactions among such nodes and possibly with other interfering devices. We focus on mechanisms where devices can make their own decision to avoid the communication overhead (i.e., downlink configuration frames) that would come with a centralized decision-making, while having them depending, at some extent, on states derived by gateway configuration and hence global automation. Hence, we plan not to hinder the end devices from having their strategy optimized globally by the orchestrator.

Besides, devices have some information not available to the network (in particular, packets whose transmission failed). But they face constraints in terms of computation, energy and memory, therefore the goal of the task will be to propose and analyze the performance of machine learning algorithms that need very little resources, like multi-armed bandit methods. The interactions among many devices implementing such methods (the so-called multiplayer multi-armed bandit problem) will need to be carefully integrated into the model. This can be done either over the L2 protocol or over the SCHC fragmentation protocol, which offers and optimizes feedback to inform devices of fragment losses.

1.1 Motivation

In LoRaWan networks, the configuration of the terminals is generally based on the reception quality of the gateways, through the Adaptive Data Rate (ADR) protocol that will be introduced later. This protocol allows, in fact, to configure the terminals efficiently, but could not be efficient if the conditions of the devices were unstable. In this paper we introduce a device-oriented approach for automatic configuration, in order to reduce the overhead, but more importantly to potentially find better configurations that is compliant with the device's objectives, these objectives might be to ensure a better energy efficiency, or to ensure a higher reliability of packets transmission. The approaches that we propose hereafter have the additional advantage of converging towards the optimum.

1.2 State of the art

Optimizing the parameters of wireless LoRaWan devices is crucial for its proper functioning as well as for its operational longevity. The optimization of these parameters is, however, a complex and tricky process. Indeed, the choice of a spreading factor, for example, affects the battery depletion speed, as well as the transmission rate, which in turn has an impact on competing communications.

In the literature, there are two classes of approaches for configuring LoRaWan devices: the first class includes centralized approaches and the other distributed approaches.

In the centralized class of approaches, we rely on the adjustment of transmission parameters via mainly the Network Server (NS). Basically, since the NS can receive the same packet from different gateways, it is the most appropriate for this type of decision. The most prominent approach in this class is certainly the Adaptive Data Rate (ADR) [28]. Based on the reception quality of a given number of packets originating from a LoRaWan device, the ADR calculates the optimal Spreading Factor (SF), the one minimizing the power consumption with a satisfying reception level. Several improvements have been proposed to enhance the performance of the ADR mechanism, considering for example other parameters such as the transmission power or the Coding Rate (CR) [35]. Although effective, these approaches do have some limitations. Indeed, the quality of reception does not only depend on the access conditions, which may fluctuate significantly, but also on the competing devices, which may themselves be subject to variations [16]. To address such an issue, several contributions have considered approaches optimizing the configuration of all the devices at the same time. In [17], the authors proposed the exact resolution of the problem for small network instances, given the NP-completeness of the problem, and a heuristic, from the family of water filling algorithms, for more realistic network sizes. Similarly, the authors, in [32], suggest a centralized coalition game-based strategy to manage LoRaWan nodes efficiently. However, all conceptual strategies of this class can present instabilities with non-controllable convergence times. Another known limitation, discussed among ADR papers [14], is that using radio quality metrics (RSSI, SNR) coming from received packets has an inherent bias because it does not considers lost packets. This happens for instance in [39] when averaging SNR measurements and it is difficult to overcome

The inherent limitations of centralized approaches have led to the interest in exploring the capability of distributed strategies (i.e., LoRaWan devices' centric) for the determination of the optimal devices' parameters. Here again, several strategies have been proposed in the literature. The standard itself proposes a terminal-based heuristic for the configuration of its parameters based on an incremental exploration of the different configurations [13]. However, the latter can only locally optimize the parameters of LoRaWan devices, without ensuring any guarantees in terms of stability or even efficiency. In this respect, and in order to achieve a global minimum, several reinforcement learning techniques have been proposed. In [10], a Q-learning-based strategy is proposed to derive the optimal devices' configuration. Even if this technique is centered on the devices, in the long run, it would allow converging to a global optimum, but the dimensions of the states' space and of the actions' space are such that the convergence is far from being guaranteed. In recent years, a number of contributions considered the use of multi-armed bandits' strategies [2]. These techniques are very effective and lightweight, even in non-stationary conditions [8]. Besides, these techniques are actually not necessarily in opposition with centralized ones.

1.3 Organization

The following chapters will be organized as follows: Chapter 2 will present the LoRaWAN architecture and parameters. Chapter 3 will introduce usage of MAC commands and reinforcement learning for parameters optimization in LoRaWAN. Chapter 4 will detail the Multi-armed bandit strategy proposed for end devices configuration automation and a strategy for slice-aware end devices configuration as well as the obtained results. Chapter 5 will provide a conclusion.

Chapter 2

LoRaWAN Architecture and Parameters

2.1 Introduction

2.1.1 LoRaWAN IoT

LoRaWAN is a Long-Range communication protocol often used to create Low Power Wide Area Networks (LPWANs) with an operating range that goes from 300 meters up to 10 kilometers. Among the main LPWANs protocol, LoRaWAN is one of the best known and used, given its open architecture and extremely low power consumption for the end devices. Talking about LoRa and LoRaWAN, it must be specified that the two terms refer to different things: LoRaWAN is a protocol in which the PHY layer is based on LoRa modulation while the Medium Access Control (MAC) layer is an open network architecture regulated by the LoRa Alliance. LoRa instead is a proprietary modulation based on Chirp Spread Spectrum (CSS) which refers to the Physical layer. LoRa protocol is also patented by Semtech Corporation, which is the only LoRa transceiver chips producer. Therefore, in the OSI protocol stack, LoRaWAN (Network Layer) relies on LoRa (Physical Layer), as visible in Figure 2.1, since it defines the network media access rules, the authentication method, device profile, and data encryption. Another difference between LoRa and LoRaWAN is the network topology, since LoRa allows only Point-To-Point links, while LoRaWAN, given its nature of Network Layers, defines all the needed rules to create a multiple stars network topology composed of many LoRaWAN end nodes and gateways. Gateways act as bridges between the LoRaWAN network and IP-based networks, delivering data from end nodes to one or more application servers and vice-versa.

2.1.2 LoRaWAN modulation

LoRa modems modulate symbols into increasing and decreasing frequency chirps, respectively called up-chirps and down-chirps.

Each LoRa transmission has a Preamble and a Start-Frame-Delimiter (SFD), which precede the encoded core data in order to initiate and lock the LoRa receiver, in order to correctly listen

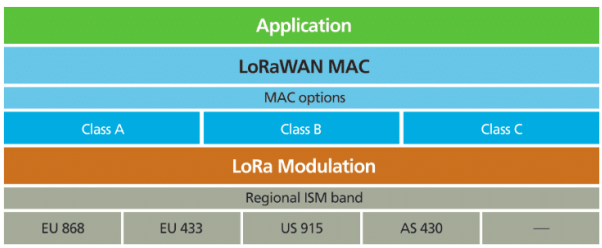


Figure 2.1: LoRaWAN vs LoRa ISO OSI layers

to the incoming transmission. In turn, SFD and Preamble have different polarities, so they use up-chirps and down-chirps, respectively, depending on these polarity settings.

LoRa modulation has many parameters, which can partially be modified, depending also on the operating region of the system. They are:

- **Carrier Frequency:** it defines the carrier frequency of the medium used for both transmission and listening operations. It also depends on the operating region: in Europe, the LoRa operating carrier frequency is the EU 863-870MHz ISM band, while in US it is the 902-928 MHz ISM band;

- **Signal Bandwidth:** it represents the width of the LoRa RF signals. It is typically set to 125kHz, but can be increased up to 250kHz or even 500kHz in some regions for particular modulations parameters;

- **Coding Rate:** it is a parameter that defines the Forward Error Correction (FEC) rate used by LoRa transmitters and receivers in order to reduce the destructive effects of RF interference. It affects the symbol airtime since it increases the symbol overhead to make it more noise-resistant. By default its value is set to $4/5$;

- **Spreading Factor:** it represents the chirp spreading parameter, defining how many chirps are sent per second. It ranges between SF7 and SF12. In detail, a large SF increases the symbol airtime and the energy consumption, thus improving the SNR resistance and, in turn, communication range, but reducing the available data rate and messages' maximum payload size;

- **Transmission Power:** it is the energy irradiated by the LoRa node's antenna. It can range from -4 dBm up to $+20$ dBm ($+14$ dBm in Europe), but different regions could have different power limits;

- **Chirp Polarity:** it defines the polarity of the transmitted chirps. It is often defined by the different protocols implementations. For example, LoRaWAN gateways transmit packets to end-nodes using an inverted polarity modulation, so that these messages are discarded by neighbor gateways, while end-devices transmit packets using non-inverted polarity, in order to be received only by the gateways;

- **Sync Word:** it is a one-byte value parameter defined by the last two up-chirps of the Preamble and used to differentiate LoRa networks using the same frequency bands. Any device configured with a given Sync Word will discard any incoming transmission if the Sync Word is different from the defined one. Typically, the Sync Word parameter for private LoRa networks are $0x12$ for Semtech $SX127x$ devices and $0x1424$ for $SX126x$ devices, while public LoRa networks (such as LoRaWAN or TTN) are represented by values equal to $0x34$ for Semtech $SX127x$ devices and $0x3444$ for $SX126x$ devices.

