Energy-Aware Spreading Factor Selection in LoRaWAN Using Delayed-Feedback Bandits

Renzo E. Navas*, Ghina Dandachi†, Yassine Hadjadj-Aoul‡, Patrick Maillé§
*§IMT Atlantique, IRISA, UMR CNRS 6074. Rennes, France.
†‡Inria, Univ. Rennes, CNRS, IRISA. Rennes, France.
*renzo.navas@imt-atlantique.fr, †ghina.dandachi@inria.fr, ‡yassine.hadjadj-aoul@irisa.fr, §patrick.maille@imt.fr

***Abstract*—LoRaWAN networks can involve large numbers of wireless devices relying on batteries to sense the environment and send data to gateways. A critical trade-off for transmission performance (packet delivery ratio) versus energy conservation (and hence, the device lifespan) appears when deciding the transmission parameters, in particular, the Spreading Factor (SF) to be used by each node.**

In this paper, we use lightweight reinforcement learning techniques, namely multi-armed bandits, for each node to select an appropriate SF, based on preferences regarding that trade-off. Unlike previous works on that topic, we relax some assumptions to aim at a realistic implementation: our solution does not assume immediate rewards, or that each device communicates with only one gateway. Additionally, we build explicit MAC commands for the method to work in practice and implement it in the ns-3 simulator using a state-of-the-art LoRaWAN module. We share the source code of our implementation and our simulation results. Those simulations show that when energy conservation is critical for IoT nodes, such lightweight learning algorithms outperform LoRaWAN's legacy Adaptive Data Rate algorithm, both in single- and multi-gateway scenarios.

***Index Terms*—LoRaWAN, reinforcement learning, energy consumption, bandits, Adaptive Data Rate, ADR, Spreading Factor selection, IoT**

I. Introduction

Low Power Wide Area Network (LPWAN) technologies are among the enablers of Internet-of-Things (IoT) devices and services. They allow low-cost and energy-efficient transmissions to interconnect thousands of geographically dispersed end devices running on batteries. The interconnection between these devices is made possible by different technologies such as Long Range Wide Area Network (LoRaWAN), SIGFOX, and NB-IoT. Particularly, LoRaWAN is the most widely adopted due to its simplicity, openness, and cost-effectiveness.

LoRa is a network physical layer owned by Semtech. It uses the Chirp Spread Spectrum (CSS) technology and is designed as an energy-efficient, long-range, and low-power wireless platform. LoRaWAN [1], [2] is a Media Access Control (MAC) layer that runs atop the LoRa physical layer.

LoRaWAN deploys a simple star-of-stars network topology architecture for simplicity and cost-effectiveness. As shown in Figure 1, end devices connect to centralized gateways which forward messages to and from a remote Network Server (NS). In particular, LoRaWAN end devices are energy-efficient sensors and actuators featuring low data rates, ranging from 0.3 to 50 Kbps [3], with long-life batteries intended to last up to 10 years [1]. LoRaWAN already counts several hundred known use cases for smart cities, homes and buildings, communities, metering, supply chain and logistics, agriculture, and more.

ISBN 978-3-903176-57-7© 2023 IFIP

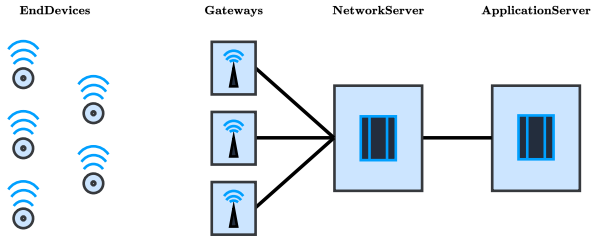

Fig. 1. LoRaWAN architecture

LoRaWAN operates in the Industrial, Scientific, and Medical (ISM) free band, with duty cycle limitations, and possibly additional limitations from network providers. Optimizing the parameters of wireless LoRaWan devices under those constraints is crucial for their proper functioning as well as for their operational longevity, but can be a complex and tricky process. Indeed, the choice of a Spreading Factor (SF), notably, affects the battery depletion speed, as well as the transmission rate, which in turn has an impact on competing communications.