2.1.3 LoRaWAN device classes

LoRaWAN end nodes can also be classified into three categories: Class A, B, and C. All LoRaWAN devices must implement Class A, whereas Class B and C are extensions of Class A devices. These classes, define the behavior of downlink packets from gateways to end nodes (see Figure 2.2 for more details). A Class A device supports bi-directional communication but, while uplink messages can be sent at any time, downlink messages can be received only during two specific windows at specified times after an uplink transmission, allowing the lowest energy consumption mode. Class B devices are suitable for downlink-related activities since a time-synchronized receiving window is periodically opened through beacons. Class C devices instead, keep the receiving window open unless they are transmitting again, strongly increasing the power consumption. Usually, LoRaWAN gateways act as Class C devices, since they are

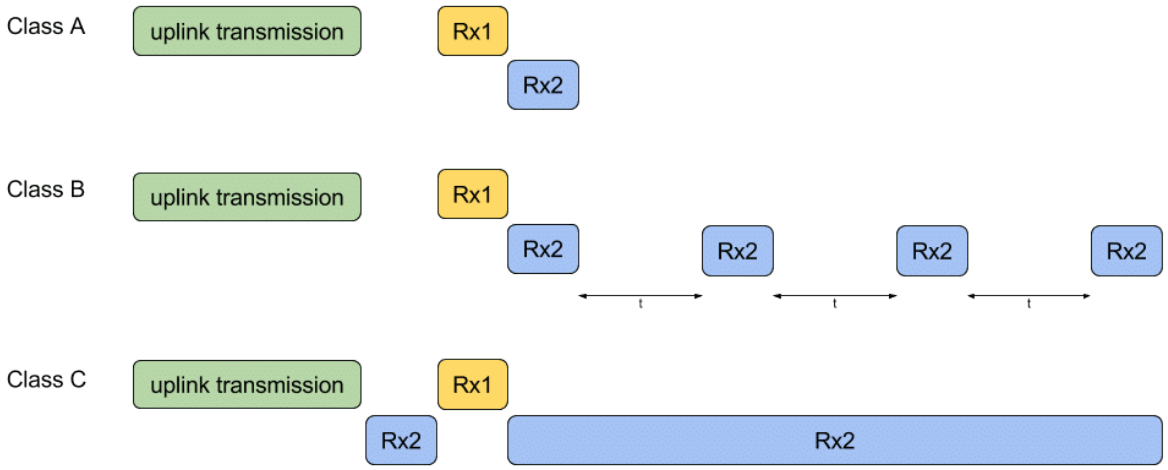


Figure 2.2: Devices classes in LoRaWAN

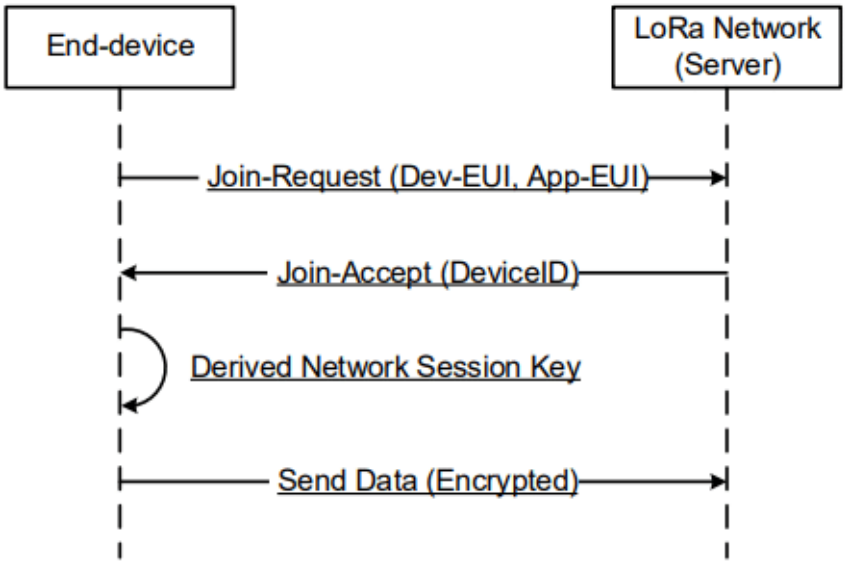


Figure 2.3: LoRaWAN over the Air Activation

constantly listening for incoming transmission. In order to transmit and receive data over the LoRaWAN network, LoRaWAN end nodes must be registered and enabled on the Application Server provider, which manages the open network gateways. Therefore, the LoRaWAN device can join the network in two ways: through an Over-The-Air-Activation (OTAA), or with an Activation-By-Personalization (ABP) method. Both methods work well, but the first one (see Figure 2.3) is more secure, since each time that the end node sends a join-request, it receives a join-accept with a NetID, DevAddr, and an AppNonce which are used by the device to generate a NwkSKey and AppSKey in a secure way. An ABP device instead, has already both DevAddr, AppSkey, and NwkSkey, which are sent over the network at each transmission in order to identify itself.

2.1.4 LoRaWAN payload, ranges, and bitrates

LoRa modulation is characterized by a Spreading Factor (SF) which defines the airtime duration of the chirp. Increasing the SF increases the symbol time and, as a consequence, the resistance to noise, allowing the signal to travel over a longer distance. Limiting the study to the free

Data rate [DR]	Spreading Factor [SF]	Bitrate [Bits/Sec]	Range [Km]	Rx Sensitivity [dBm]	Max Payload [Bytes]
0	12	290	12+	-136	51
1	11	440	10	-133	51
2	10	980	8	-132	51
3	9	1760	6	-129	115
4	8	3125	4	-126	222
5	7	5470	2	-123	222

Figure 2.4: LoRaWAN theoretical protocol parameters

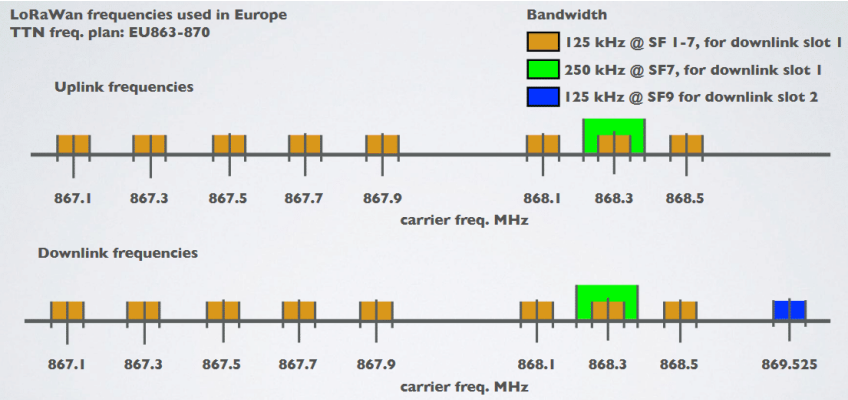


Figure 2.5: LoRaWAN channels and Frequencies table

868 MHz European band used by LoRa, SF can vary between 7 and 12, where SF equal to 7 allows the greater data rate and lower symbols airtime, while SF equal to 12 enables the highest sensitivity and transmission range with the lowest data rate and higher energy consumption, given by the longer transmission duration.

One of the most important things to know when working with LoRaWAN is the maximum packet payload for each SF. LoRaWAN network layer typically uses 13 bytes as packet header for the operation of the protocol, a not negligible value that at high SF significantly affects the maximum payload of the packet. Maximum payload size is reached with SF7, which allows up to 222 bytes of user's data inside a single LoRa packet. The minimum, instead, is reached with SF set to 12, with a limit of 51 bytes for the user's data. Payload limits, data rates, bitrates, receiver sensitivity, and typical operating range related to each SF with a bandwidth of 125 kHz at a carrier frequency of 868 MHz, are shown in the following Figure 2.4.

2.1.5 LoRaWAN European channels and frequencies

LoRaWAN protocol uses up to 16 uplink (from end nodes to gateways) channels defined inside the EU863-870 MHz free ISM band. Uplink channels can also be used as downlink channels on the first receiving window, but there is also another channel defined at the frequency of 869.525 MHz used only at downlink for the second receiving window. Both uplink and downlink channels of the EU863-870 MHz free ISM band are shown in Figure from The Things Netwrork configuration 2.5.

There is also another uplink channel, which has a fixed frequency of 868.8 MHz but uses an

FSK modulation. Another key feature of the LoRaWAN protocol is the Radio Medium Access: devices are tasked to randomly select one of the assigned frequencies for each new transmission. The resulting random access to the medium is similar to the one of the early ALOHA protocol, developed at the University of Hawaii back in the 1970s.

Duty cycle

Duty Cycle indicates the fraction of time a resource is busy. When a single device transmits on a channel for 1 time unit every 100-time units, this device has a duty cycle of 1%. However, if we also consider channels, things get a bit more complicated. When we have a device that transmits on 3 channels instead of one, each individual channel is still occupied for 1 time unit every 100 time units (so 1%). However, the device is now transmitting for 3 time units every 100-time units, giving it a duty cycle of 3%.

In our European frequency plan, we have channels in different sub-bands, so when considering the duty cycle, we also have to consider these. Let's say the 3 channels we used before, are in 2 different sub-bands. Each separate channel still has a duty cycle of 1%, and the device still has a duty cycle of 3%, but we now see that Band 1 is in use for 1-time unit every 100-time units (1%), while Band 2 is in use for 2-time units every 100-time units (2%).

The duty cycle of radio devices is often regulated by the government. If this is the case, the duty cycle is commonly set to 1%.

In Europe, duty cycles are regulated by section 7.2.3 of the ETSI EN300.220 standard. This standard defines the following sub-bands and their duty cycles:

- g (863.0 – 868.0 MHz): 1%
- g1 (868.0 – 868.6 MHz): 1%
- g2 (868.7 – 869.2 MHz): 0.1%
- g3 (869.4 – 869.65 MHz): 10%
- g4 (869.7 – 870.0 MHz): 1%

Additionally, the LoRaWAN specification dictates duty cycles for the join frequencies, defined as the frequencies that all the LoRaWAN devices use for over-the-air activations (OTAA). In most regions this duty cycle is set to 1%. Some regions specify the join frequency selection explicitly, such as in Europe. Other regions leave the join frequencies open to the available spectrum.

Fair Use Policy On the community public network of the Network of Things, a fair use policy applies limiting uplink airtime to 30 seconds per day (24 hours) per node and downlink messages to 10 messages per day (24 hours) per node. If one is using a private network, these limits do not apply, but one must still comply with government and LoRaWAN limits.

2.2 Interference Parameters

There are two sources of interference: from non-LoRa signals, and LoRa signals. For the first category, it was estimated [12] that a single tone pulse is not a problem if it is less than 5 dB (resp 19.5 dB) above the desired signal for SF = 7 (resp SF =12) with an error correcting scheme of 4/6. For the second category, one may note that two devices can not use the same SF, on the same frequency at the same time. Indeed, detection is a linear process. Thus, with two devices transmitting, the FFT output would provide the summation of each FFT, leading to 2 indiscernible peaks. The receiver would not be able to identify which offset to take into account. Nevertheless, one transmission over the two can be successful if one signal is received at least 6 dB above the other (Figure 2.6). Finally, we have computed and reported in Figure 2.6, the co-channel rejection when considering all couples of SF. We can observe that two devices using different spreading factors can transmit their data simultaneously, as long as none is received

with power significantly higher. We can also note that the rejection coefficient increases with the spreading factors. Thus, the high SF usually assigned to distant nodes for the noise sensitivity also permits overcoming the impact of closer devices that are likely to be received with a higher power level.

interferer desired	7	8	9	10	11	12
7	-6	16	18	19	19	20
8	24	-6	20	22	22	22
9	27	27	-6	23	25	25
10	30	30	30	-6	26	28
11	33	33	33	33	-6	29
12	36	36	36	36	36	-6

Figure 2.6: Co-channel rejection (dB) for all combinations of spreading factor for the desired and interferer user [19].

2.3 LoRaWAN Functional Architecture in INTELLIGENTSIA

The functional architecture considered in this project to model LoRaWAN networks is depicted in Figure 2.7 and is composed of:

- a substrate network layer comprising the radio gateways and the cloud infrastructure for hosting the VNFs associated with the LoRa network (LNS and Application servers)
- a telemetry layer collecting analytics from the substrate network and exposing synthesized metrics (after filtering, aggregation, etc.) to the orchestration layer;
- an orchestration layer for the management: lifecycle management of the VNFs, resource allocation, etc.

The control plane of the LoRa network is executed by vLNS. The LoRa-enabled server processes the data plane (similar to the UPF of 5G control plane), the Application server ensures the control of sessions (the equivalent of the AMF and SMF functions of the 5G control plane) and the Join Server authenticates devices (similar to the Authentication Server Function (AUSF) function of the 5G control plane).

The Virtual Infrastructure Management (VIM) of edge and centralized cloud platforms will ensure the deployment of the containers hosting the VNFs in collaboration with the orchestration layer. The VIM of a cloud platform is in charge of managing the resources of the platform. The orchestration may move some VNFs or require scaling up/down some functions when analyzing the measures from the network provided by the Telemetry layer aggregating LoRa analytics (radio metrics) and Prometheus analytics (from the cloud infrastructure). This loop is spanned over the Data Analytics Engines, the Optimization Function, and the Decision Engine of the orchestration layer.

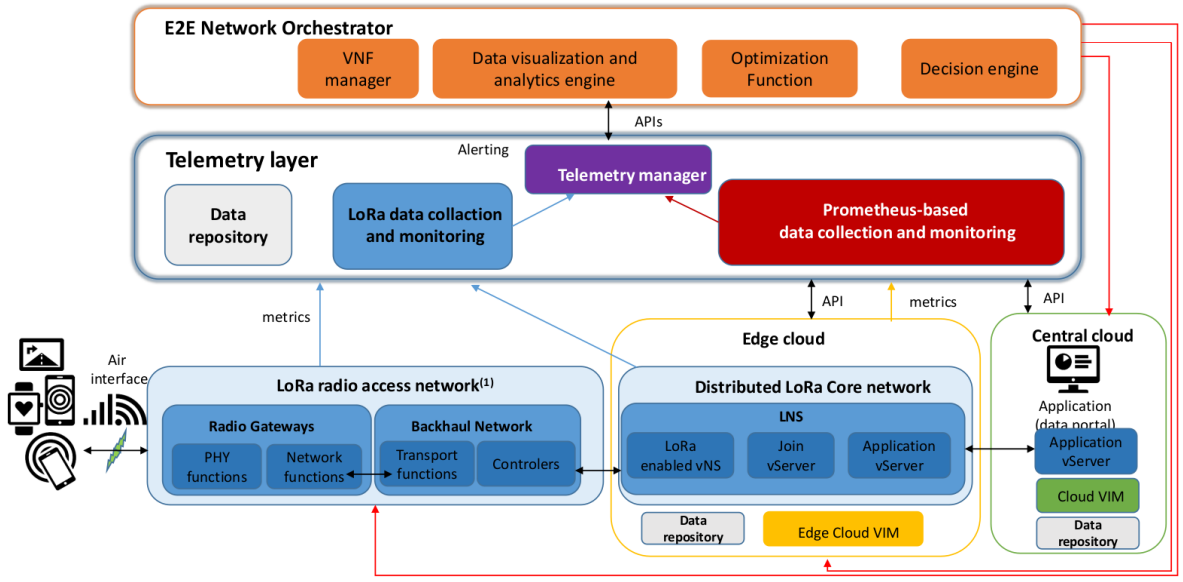


Figure 2.7: Reference functional architecture [22]

Chapter 3

Parameters Optimization in massive IoT Scenarios

3.1 Introduction

Our objective is having a large number of IoT nodes to learn what parameters to use (duty cycle, power, spreading factor, etc.) for uploading data, and optimizing their global performance. In this deliverable, we focus on the mechanisms where IoT nodes make their own decisions (decentralized). However, end devices may still have their strategy optimized globally by the orchestrator if the network server finds there is a need to improve the global performance metrics.

Given that the end nodes are generally on battery with limited autonomy, we propose and analyze the performance of machine learning algorithms that need few resources, such as multi-armed bandit methods.

We will start with a state-of-the-art presentation of the existing proposals to improve the LoRaWAN medium access, then we will propose a lightweight reinforcement learning strategy for devices' configuration. Multi-armed bandit is introduced with additional details on delayed feedback bandits.

3.2 LoRaWAN ADR Description

Technically, LoRaWAN operates in the Industrial, Scientific, and Medical (ISM) free band with duty cycle limitations with additional limitations from the network providers. Thus, LoRaWAN must have intelligent network management for the end nodes' transmission parameters to optimize the network performance. Generally, the mechanism deployed in LoRaWAN to achieve this goal is the Adaptive Data Rate (ADR).

ADR is an algorithm implemented at the network server-side or the end node side, that aims to improve the packet delivery ratio (PDR). ADR guides the end devices in the choice of transmission parameters such as: Spreading Factors (SF), Bandwidth (BW), Transmission Power (TP), and Coding Rate (CR). This guidance allows to optimize the network coverage, the end devices data rate, and energy efficiency. The data packets transmitted using different SFs are nearly orthogonal and can be transmitted concurrently [30]. It is also possible to increase the capacity of the network and its scalability by adding new gateways, for example the devices near new gateways can lower the SF.

There are two types of ADR depending on the end device mobility state [21]:

Static ADR In the case of stationary end nodes and stable radio channel environments, the NS manages the ADR depending on the history of the UL packets received. The LoRaWAN MAC layer contains four different elements related to ADR, shown in Table 3.1.

An end node may use any set of transmission parameters to communicate with the gateway without handshaking. Message transmission from gateways to end nodes occurs on a pro-

Table 3.1: LoRaWAN Adaptive Data Rate Primitives

Name	Type	Description
`ADR`	Uplink header bit	Enables the NS to control number of retransmissions, data rate, and TX power of the sender
`ADRACKReq`	Uplink header bit	Used by end nodes to periodically request confirmation that the NS is receiving UL messages
`LinkADRReq`	Downlink command	Sent by the NS, requests the end-device to change data rate, TX power, redundancy, or channel mask
`LinkADRAns`	Uplink command	End node acknowledgment to a `LinkADRReq`, indicating which command elements were accepted and which were rejected

grammable offset from the UL data rate in the first receiving window, and typically with the highly robust setting, the lowest data rate in the second receiving window.

An end node notifies the NS that it requires the use of ADR by configuring the ADR bit in the frame header. Once ADR is configured, the NS uses `LinkADRReq`, the MAC command that controls the end node's data rate and TP. The end node will respond with the `LinkADRAns` command to indicate acceptance or rejection of the new settings.

The ADR algorithm comprises of an acknowledgment system that is devised to permit end nodes to intermittently verify that the NS received the UL message. If an ACK message is not received by the end node, the end node will switch to a lower data rate in an attempt to regain connectivity.

Data rate backoff strategy In the case of mobile end nodes, the network-based ADR approach does not work because of channel attenuation when the device moves. The end nodes also have the capability of managing the ADR transmission parameters using the ADR system that is nested on the end node side. This means that, the ADR scheme can run asynchronously at the NS side and the end node side.

As stated in the LoRaWAN standard, there are two parameters that have been specified for this case, namely `ADR_ACK_LIMIT` and `ADR_ACK_DELAY`. The default values for these parameters have been set to 64 and 32, respectively.

For every UL packet that an end node transmits, `ADRACKCnt` counter is increased by one. Once the `ADRACKCnt` becomes equal to `ADR_ACK_LIMIT` without any DL response, the end node sets the `ADRACKReq` bit and waits for an ACK from the gateway for the subsequent `ADR_ACK_DELAY` UL packets.