In this paper, we specifically focus on the SF selection, which is currently managed through the Adaptive Data Rate (ADR) [4] algorithm, which aims to improve the Packet Delivery Ratio (PDR). For static nodes and stable radio channel environments, the NS manages the ADR depending on the history of the uplink (UL) packets received. Based on the reception quality of a given number of packets originating from a LoRaWan device, ADR calculates a new SF, to maintain a satisfying reception level while limiting energy consumption. However, reception quality does not only depend on channel conditions but also on the competing devices, whose behavior may vary [5]. To address such an issue, solving a global optimization problem has been envisioned [6], [7], but the heuristics proposed can present instabilities with non-controllable convergence times. The inherent limitations of centralized approaches have led to an interest in *distributed*

strategies (i.e., node-centric). The standard itself proposes a terminal-based heuristic for the configuration of its parameters based on an incremental exploration of the different configurations [8], unfortunately without guarantees in terms of stability or even efficiency. A particularly interesting direction, that we also follow in this paper, is the use of *reinforcement learning* techniques to be applied by the nodes. While some methods like Q-learning can use significant resources [9], others like *multi-armed bandits* [10] require very limited computing power and memory–what we want for battery-powered nodes–and can be effective even in non-stationary conditions [11].

This paper further develops the idea of applying bandit-like algorithms to LoRaWAN nodes, by relaxing several key assumptions generally made in the literature in order to get closer to real-life implementable solutions. Specifically,

(i) we do not assume nodes get immediate feedback after sending a frame (which would mean a downlink message for each uplink one, something saturating the scarce downlink radio spectrum). Instead, a node has to specifically ask for batched (aggregated) feedback for a number of sent uplink frames;

(ii) we do not assume each node's uplink frames reach only one gateway, but design a reinforcement learning approach compatible with the multi-gateway case;

(iii) we apply lightweight reinforcement learning techniques that are compatible with the "no immediate feedback" situation, namely delayed-feedback bandits, in particular, some reflecting the node energy consumption;

(iv) we design new LoRaWAN MAC commands for those techniques to be implemented in real LoRaWAN networks, and implement our proposal on a realistic (ns-3-based) LoRaWAN network simulator;

(v) we consider not only communication performance (Packet Delivery Ratio), but also energy consumption; in that sense this paper can be associated with the recent trend in research work focusing on power allocation and energy consumption for IoT (and especially LoRaWAN [12], [13]), in contrast to the (still mainstream) focus on PDR [14]. More specifically, we discuss performance metrics for LoRaWAN, in particular those reflecting the energy/success rate trade-off, that we use to compare our proposals to ADR.

The remainder of this paper is structured as follows. Section II introduces the main background concepts and related work. Our proposal is presented in Section III, while Section IV describes our simulator and provides performance results. Finally, Section V discusses the paper's contributions and suggests directions for future work.

II. State of the art: Multi-Armed Bandits and applications in IoT

This section first describes bandit problems and their delayed-feedback versions, then summarizes the existing literature on the application of such methods to IoT networks.

A. Background: The Basic Bandit problem

Multi-Armed Bandit (MAB) 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), as the situation is similar to that of a gambler having to maximize their gain by choosing among several machines. A specificity of bandit problems is that the decision maker 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–which selects the arm with the highest upper bound of the confidence interval estimating the mean–and Thompson Sampling (TS) [15]–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*, which compares the cumulative reward from an algorithm to the cumulative reward given by the best arm. The goal is generally to keep that regret as low as possible over time; Thompson Sampling is optimal in that sense [16], [17], and UCB also offers near-optimal performance [18].

B. 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 need to be adapted, and their performance is affected. That type of problem is particularly adapted for our LoRaWAN setting, in which an objective is to minimize the number of downlink transmissions, and thus it is not reasonable to assume that a node obtains some feedback (acknowledgment) after each frame emission.

The seminal work of Chapelle and Li [16] empirically validates TS in several bandit settings, including delayed feedback. The authors conclude that TS is very effective for handling the exploration-exploitation trade-off in all settings and is easy to implement (not requiring any parameter setting in its simplest form).

For delayed-feedback bandits, P. Joulani et al. [19], [20] study the case of non-constant and unbounded (but finite) delay, and propose a black-box approach to solve the delayed MAB problem using unmodified non-delayed MAB algorithms, through a meta-algorithm. The derived upper-bound guarantees for the stochastic and adversarial (a more general case than stochastic) settings suggest the price of delay is a multiplicative regret increase for adversarial problems and an additive increase in stochastic problems. Also, they provide UCB-based algorithms (non-back-box) for the MAB problem with delayed feedback, with similar performance (from

simulations). Desautels et al. [21] study how to improve the exploration-exploitation trade-off in the delayed reward setting; they propose two UCB-based algorithms that can select batches of experiments (i.e., arms selections) and can be run in parallel (i.e., independently of getting the results/rewards of the other batch).