In the absence of an ACK ahead of `ADR_ACK_DELAY` UL message, the end node decreases the data rate, attempting to re-establish network connectivity. In accordance with the latest release, end nodes initially increase TP to secure connectivity. If that is inadequate, the end nodes then reduce the data rate as an element of the subsequent stage.

However, ADR in LoRaWAN suffers from unfairness between end devices, slow convergence and high collision rates. This limitation results in a low LoRaWAN reliability and limited scalability. In order to address these challenges, several techniques have been proposed.

3.3 SotA/Positioning

3.3.1 LoRaWAN Metrics

The LoRaWAN ADR is a scheme that allows to optimize the network's data rates, ToA and energy consumption by controlling the data rate and TP for all the end nodes independently. In literature, several ADR schemes have been proposed in attempt to enhance the network performance metrics. Authors in [27] identified 20 articles on the subject of LoRaWAN's ADR. We synthesize the Metrics from those articles in Table 3.2.

Table 3.2: Synthesis of Metrics used on LoRaWAN's ADR Bibliography

Metric Name	Definition	NS wide	Per SF	Per Frq	Per GW	Per Node
Data Extraction Rate (DER) [9]	The ratio of received messages to transmitted messages over a period of time.	✓	✗‡	✗‡	✗‡	✓
Network Energy Consumption (NEC)[9]	Energy spent by the network to successfully extract a message†	✓	✓	✓	✓	✓
Packet Delivery Rate (PDR)	*—equivalent to DER metric—*					
Packet Delivery Ratio (PDRo)	$\frac{\#msg\_sent\_by\_nodes}{\#msg\_rcv\_NS}$	✓	✓	✓	✓	✓
Packet Error Rate (PER)	$\frac{\#packets\_crc\_error}{\#sent\_packets}$, over a period of time.	✓	✓	✓	✓	✓
Packet Loss Rate (PLR)	$\frac{\#lost\_packets}{\#sent\_packets}$, over a period of time†	✓	✗‡	✗‡	✗‡	✓
Packet Loss Ratio (PLRo)	$\frac{\#lost\_packets}{\#sent\_packets}$ (See§)	✓	✗‡	✗‡	✗‡	✓
Jain's Fairness index [17]	$\frac{(\sum_{i=1}^{n} x_i)^2}{n\sum_{i=1}^{n} x_i^2}$ (See¶)	✓	✓	✓	✓	✗

‡ Not applicable in a real deploy, because a lost packet can not be –easily– attributed to a 'PHY-link'. But applicable on simulation.
† Does not count energy spent on lost packets.
§ In the bibliography, PLRo is used as 1-PDRo: Does not discriminate CRC errors.
¶ NB1: x_i denotes the *normalized throughput* of each device and n the total number of active devices in each "slice". NB2: Index varies between $1/n$ and 1, with 1 being perfectly fair.

We start with the Data Extraction Rate (DER) metric, also known as the Packet Delivery Rate (PDR), introduced in [9]. It is defined as the ratio of received messages to transmitted messages over a period of time. This metric allows to evaluate the scalability of the network. The second metric is the Network Energy Consumption (NEC) defined as the energy spent by the network to successfully extract a message, however that does not count the energy spent on lost packets. Moreover, the Packet Delivery Ratio (PDRo) reflects the ratio between the number of messages sent by nodes to the number of messages received by the NS.

The Packet Error Rate (PER), presented in [36, 20], is defined as the number of packets received with a CRC error to the number of sent packets over a period of time.

The Packet Loss Rate (PLR) is the number of lost packets to the number of sent packets. The authors in [38] investigated the correlation between the number of end nodes and PLR in the LoRa network using one gateway. The Packet Loss Ratio (PLRo) is defined as $1 - \text{PDRo}$.

Finally, the use Jain's Fairness index of each slicing strategy is introduced in [17] and is calculated using the normalized throughput of each IoT device x_i and the total number of active devices n in each slice.

Empirical LoRaWAN measures from The Thing Network are introduced in [6] to justify different parameters in LoRaWAN such as SNR, data rates, time-on-air, packet loss, and rate of collision.

3.3.2 Proposals to Improve LoRaWAN Medium Access

In this subsection, we will present different techniques proposed to improve ADR's drawbacks, aiming mainly at reducing the collision rates.

ADR in LoRaWAN is dependent of the wireless condition, which is obtained from the reception status. However, the judgment of wireless condition based on the reception status of ACK messages does not reflect a congestion and leads to an erroneous ADR decision. Authors in [25] proposed a congestion classifier with a modified ADR, so that the data rate is controlled based on the congestion estimation of the system. A collision-aware ADR (CA-ADR) was proposed in [33] to minimize the collision probability when assigning data rates by considering the entire set of EDs in the network instead of assigns transmission parameters to EDs one-by-one. Authors in [41] proposed a new protocol to increase fairness and improve collision avoidance that acts on the medium access control protocol. This strategy proposes to broadcast becon frames from the gateways in order to synchronize the communication with devices.

However, these strategies are network server/gateway centric and require further communication between the end device and the gateway. Our proposed strategy is end device centric with low complexity and allows to the end devices to choose their SFs independently.

3.4 Lightweight Reinforcement-Learning Strategy for devices configuration

3.4.1 Basic Bandit problem

Multi-armed bandit problems refer to situations where a decision-maker has to repeatedly select one option (called an arm) in a finite set, and then obtains some reward that depends on the arm choice. The vocabulary stems from gambling situations with coin slot machines (called one-armed bandits): the situation here is indeed similar to that of a gambler having to maximize their gain by choosing among several machines. A specificity of bandit problems is that one only observes the reward for the selected (played) arm, and not the other ones. This leads to a well-known trade-off in reinforcement learning, between *exploration* (playing each arm sufficiently to estimate its performance) and *exploitation* (playing the best arms to maximize gains).

When rewards are independently drawn from unknown arm-specific distributions (we then talk of stochastic bandits), some well-known algorithms managing the exploration-exploitation dilemma include UCB–that selects the arm with the highest upper bound of the confidence interval estimating the mean–and Thompson Sampling (TS) [40]–that uses beliefs to select the arm to play and updates those beliefs.

A metric often used for arm selection algorithms performance evaluation is the *regret*, that compares the cumulative reward from an algorithm to the cumulative reward given by the best arm: the ED keeps tracks of the number of packets sent per arm, and can calculate the number of packets lost per arm k. The goal is generally to keep that regret as low as possible over time, and Thompson Sampling has been shown to be optimal in that sense [11, 1]. UCB also offers near-optimal performance [3].

3.4.2 Delayed feedback bandits

Delayed feedback refers to the decision maker not immediately observing the reward after pulling an arm; depending on the type and amplitude of the delay, the algorithms then need to be adapted, and their performance is affected.

According to authors in [23], a study on a Stochastic Black-box Bandit with Delays (SBBD) have proved that a a delay between 50 and 100 would be suitable for a time horizon of 1000 periods to converge to a similar non-delayed system. They also provided a comparison between the delayed Upper Confidence Bound (UCB) algorithms and their black box versions: For UCB1, the regrets of the two algorithms are very similar. For KL-UCB, while the situation is similar, the difference vanishes more slowly. The more interesting case is that the black-box versions seems to have worked better than the modified version. The difference, however, is insignificant for all of the simulations.

A stochastic bandit models for delayed conversions is proposed in [42], where the authors notice that the empirical performances of SBBD queuing-based heuristic are not fully satisfying because of the lack of variability in the decisions made by the policy while waiting for feedback. The authors also distinguish in [31] between online and batch updating. The distinctive features of the approach proposed in [42] is to consider potentially infinite stochastic delays, resulting in some feedback being censored (ie. not observable). Therefore, a possible way to handle bounded delays would be to plan ahead the sequence of pulls by batches, following the principles of Explore Then Commit.

Moreover, authors in [18] introduced the concept of "parallel selection". In this strategy, a Gaussian Process Batch Upper Confidence Bound (GP-BUCB), i.e. an upper confidence bound-

based algorithm, is considered which models the reward function as a sample from a Gaussian process. This allows to select batches of experiments to run in parallel.

3.4.3 Bandits for cognitive radio

Authors in [37] proposed stochastic competitive bandits. However, the proposed strategy does not apply for us because they assume that the number of players is always less than the number of arms. A study of more general congestion games in a game-theoretic setting was proposed in [26].

Authors in [4] proposed that selfish users need to coexist without any side communication between them, implicit cooperation or common control. Even the number of users may be unknown and can vary as users join or leave the network. They propose an algorithm that combines an e-greedy learning rule with a collision avoidance mechanism called MEGA.

Authors in [24] proposed a light-weight learning algorithm, namely the multi-armed bandit algorithm, for nodes to select the communication parameters.

In [7], the author did not consider lora but only narrowband. Then, in more recent studies [5, 34], they consider a LoRa testebed. However, they focus on Frequency Selecion (IoTligent), not Spreading Factor and no information is provided regarding the protocol overhead despite the ACK messages per packet, which results in an unrealistic setting.

3.5 Conclusion

In this chapter, we introduced the Adaptive Data Rate strategy in LoRaWAN for parameters optimization while detailing both static ADR and data rate backoff strategies. A literature review is presented for LoRaWAN metrics and the works that were based on each of it, with the proposed strategies to improve ADR and reduce the collision rates between end devices. Finally, the bandit problem was introduced and more precisely the delayed feedback bandits as well as bandits for cognitive radio.

In order to propose a strategy that takes into consideration end devices' behavior, we present in the next chapter a new strategy for devices configuration with multi-armed bandit with their corresponding results analysis.

Chapter 4

Devices configuration with Multi-Armed Bandit

TA: ajout du texte pour structurer et introduire les deux contributions

4.1 Introduction

Each node in the IoT has to be implemented with a certain degree of automation. Such nodes can therefore collect information about their respective environments and make decisions for their own behaviors. In this case, the data (including instructions and information) transmitted amongst nodes in a fully distributed form, is restricted into a limited area, which decreases heterogeneity of the whole network. Such distributed architecture also enables the automatic configuration scheme to be scalable. Our approach is organized with respect to two contributions based on several automatic capabilities:

1. Energy-aware automatic devices configuration (see Sec. 4.2).
2. Slice-aware automatic devices configuration (see Sec. 4.3).

4.2 First contribution: Automatic devices configuration description

In the literature, proposals use strong hypotheses or simplified models, e.g., only one node uses RL, immediate feedback, independence of configuration parameters. These proposals tend to the theoretical-side. However, evaluation/comparison against non-bandit proposals will be a challenge. A differentiating factor of the proposed Bandit-based Algorithm(s) will be this applicability/evaluation in realistic LoRaWAN scenarios. We introduce the bandit rewards for automatic devices configuration in Sec. 4.2.2.

4.2.1 Introduction to bandits in the case of SF choice

Multi-armed bandit (MAB) problems are a class of sequential resource allocation problems concerned with allocating one or more resources among several alternative (competing) projects. The proposal consists on a Bandit Agent at the End Device, learning about k different LoRa PHY configurations (*arms*).

In this approach, we propose a Thompson Sampling (TS) Bandit without black-box adaptation and define six PHY configurations ($k = 6$ *arms*) that correspond to LoRa's SFs from 12 to 7. The TX power is fixed, and the PHY central frequency is selected randomly for each new transmission[1]; i.e., not subject to RL.

[1]TX power = 14dBm. Three frequency values: {868.1, 868.3, 868.5} MHz.

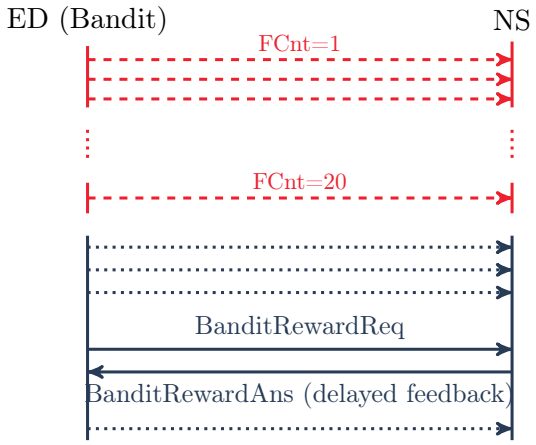


Figure 4.1: Overview of our proposal, with emphasis on feedback request. At the top, the initial phase of $b = 20$ messages without feedback request. Then, the long-term strategy, in which every uplink message can trigger a feedback request with a probability $p = 1/20$. In solid lines, an example of a successful feedback request and response.

We consider that the *feedback* or *reward* needs to be actively requested by the Bandit Agent and involves a two-message protocol between the End Device (ED) and the Network Server (NS). In the following, the *feedback request* mechanism is detailed and the *reward*'s function in Sec. 4.2.2.

Feedback request: triggering strategy

The frequency or strategy used for triggering feedback requests will impact greatly the overall network performance[2]. Our feedback request strategy consists of:

1. An initial phase of $b = 20$ application messages with no feedback request.
2. A long-term strategy where every application message can trigger a feedback request with probability $p = 1/20$ ($\sim$Bernoulli).

In Fig. 4.1, we illustrate our proposal from a single ED point of view, with an emphasis on the feedback request strategy.

Feedback request: protocol definition

To implement the delayed-feedback Bandit on a LoRaWAN network, we defined two custom MAC commands compliant with the LoRaWAN L2 1.0.4 Specification:

- `BanditRewardReq`: transmitted by an End-device to request rewards statistics (Size: 4 Bytes).
- `BanditRewardAns`: transmitted by a Network Server to send rewards statistics (Size: 7 Bytes).

Both commands are smaller than 15 Bytes and can be piggybacked with application data.

The `BanditRewardReq` command syntax can be seen in Table 4.1. The command will trigger a BanditRewardAns containing feedback/statistics on messages with a frame counter (FCnt) number between [`Max FCnt − Delta` , `Max FCnt`].

The `BanditRewardAns` command syntax can be seen in Table 4.2. In response to a `BanditRewardReq` (i.e., respecting the FCnt range), the command will answer with the number of packets coming from the device and received by the NS discriminated per SF $\in \{7, 8, 9, 10, 11, 12\}$—which corresponds to the k-arms ($k = 6$) of the Bandit.

[2]LoRA GWs are half-duplex (i.e., RX is not possible while TX) and are limited by regulatory duty-cycle restrictions (e.g., The Things Network's fair use: at most 10 downlink messages per 24 hours).

Table 4.1: MAC command: BanditRewardReq (4 Bytes)

Payload	Size (Bytes)
`CID (0xBB)`	1
`Max FCnt`	2
`Delta`	1

Table 4.2: MAC command: BanditRewardAns (7 Bytes)

Payload	Size (Bytes)
`CID (0xBB)`	1
`#Pkt_RCV SF12 (DR0)`	1
`#Pkt_RCV SF11 (DR1)`	1
`#Pkt_RCV SF10 (DR2)`	1
`#Pkt_RCV SF9 (DR3)`	1
`#Pkt_RCV SF8 (DR4)`	1
`#Pkt_RCV SF7 (DR5)`	1

For example, in the exchange of MAC commands:

1. ED→NS: `[0xBB, 0x01 0x00, 0x00]`
2. NS→ED: `[0xBB, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00]]`

The message (1) is a `BanditRewardReq` with (`Max FCnt = 1`[3], `Delta = 0`) and is requesting feedback statistics for Frame #1 to Frame #1. The message (2) is the `BanditRewardAns` with (`#Pkt_RCV SF9 = 1`, `0` for other fields) answering that the NS received 1 packet in SF9 and 0 in other SFs.

4.2.2 Bandit rewards

The *reward* definition synthesizes the optimization objective. Our *generic* reward definition is shown in Eq. 4.1 and has a per-packet nature (i.e., *a pull of the bandit arm ≡ a LoRa-PHY packet sent*).

$$reward = \begin{cases} r_k, \text{if packet received } (\textit{See. Table 4.3}) \\ 0, \text{otherwise} \end{cases} \tag{4.1}$$

We explored three *reward* definitions, one based on raw-PDR optimization (**PDR**) the second based on PDR with energy-efficiency considerations (**Energy-PDR**), and the third based on a combination between both PDR and energy-efficiency (**EAPA**). In **Energy-PDR**, each *arm* reward is inversely proportional to the energy used by the LoRa-PHY layer. In **EAPA**, the value of the arm reward is dynamic and defined as follows:

$$r_k = 25 * (PDR(k)^2) + R(ToA(k)) \tag{4.2}$$

with $PDR(k)$ being the ratio of packets received at SF k over the total packets transmitted, R being the reward proportional to the time-on-air for a given SF, and $ToA(k)$ is the time-on-air corresponding to a packet of 32 bytes. In Table 4.3, we show the values of r_k that must be used on Eq. 4.1 to instantiate our reward definitions for **PDR**, **Energy-PDR** and **EAPA** reward definitions.

In our setting, the *rewards* are calculated by the ED when the *delayed feedback* is received[4]. These (*delayed*) *rewards* are fed to an unmodified Thompson Sampling (TS) Bandit. The *reward* calculation and input is done as follows:

[3]The over-the-air octet order for all multi-octet fields is little endian.

[4]This in-node *reward* calculation allows for flexibility (e.g., each Bandit can have different optimization objectives).

Table 4.3: Packet Received Rewards r_k Definition

Optimization Objective	**Packet Received Reward r_k per arm k**					
	r_1 (SF12)	r_2 (SF11)	r_3 (SF10)	r_4 (SF9)	r_5 (SF8)	r_6 (SF7)
PDR	1	1	1	1	1	1
Energy-PDR	1	2	4	8	16	32
EAPA	$25PDR^2 + 1$	$2PDRp^2 + 1.8$	$25PDR^2 + 4$	$25PDR^2 + 7.3$	$25PDR^2 + 13.6$	$25PDR^2 + 25.2$

1. The `BanditRewardAns` feedback message contains the `Number of packets received per arm k` ($\cup_{k=1}^{6} RX_k$).
2. The ED keeps tracks of the number of packets sent per arm, and can calculate the `Number of packets lost per arm k` ($\cup_{k=1}^{6} N_k$).
3. Then, for every *arm* $k \in \{1, ..., 6\}$ and every packet sent ($RX_k \cup N_k$), its associated *reward* –as defined in Eq. 4.1– is fed into the TS Bandit.

4.2.3 Results

The source code of our implementation `LoRaWAN-Bandits` can be found on [GITLAB-INRIA]. Our proposal is based on the `ns-3`[5] discrete-event network simulator and the `LoRaWAN ns-3 module`[6][29].

In Table 4.4, we define common set-up parameters that will be used on all our simulation scenarios.

Table 4.4: Common configuration parameters for all experiments

Parameter	**Value**
PHY: End Device mobility	No mobility (static)
PHY: Propagation Loss Model	`LogDistancePropagationLossModel` (default)
LoRa-PHY: TX power	14 dBm (Except LoRaWAN ADR)
LoRa-PHY: Bandwith	125 kHz
LoRa-PHY: Frequency Carrier	$\mathcal{U} \in \{868.1, 868.3, 868.5\}$ MHz (EU868)
LoRa-PHY: Interference Matrix	Croce et al. [15]
APP: Application Packet Size	32 Bytes (45B@LoRA-PHY)
APP: Time Between Packets	20 min, w/initial delay = $\mathcal{U}[0, 20]$ min
Simulated Number of EDs	1000 (single-GW) / 2000 (multi-GW)
Simulated Events/Time	100 packets per ED $\sim$ 33h20m

Case of one gateway

In Fig. 4.2, we present a comparison between different strategies. The PDR variation as a function of periods is presented. The MAC responses feedback starts at 10 periods, this is were we see a little drop in the PDR value. We consider that each period a message is transmitted by the end device.