Mandel et al. [22] build upon the black-box approach of [19], aiming to improve the exploration part through a queuing-based heuristic that adds (synthetic) exploration to the black-box algorithm, without having to pull/interact with the real environment. They show that some heuristics can improve performance while maintaining regret bounds close to the best-known. Using TS as the black box and choosing a heuristic algorithm based on its empirical performance [16] and its theoretical proof of optimal regret bounds [17] yields good results, while UCB-based variants perform worse, particularly for the cases with few rewards/samples available.

Vernade et al. [23] further explore the stochastic delayed bandit setting, using queuing and UCB-based approaches for the case where some feedback delay can be "censored" (e.g., not observable, infinite, lost).

For our problem, we will limit ourselves to implementing the original TS algorithm, with the only difference being that some feedback is delayed. Unlike some deterministic algorithms like UCB, the randomness inherent to TS still allows pulling different arms (exploration) even without receiving new feedback information, while deterministic approaches would result in always the same arm being pulled between two batches of feedback.

C. Bandits for wireless IoT networks

The literature contains several propositions for applying reinforcement learning in wireless communications, especially bandit algorithms because of their being lightweight and having performance guarantees.

In 2014, Avner and Mannor [24] consider a cognitive radio setting, where nodes have to choose a channel (an arm) to send data while avoiding collisions; they suggest a greedy bandit learning method for which performance guarantees are derived. Key assumptions include the reward distribution over arms being the same for all nodes, which is unrealistic for IoT LoRa networks, where nodes can experience very different radio conditions. For a similar setting, Rosenski, Shamir, and Szlak [25] improve the performance guarantee bounds, but with the additional assumption that the number of nodes is smaller than the number of arms, another constraint we cannot satisfy in IoT networks.

Closer to the present work, the authors of [26] implement several classical bandit methods for nodes to select their SF (and possibly emission power), in a LoRa simulator. However, that contribution focuses on the single-GW scenario, with only one node implementing learning while our approach is compatible with the multi-GW setting, and we have all nodes apply learning methods. More importantly, the learning algorithms compared in [26] assume immediate feedback after each uplink emission and ignore the half-duplex constraint of gateways.

Having several nodes learn simultaneously using bandits in IoT is the focus of [27]–[29], all authored by the same team. In [27], some nodes learn which channel to use while others maintain a constant choice, creating collisions. Note that the simulations consider perfect orthogonality among channels, and collisions are the only cause of packet loss. In this paper, we consider nodes spread over a geographic region (i.e., with different radio conditions), non-perfect orthogonality among arms (here, SFs, as quantified experimentally in [30]), and several possible reasons for packet loss. But we follow a similar approach, in applying stochastic bandit solutions to situations that do not perfectly fit the mathematical model they were designed for. More recently, the same team implements bandit algorithms in a real LoRaWAN network [28], [29], focusing on frequency selection, and not on SF as we do here. The main drawback of that approach is that each uplink frame requests an immediate downlink acknowledgment, causing a significant communication overhead that may be costly in realistic settings, and even possibly forbidden due to duty cycle constraints of the GW, and the half-duplex limitation.

In contrast with those contributions, which exhibited promising results under strong assumptions, in this paper, we aim at getting closer to realistic settings, by addressing multi-gateway scenarios, with a large number of nodes having node-specific radio conditions and learning based on delayed feedback information (imposed to limit the communication overhead). The solution we propose conforms with the LoRaWAN specifications and relies on the definition of two new LoRaWAN MAC messages; we implement it on the state-of-the-art simulator for LoRa networks, namely the LoRaWAN ns-3 module [31].

III. Proposal: Delayed-feedback Bandits on LoRaWAN

In this section, we describe the technical solution we propose for nodes to select their SF. The following subsection provides an overview of our proposal, while the two next subsections detail the *feedback request* mechanism, and finally, Subsection III-D explains and motivates the *bandit reward* function we choose to apply.

A. Overview

Our proposal consists of a Bandit Agent at the node, learning about k different LoRa PHY configurations (*arms*).

In this paper, we instantiate a Thompson Sampling (TS) Bandit without black-box adaptation, and we define six PHY configurations ($k = 6$ *arms*) that correspond to LoRa's SFs from 12 to 7. The TX power is fixed to 14dBm, and the PHY central frequency is chosen randomly and uniformly in $\{$868.1 MHz, 868.3 MHz, 868.5 MHz$\}$, i.e., not subject to learning.

In our proposal, the *feedback* or *reward* needs to be actively requested by the Bandit Agent (hence batched and delayed feedbacks) and involves a two-message protocol between the node and the Network Server (NS). That feedback is then

appended to the node's feedback information in order to select the SFs of the next emissions.

B. Feedback request: triggering strategy

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

1) An initial phase of b application messages with no feedback request. In our simulations, we take $b = 15$.
2) A long-term strategy to request feedback. A possibility that minimizes the risk of collisions, which we implement in our simulations, is for each application message to trigger a feedback request with probability $p = 1/20$ ($\sim$Bernoulli).

Fig. 2 illustrates the proposal from a single node's point of view, with an emphasis on the feedback request strategy.

The choice of b ***and*** p***:*** Both parameters can be adapted according to the particularities of the use case. Here, we describe design recommendations and lessons learned.

About b***:*** The parameter b is proportional to the number of bandit's arms (k) and the average number of trials/samples ($\hat{n}$) per arm we want to start the long-term strategy with (i.e., when a feedback request is asked, each arm will have–on average–$\hat{n}$ rewards' information); in our case $k = 6$, and we wanted *at least* two trials per arm. To approximate "*at least two*" we increased the average by 25% ($\hat{n} = 2.5$; $b = k.\hat{n} = 15$). This initial phase guarantees a uniform exploration of the arms, and this kind of exploration is useful if (we consider that) the current system state is not representative of the long-term state (i.e., we avoid potential early–unrepresentative–bias). In our simulations, the early system state is not stable—as all the nodes on the network are in the early learning phase. Empirically, we tried $b = \{10, 15, 20\}$ with no substantial differences. Otherwise, if we deploy a learning agent on a "stable" system, we could set $b = 0$.

About p***:*** The parameter p depends on the number of nodes, the uplink traffic periodicity, and the regional ISM bands spectrum restrictions (e.g., in EU the LoRaWAN GWs downlink duty-cycle is very limited–1%-10%– and is shared among all the nodes). For example, in The Things Network's (TTN's) public community network a Fair Use Policy applies which limits the downlink messages to 10 messages per day per node. For a private network, the allowed downlink messages per day will depend on the total number of nodes attached to a GW and the time-on-air used for the downlinks. Moreover, current Semtech-based **GWs are half-duplex** (i.e., RX is not possible while TX) and, while RX supports parallel demodulation of all SFs, TX/downlink supports only one SF at a time. **Downlink traffic greatly impacts the overall network performance and should be minimized**. Thus, we adhere to TTN's fair use policy and suggest setting p to request *at most* 10 downlink messages per 24 hours. In our simulations, we have an uplink packet frequency of 3 per hour (72 messages per day per node) and with $p = 1/20$ we have on average 3.6 ($= 72 \cdot 1/20$) downlink messages per day per node. In our simulations, aside from the chosen $\frac{1}{20}$ we tried $p = \{\frac{1}{15}, \frac{1}{10}\}$. While in the Single-GW scenario both yield improvements of $\approx +3\%$ on the main metric for a given optimization goal, in the Multi-GW scenario the results were worse; thus, we set $p = 1/20$ as a conservative choice. In future work, it would be interesting to use variable values for p (e.g., higher values in the early learning phase, and progressively decreasing its value), i.e., adaptive strategies.

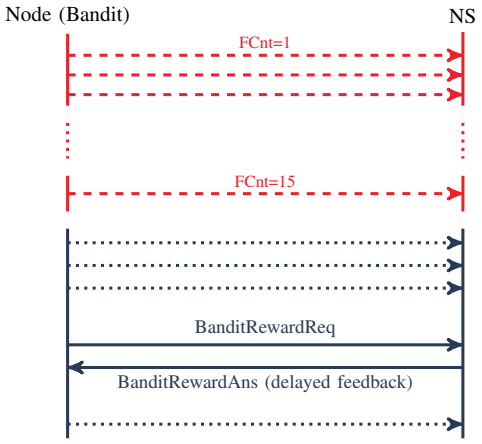

Fig. 2. Overview of our proposal, with emphasis on feedback requests. At the top, the initial phase of $b = 15$ messages without feedback requests. Then, the long-term strategy, in which every uplink message can trigger a feedback request with a probability $p = 1/20$. In solid lines, an example of a successful feedback request and response.