We observe, in Fig. 4.2a, that the legacy ADR achieves the highest value of system PDR and converges to this value around the message 20. A pure PDR strategy converges slowly with a lower PDR, as shown in Fig. 4.2c. As for EDs spatial distribution, we observe that with a pure PDR strategy, see Fig. 4.2d, the devices are more likely to select a higher SF than that proposed by ADR in Fig. 4.2b.

[5] https://www.nsnam.org/
[6] https://github.com/signetlabdei/lorawan

As for the Energy-PDR strategy, we observe a lower PDR in Fig. 4.2e with a choice of lower SFs for the devices compared to ADR, see Fig. 4.2f. Finally, the energy-aware-PDR-aware strategy (EAPA), we have a PDR similar to that obtained by ADR, see Fig. 4.2g, with a similar end devices spatial distribution in 4.2h.

Figures 4.3 and 4.4 show the performance comparison of delayed MAB under different rewards with ADR. This shows that a distributed decision with EAPA allows to have similar performance as ADR. However, when the energy-aware strategy is considered the ToA drops signifcantly. The end devices are more likely to select a lower SF that ADR, while reducing their energy consumption for packets transmission.

Case of multi-gateways

In this section, we present in Fig. 4.5 a comparison of MAB under different rewards with the ADR. We show the histogram of the SFs choice, instead of the PDR for the seven gateways deployed, as well as the end devices spatial distribution.

We observe with ADR in Fig. 4.5a, the SFs are mainly SF7 or SF12, with lower number of devices choosing the other SFs. In Fig. 4.5b, we can see that the devices choose SF7 when they are closer to a given gateway. The EAPA strategy presents a similar performance as ADR, see Figs. 4.5g and 4.5h.

In comparison with ADR, we observe in Figs. 4.5c and 4.5d that the devices choose the SFs in a more random fashion, while trying to choose the maximum possible SF. As for the energy aware strategy, we observe in Fig. 4.5e and 4.5f, similarly to the case of one gateway, the devices are more likely to choose the lowest possible SF, with a huge difference between the number of devices that chose the SF7 and the other SFs.

4.3 Second contribution: Slice-aware automatic devices configuration

4.3.1 Introduction to slice differentiation in LoRaWAN

In this contribution, we consider the SF configuration selection by each node as a function of the slice for which the node is communicating, i.e., the service ensured by this node.

We consider the same setting as before, the LoRaWAN simulator in ns-3. All the nodes are using delayed-MAB algorithm. However, what differs between two slices is the reward calculation that is used to updates the decision of the end node.

4.3.2 System Architecture

We consider a LoRaWAN network with network slicing. The main elements of this network are:

- End Devices: $D = \{1, 2, \cdots, d\}$. These devices may have heterogeneous QoS requirements in terms of reliability and energy efficiency depending on the IoT service provided.
- LoRa Gateways: $G = \{1, 2, \cdots, g\}$.
- LoRa Network Server (LNS) that is also connected to other application servers.

Network slicing in LoRaWAN allows to virtualize resouces allocation to specific slices with different objectives. Let $L = \{1, 2, \cdots, l\}$ be a set of slices each with specific priority. These slices will be sharing the GWs bandwidth. In this contribution, we consider the case of different slices objectives. More specifically, we consider two different requirements:

- Energy consumption
- Reliability

Both requirements can be also considered at the same time by some slices.

In the following three slices of different requirements are considered:

- G_f: A set of devices with reliability critical requirements. These devices are of higher priority.
- G_h: A set of devices with hybrid requirements in terms of reliability and energy efficiency, but are not critical, and are of lower priority than the first set G_f
- G_{BE}: A set of devices that does not require guarantees in term of reliability called "Best Effort". These devices have the lowest priority among all the devices.

4.3.3 Problem Formulation

In this contribution, the aim is to optimize the global performance of each slice $l_i \in L$ in terms of energy efficiency and reliability. It can be considered a multi-objective problem. We consider the case of d devices, g gateways, one LNS and l slices. We evaluate the reliability as the Packet Delivery Ratio (PDR) and the energy consumption considers only the energy used for packets transmission. The problem consists of:

$$\max_{\forall d \in G_f \vee d \in G_h \vee d \in G_{BE}} PDR(d) \tag{4.3}$$

$$\min_{\forall d \in G_{BE} \vee d \in G_h} E(d) \tag{4.4}$$

subject to:

$$\cap_{i=1}^{l} l_i = \emptyset, \forall l_i \in L \tag{4.5}$$

$$PDR(l_i) \geq PDR_{target} \forall l_i \in L \wedge \forall d \in l_i \Rightarrow d \in G_f \tag{4.6}$$

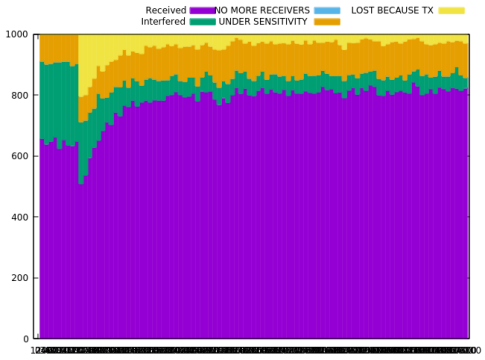
4.3.4 Results

The rewards updates follow the same definition presented in Table 4.3, with two slices of two different objectives: PDR optimization (slice 2), and Energy optimization (slice 1).

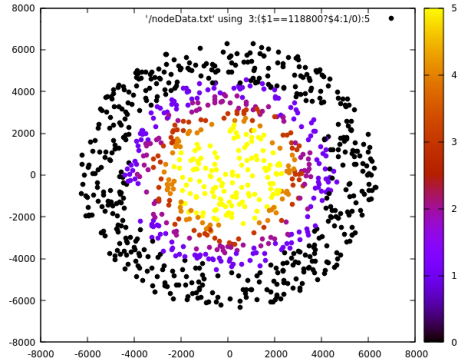
We observe in Fig. 4.6, the difference of PDR performance between both strategies for reward calculation. This allows to have, in the same LoRaWAN system, different goals for different devices based on their service requirement, such as: energy consumption optimization, and PDR maximization.

4.4 Conclusion

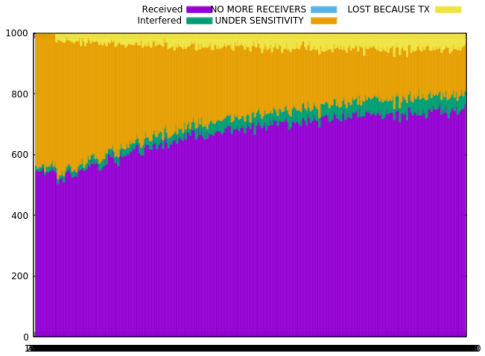
In this section, we presented the multi-armed bandit based strategy for SFs selection in LoRaWAN end devices. This strategy is decentralized and allows to each device to optimize its performance independently of the other devices in the network. An energy aware strategy was proposed for reward selection and decision taking. This allows to reduce the energy consumption at the end devices. The case of automatic configuration per slice was also presented, where the devices of different slices follow different automatic configuration strategies.



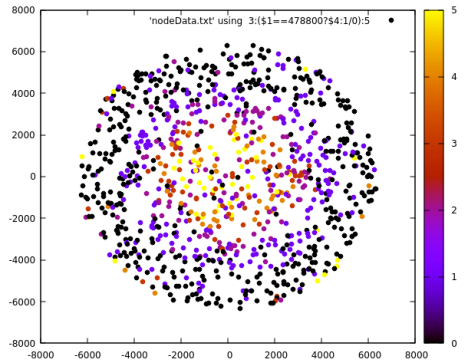
(a) Transmission outcomes in the case of one gateway using the ADR strategy.



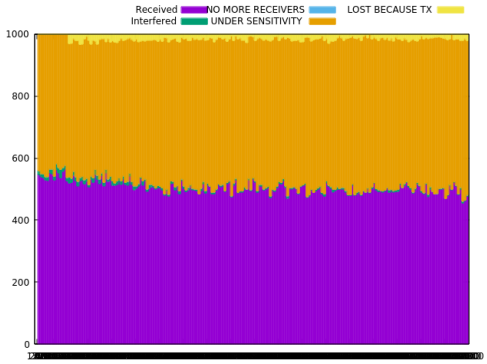
(b) End devices spatial distribution using the ADR strategy after convergence, case of 100 periods.



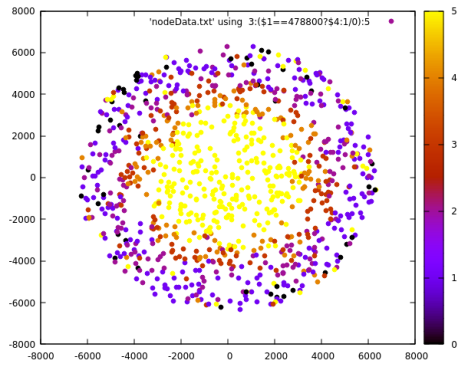
(c) Transmission outcomes in the case of one gateway using the PDR strategy.



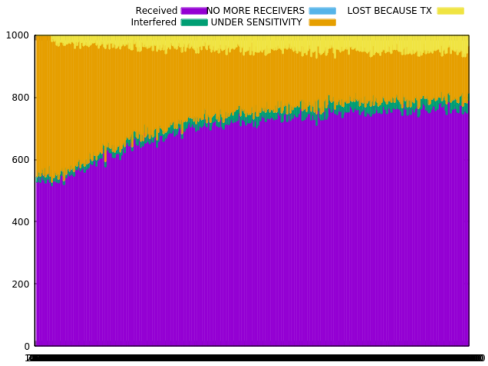
(d) End devices spatial distribution using the PDR strategy after convergence, case of 400 periods.



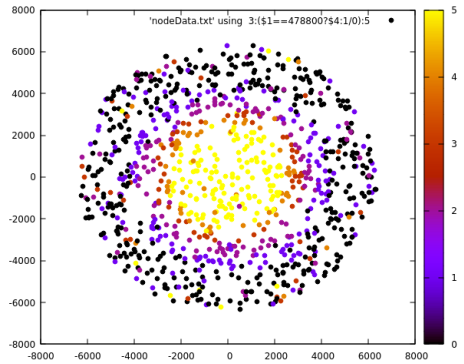
(e) Transmission outcomes in the case of one gateway using the Eenergy-PDR strategy.



(f) End devices spatial distribution using the Energy-PDR strategy after convergence, case of 400 periods.



(g) Transmission outcomes in the case of one gateway using the EAPA strategy.



(h) End devices spatial distribution using the EAPA strategy after convergence, case of 400 periods.

Figure 4.2: Comparison of different reward strategies with the ADR strategy, case of one gateway.

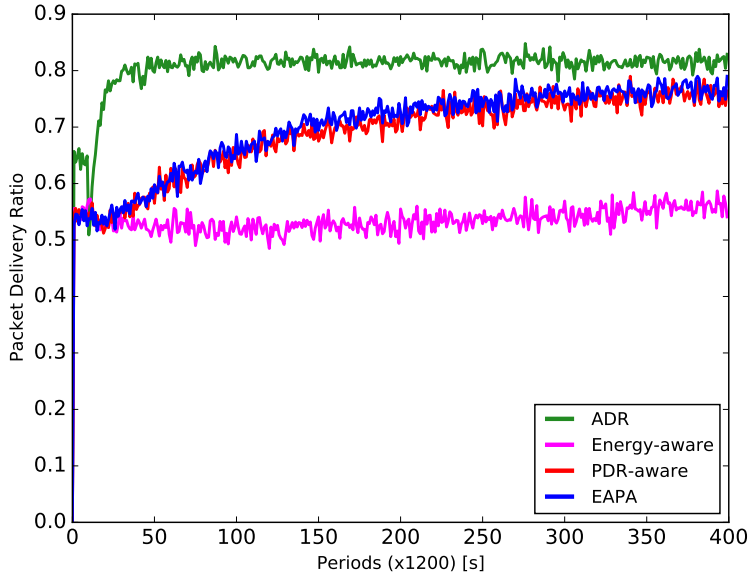


Figure 4.3: PDR variation for different bandit strategies compared with ADR.

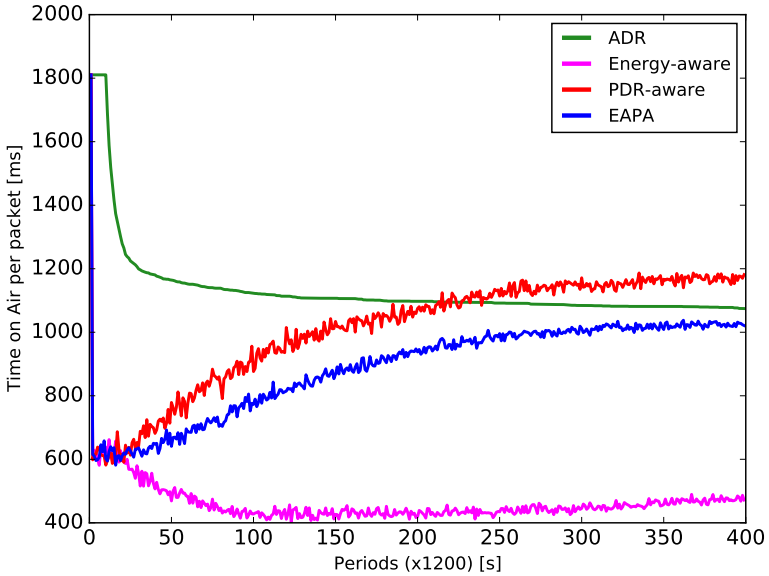
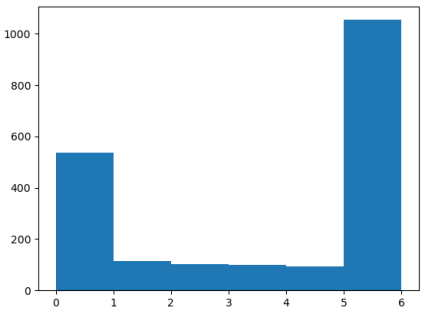
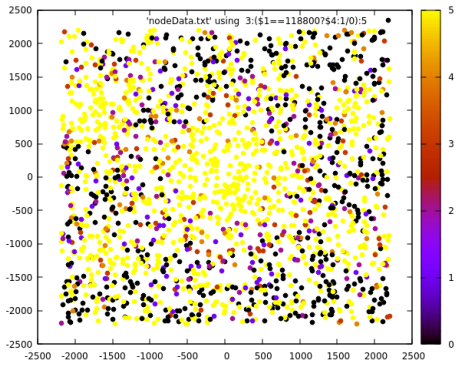


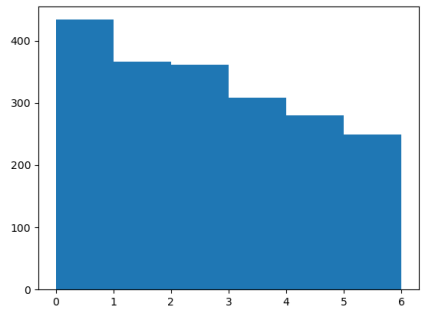
Figure 4.4: Time-on-Air variation for different bandit strategies compared with ADR



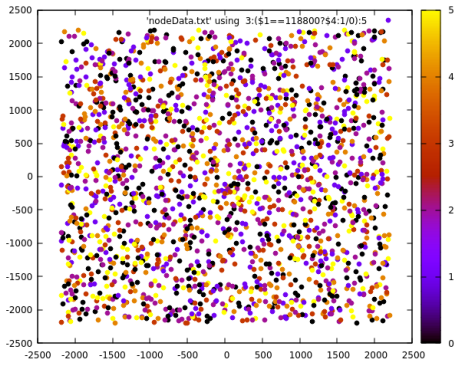
(a) SF distribution in the case of multi-gateway using the ADR strategy.



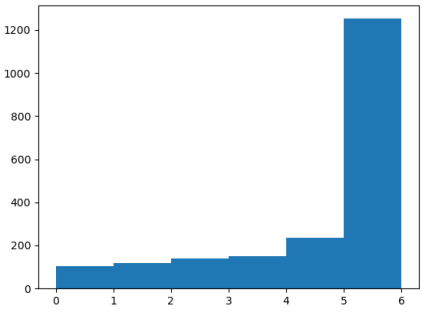
(b) End devices spatial distribution using the ADR strategy, case of 100 periods.



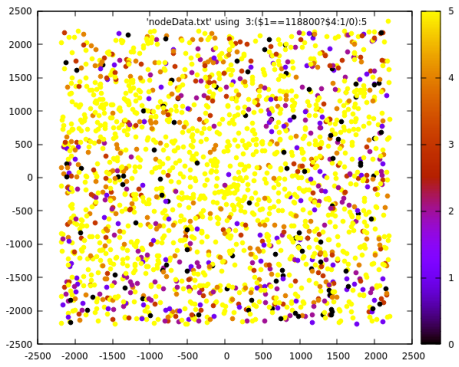
(c) SF distribution in the case of multi-gateway using the PDR strategy.



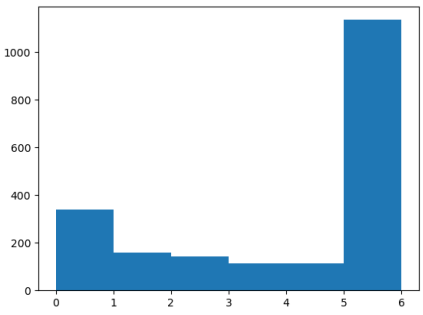
(d) End devices spatial distribution using the PDR strategy, case of 400 periods.



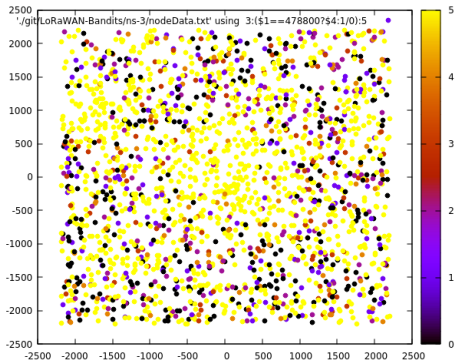
(e) SF distribution in the case of multi-gateway using the EA strategy.



(f) End devices spatial distribution using the EA strategy, case of 400 periods.



(g) SF distribution in the case of multi-gateway using the EAPA strategy.



(h) End devices spatial distribution using the EAPA strategy, case of 400 periods.

Figure 4.5: Comparison of different reward strategies with the ADR straegy, case of multi-gateway.