For **node-mobility scenarios**, past learned experience should be neglected and new one prioritized. For example, if mobility is detected, p should be increased. The previous learning experience can be discarded (as is no longer valid) and the learning agent may transition to an initial learning phase of b messages—where uniformly random arm-pull is prioritized. **We designed and instantiated our solution to achieve reasonable results for 6-armed Bandits in the order of 100 uplink messages**. A lower number of arms will yield a faster convergence, so in high-mobility scenarios one may use, for example, 3-armed Bandits (e.g., learning on SF $\in \{8, 10, 12\}$) and increase p. However, if the node's movement is *too fast*[1], for example, if every 10 uplink messages the node drastically changed position/environment, MAB (rather, reinforcement learning in general) may not provide a viable solution.

C. Feedback request: command format

To implement the delayed-feedback Bandit on a LoRaWAN network, we define two additional MAC commands, that comply with the LoRaWAN L2 1.0.4 Specification (note that both commands are smaller than 15 Bytes and can be piggybacked with application data):

• `BanditRewardReq` (UL): transmitted by a node to request reward statistics (Size: 4 Bytes). That command, whose syntax is given in Table I, triggers feedback/statistics for messages

[1]*fast* is relative to (a) the stability of the environment and (b) the convergence rate achievable for a given MAB.

with a frame counter (FCnt) number between [`Max FCnt − Delta` , `Max FCnt`], hence regarding `Delta`+1 frames.

TABLE I
MAC COMMAND: BANDITREWARDREQ (4 BYTES)

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

• `BanditRewardAns` (DL): transmitted by a Network Server to send reward statistics (Size: 7 Bytes). Table II details the command's syntax: in response to a `BanditRewardReq` (i.e., respecting the `FCnt` range), the command will answer with the number of packets received by the NS discriminated per SF $\in \{7, 8, 9, 10, 11, 12\}$—which corresponds to 6 arms of the Bandit.

TABLE II
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

As an illustrative example, consider the following exchange of MAC commands:

- ED→NS: `[0xBB, 0x08 0x00, 0x03]`
- NS→ED: `[0xBB, 0x00, 0x00, 0x00, 0x01, 0x00, 0x02]]`

The first message (1) is a `BanditRewardReq` with (`Max FCnt`[2] `= 8`, `Delta = 3`) and is requesting feedback statistics for Frame #5 to Frame #8. The message (2) is the `BanditRewardAns` with (`#Pkt_RCV SF9 = 1`, `#Pkt_RCV SF7 = 2`, `0` for other fields) answering that the NS received 1 packet in SF9, 2 packets in SF7, and 0 in other SFs. Upon receiving that `BanditRewardAns` message, the node can deduce that one among Frames #5 to #8 was lost. Note that only the node (not the GW) can know what SF was used for that lost frame.

D. Bandit rewards: Energy-aware approach

The *reward* definition synthesizes the optimization objective of each node. Our *generic* reward definition, given in (1) below, is defined per packet, since *a pull of the bandit's arm* $\equiv$ *a LoRa-PHY packet sent*. For a pull of arm $i \in \{1, ..., 6\}$, i.e., a packet sent with SF $13 - i$, we consider a reward

$$\text{reward} = \begin{cases} r_i & \text{if the packet is received (\textit{See Table III})} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

In this paper, We explore two reward definitions, leading to two SF selection strategies named as follows:

[2]The over-the-air byte order for all multi-byte fields is Little Endian.

• **PDR** focuses on packet delivery ratio (PDR) optimization, hence $r_i = 1$ for each arm i.
• **Energy-PDR** is based on PDR with energy-efficiency considerations, more specifically each arm reward is inversely proportional to the energy used by the LoRa-PHY layer, hence $r_i = 2^{i-1}$, since the time-on-air (and thus, the energy) approximately doubles when increasing the SF. Note that over several emissions with the same SF and empirical PDR p, the average reward is $p\ 2^{i-1}$, which is proportional to the amount of information successfully received per unit of energy spent on emissions.

In Table III, we summarize these reward definitions.