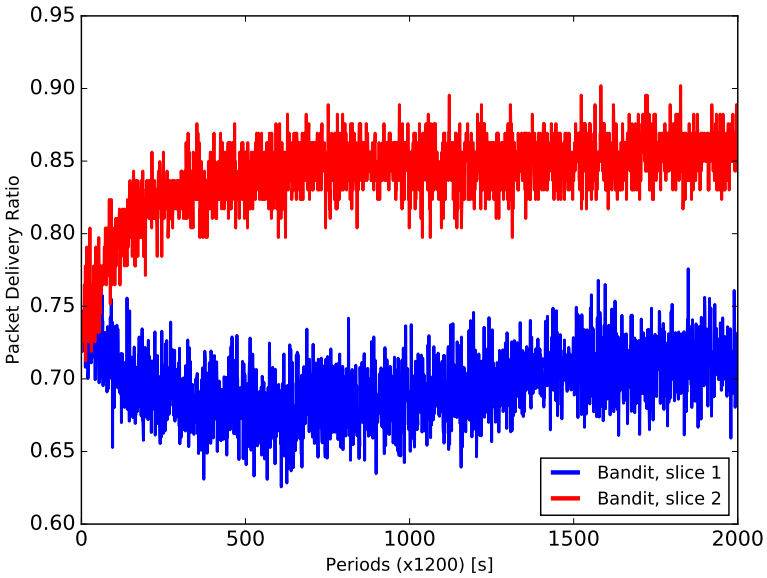


Figure 4.6: PDR evaluation for end devices of different slices

Chapter 5

Conclusion

This Deliverable presented the work done in the task T3.1 entitled Automating IoT device configuration. The proposed work, consists on using the multi-armed bandit strategy at the end devices side in order to optimize their global performance from a PDR or energy point of view. The solution proposed evaluated the case of spreading factor selection, but can also be extended to the other parameters such as the transmission power and the back-off time between two packets. The aim is to reduce the communication overhead, by decentralizing the parameters selection. However, we plan not to hinder the end devices from having their strategy optimized globally by the network server when needed.

Bibliography

[1] Shipra Agrawal and Navin Goyal. Analysis of Thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*. Journal of Machine Learning Research, 2012.

[2] Samar Adel Almarzoqi, Ahmed Yahya, Zaki Matar, and Ibrahim Gomaa. Re-learning exp3 multi-armed bandit algorithm for enhancing the massive iot-lorawan network performance. *Sensors*, 22(4), 2022.

[3] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[4] Orly Avner and Shie Mannor. Concurrent bandits and cognitive radio networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 66–81. Springer, 2014.

[5] Lilian Besson. *Multi-Players Bandit Algorithms for Internet of Things Networks*. PhD thesis, CentraleSupélec, 2019.

[6] Norbert Blenn and Fernando Kuipers. Lorawan in the wild: Measurements from the things network. *arXiv preprint arXiv:1706.03086*, 2017.

[7] Rémi Bonnefoi, Lilian Besson, Christophe Moy, Emilie Kaufmann, and Jacques Palicot. Multi-armed bandit learning in iot networks: Learning helps even in non-stationary settings. In *International Conference on Cognitive Radio Oriented Wireless Networks*, pages 173–185. Springer, 2017.

[8] Rémi Bonnefoi, Lilian Besson, Christophe Moy, Emilie Kaufmann, and Jacques Palicot. Multi-armed bandit learning in iot networks: Learning helps even in non-stationary settings. In Paulo Marques, Ayman Radwan, Shahid Mumtaz, Dominique Noguet, Jonathan Rodriguez, and Michael Gundlach, editors, *Cognitive Radio Oriented Wireless Networks*, pages 173–185, Cham, 2018. Springer International Publishing.

[9] Martin C Bor, Utz Roedig, Thiemo Voigt, and Juan M Alonso. Do lora low-power wide-area networks scale? In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 59–67. ACM, 2016.

[10] Rodrigo Carvalho, Faroq Al-Tam, and Noélia Correia. Q-learning adr agent for lorawan optimization. In *2021 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pages 104–108, 2021.

[11] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24:2249–2257, 2011.

[12] Semtech Corporation. Sx1272/73 - 860 mhz to 1020 mhz low power long range transceiver datasheet, 2015.

[13] Semtech Corporation. Lorawan – simple rate adaptation recommended algorithm, 2016.

[14] Ulysse Coutaud, Martin Heusse, and Bernard Tourancheau. Lora channel characterization for flexible and high reliability adaptive data rate in multiple gateways networks. *Computers*, 10(4):44, 2021.

[15] Daniele Croce, Michele Gucciardo, Stefano Mangione, Giuseppe Santaromita, and Ilenia Tinnirello. Impact of lora imperfect orthogonality: Analysis of link-level performance. *IEEE Communications Letters*, 22(4):796–799, 2018.

[16] Francesca Cuomo, Manuel Campo, Alberto Caponi, Giuseppe Bianchi, Giampaolo Rossini, and Patrizio Pisani. Explora: Extending the performance of lora by suitable spreading factor allocations. In *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–8, 2017.

[17] Samir Dawaliby, Abbas Bradai, and Yannis Pousset. Adaptive dynamic network slicing in LoRa networks. *Future Generation Computer Systems*, 98:697–707, September 2019.

[18] Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *Journal of Machine Learning Research*, 15:3873–3923, 2014.

[19] Claire Goursaud and Jean-Marie Gorce. Dedicated networks for IoT : PHY / MAC state of the art and challenges. *EAI endorsed transactions on Internet of Things*, October 2015.

[20] Vojtěch Hauser and Tomáš Hégr. Proposal of adaptive data rate algorithm for lorawan-based infrastructure. In *2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 85–90. IEEE, 2017.

[21] Jetmir Haxhibeqiri, Eli De Poorter, Ingrid Moerman, and Jeroen Hoebeke. A survey of lorawan for iot: From technology to application. *Sensors*, 18(11):3995, 2018.

[22] INTELLIGENTSIA Project. Reference architecture for slicing in lorawan networks, 2021.

[23] Pooria Joulani. Multi-armed bandit problems under delayed feedback. Master's thesis, Department of Computing Science, University of Alberta, 2012.

[24] Raouf Kerkouche, Réda Alami, Raphaël Féraud, Nadège Varsier, and Patrick Maillé. Node-based optimization of lora transmissions with multi-armed bandit algorithms. In *2018 25th International Conference on Telecommunications (ICT)*, pages 521–526. IEEE, 2018.

[25] Dae-Young Kim, Seokhoon Kim, Houcine Hassan, and Jong Hyuk Park. Adaptive data rate control in low power wide area networks for long range iot services. *Journal of Computational Science*, 22:171–178, 2017.

[26] Robert Kleinberg, Georgios Piliouras, and Eva Tardos. Multiplicative updates outperform generic no-regret learning in congestion games: Extended abstract. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 533–542, New York, NY, USA, 2009. Association for Computing Machinery.

[27] Rachel Kufakunesu, Gerhard P Hancke, and Adnan M Abu-Mahfouz. A survey on adaptive data rate optimization in lorawan: Recent solutions and major challenges. *Sensors*, 20(18):5044, 2020.

[28] Charles Lehong, Bassey Isong, Francis Lugayizi, and Adnan M. Abu-Mahfouz. A survey of lorawan adaptive data rate algorithms for possible optimization. In *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, pages 1–9, 2020.

[29] Davide Magrin, Martina Capuzzo, and Andrea Zanella. A thorough study of lorawan performance under different parameter settings. *IEEE Internet of Things Journal*, 7(1):116–127, 2019.

[30] Davide Magrin, Marco Centenaro, and Lorenzo Vangelista. Performance evaluation of lora networks in a smart city scenario. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–7, 2017.

[31] Travis Mandel, Yun-En Liu, Emma Brunskill, and Zoran Popović. The queue method: Handling delay, heuristics, prior data, and evaluation in bandits. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[32] Fatima Zahra Mardi, Miloud Bagaa, Yassine Hadjadj-Aoul, and Nabil Benamar. An efficient allocation system for centralized network slicing in lorawan. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 806–811, 2022.

[33] Riccardo Marini, Walter Cerroni, and Chiara Buratti. A novel collision-aware adaptive data rate algorithm for lorawan networks. *IEEE Internet of Things Journal*, 8(4):2670–2680, 2021.

[34] Christophe Moy, Lilian Besson, Guillaume Delbarre, and Laurent Toutain. Decentralized spectrum learning for radio collision mitigation in ultra-dense iot networks: Lorawan case study and experiments. *Annals of Telecommunications*, 75(11):711–727, 2020.

[35] Junhyun Park, Kunho Park, Hyeongho Bae, and Chong-Kwon Kim. Earn: Enhanced adr with coding rate adaptation in lorawan. *IEEE Internet of Things Journal*, 7(12):11873–11883, 2020.

[36] Brecht Reynders, Qing Wang, Pere Tuset-Peiro, Xavier Vilajosana, and Sofie Pollin. Improving reliability and scalability of lorawans through lightweight scheduling. *IEEE Internet of Things Journal*, 5(3):1830–1842, 2018.

[37] Jonathan Rosenski, Ohad Shamir, and Liran Szlak. Multi-player bandits–a musical chairs approach. In *International Conference on Machine Learning*, pages 155–163. PMLR, 2016.

[38] Ruben M. Sandoval, Antonio-Javier Garcia-Sanchez, and Joan Garcia-Haro. Optimizing and updating lora communication parameters: A machine learning approach. *IEEE Transactions on Network and Service Management*, 16(3):884–895, 2019.

[39] Mariusz Slabicki, Gopika Premsankar, and Mario Di Francesco. Adaptive configuration of lora networks for dense iot deployments. In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9. IEEE, 2018.

[40] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[41] Anna Triantafyllou, Panagiotis Sarigiannidis, Thomas Lagkas, Ioannis D. Moscholios, and Antonios Sarigiannidis. Leveraging fairness in lorawan: A novel scheduling scheme for collision avoidance. *Computer Networks*, 186:107735, 2021.

[42] Claire Vernade, Olivier Cappé, and Vianney Perchet. Stochastic Bandit Models for Delayed Conversions. In *Conference on Uncertainty in Artificial Intelligence*, Sydney, Australia, August 2017.