TABLE III
BANDITS: PACKET RECEIVED REWARD r_i DEFINITION

Optimization Objective	Packet Received Reward r_i per arm i					
	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

The delayed feedback message is then used to compute the rewards of the node (e.g., PDR for a mains-powered node, or Energy-PDR for a battery-powered one). Concretely, the *reward* calculation and input are done as follows:

1) The `BanditRewardAns` feedback message contains the number of packets RX_i received per arm i.
2) The ED keeps track of the number of packets sent per arm, and can then calculate the number of packets $\overline{\mathrm{RX}}_i$ lost per arm i.
3) Then, for every *arm* $i \in \{1, ..., 6\}$ and each sent packet, its associated *reward*–as defined in Eq. (1)–is computed.

Those *rewards* are fed to an unmodified Thompson Sampling (TS) Bandit. Note that for each SF, the order in which sent packets are lost is not contained in the `BanditRewardAns` message, but that information is not used by TS.

E. Performance metrics for LoRaWAN networks

In this subsection, we present performance metrics used in LoRaWAN networks to evaluate ADR proposals. We also suggest a new metric that will be relevant for our setting.

The 2020 survey article by Kufanesu et al. [14] identifies 22 ADR optimization proposals, we summarize the metrics used by those proposals in Table IV.

The Data Extraction Rate (DER) metric [2], is defined as the ratio of received messages to transmitted messages over a period of time and is equivalent to the Packet Delivery Ratio (PDR) which is one of the most used metrics in the literature. Another metric of interest is the Network Energy Consumption (NEC), defined as the energy spent by the network to successfully extract a message, however, it is not clear whether it accounts for energy spent on lost packets. Metrics focusing on errors/losses like the Packet Error Rate (PER) [32], [33] or the Packet Loss Rate (PLR) [34] investigated the correlation between the number of end nodes and PLR in the LoRa network using one gateway.

TABLE IV
PERFORMANCE METRICS USED ON LORAWAN'S ADR
(OUR SYNTHESIS FROM REFERENCES ON [14])

Metric Name	Definition	NS wide	Per SF	Per Frq	Per GW	Per Node
Data Extraction Rate (DER) [2]	The ratio of received messages to transmitted messages over a period of time	✓	✗‡	✗‡	✗‡	✓
Network Energy Consumption (NEC) [2]	Energy spent by the network to successfully extract a message, but does not count energy spent on lost packets	✓	✓	✓	✗	✗
Ubiquitous NEC (proposed here)	Energy spent by the network to successfully extract a message, counts energy spent on lost packets†	✓	✓	✓	✗	✓
Packet Delivery Ratio (PDR)	*—equivalent to DER metric—*					
Packet Error Ratio (PER)	$\frac{\#packets\_crc\_error}{\#sent\_packets}$	✓	✓	✓	✓	✓
Packet Loss Ratio (PLR)	$\frac{\#lost\_packets}{\#sent\_packets}$ (See§)	✓	✗‡	✗‡	✗‡	✓
Jain's Fairness index [6]	$\frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}$ (See¶)	✓	✓	✓	✓	✗

‡ Not applicable in a real deployment, because a lost packet can not be –easily– attributed to a 'PHY-link'. But applicable in a simulated environment.
† Not easily applicable in a real deployment, but applicable in a simulated environment.
§ In the bibliography, PLR is used as 1-PDR: 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 0 and 1, with 1 being perfectly fair.

Finally, we introduce a metric based on the Network Energy Consumption (NEC) [2], which we name "ubiquitous NEC" (uNEC). The uNEC metric measures the average energy used by the network to successfully transmit a message and counts the energy spent for lost packets. In other words, **the uNEC metric measures the average energy consumption of nodes per uplink packet, divided by the PDR**. In a real-world network, a LoRAWAN NS does not immediately have the PHY information of lost packets and, hence, the uNEC will not be easy to calculate. In that case, the NEC could be applied, but the reader must be aware that the NEC does not include a ponderation of the PDR of the network. The uNEC metric will be used in Section IV to compare the solutions from an energy point of view, a crucial aspect for battery-powered nodes.

IV. EVALUATION

In this section, we evaluate our proposal in the context of single- and multi-GW scenarios. We compare delayed-feedback bandits using PDR and Energy-PDR rewards, and the default LoRaWAN's ADR mechanism.

A. Implementation

The source code for our implementation of `LoRaWAN Bandits` is at https://github.com/renzoe/LoRaWAN-Bandits [35]. Our proposal is based on the `ns-3`[3] discrete-event network simulator and the `LoRaWAN ns-3 module`[4] [31]. For the Reinforcement Learning aspects, we use the `AIToolbox`[5] C++ library [36]. Most of our code is in the folder `/ns-3/src/lorawan/model/bandits`, but we also modified the base `lorawan/model`. Notably, we extended `Class A` nodes to add the delayed-feedback Bandit functionality, defined the delayed feedback MAC commands described before, and extended the NS capabilities to keep track of the required statistics. We also added `.pcap` tracing capabilities.

[3]https://www.nsnam.org/
[4]https://github.com/signetlabdei/lorawan
[5]https://github.com/Svalorzen/AI-Toolbox

B. Single-Gateway and Multi-Gateway scenarios

Our simulations consider both a single-gateway (single-GW) and a multi-gateway (multi-GW) case, described below.

Single-GW. A unique LoRaWAN's GW is located at the center of a 6400 meters-radius disc, in which 1000 nodes are uniformly distributed.

Multi-GW. This setup tries to reproduce a city-like scenario. Seven GWs are present, each one at the center of a hexagonal "cell", and separated by 2000 meters from contiguous GWs. 2000 EDs are uniformly distributed within a square area of 4400 meters per 4400 meters with one GW at the center. For this scenario, we included buildings simulation from LoRaWAN's module (around 1400 buildings) which adds attenuation.

C. Set up

The simulation parameters are given in Table V; each scenario is run three times, each time with a different ADR strategy implemented by nodes: (a) Legacy ADR, (b) Bandit PDR reward, (c) Bandit Energy-PDR reward.

TABLE V
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. [30]
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

Note that we use the Interference Matrix from Croce et al. [30] to simulate transmission failures, instead of the LoRaWAN module's default one which tends to underestimate inter-SF interference. Also, the application payload is 32B,

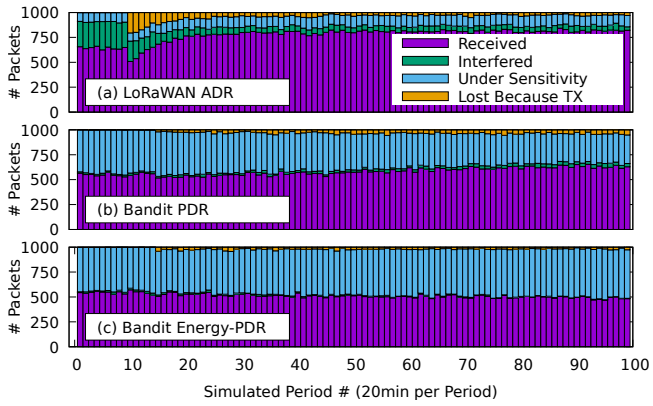

Fig. 3. Single-GW scenario packet performance results for (a) LoRaWAN ADR, (b) Bandit w/PDR rewards, and (c) Bandit w/Energy-PDR rewards, with the reasons for failure.

TABLE VI
RESULTS SINGLE-GW SCENARIO:
METRICS FOR THE LAST 10 PERIODS (1000 PACKETS PER PERIOD)

ADR Strategy	Metrics (See Table IV)					
	Total Energy [J]		PDR		Ubiquitous NEC [mJ]	
	mean	σ	mean	σ	mean	σ
LoRaWAN ADR	28.18	0.06	**81.8%**	0.7%	34.47	0.34
Bandit PDR	21.95	0.63	63.2%	1.4%	34.76	1.20
Bandit Energy-PDR	**10.06**	0.21	48.9%	1.3%	**20.63**	0.66

which allows for the delayed feedback MAC commands to be piggybacked.

Bandit Bootstrapping. The `AIToolbox`'s TS implementation needs at least two samples/rewards per arm (to have a *mean* and a *variance*). We manually set two rewards of `0` and `1`, inducing a mean reward of `0.5`, to all arms. Thus, in the node's bootstrapping phase–before the first feedback request (See Sec. III-B)–arms are chosen following a uniform distribution. We tried other bandit bootstrapping strategies (e.g., to prioritize low-energy arms), but the chosen strategy worked consistently well in diverse scenarios.

D. Single-GW Results

Fig. 3 shows the system packet statistics over time, i.e., whether frames are received or not, with the reason for no-reception:

- "Interfered": inter- and intra-SF interference, with probabilities taken from [30]);
- "Under sensitivity": received power below LoRa PHY sensitivity threshold;
- "Lost because TX": UL frame overlapping with DL transmission).

Fig. 4 represents the spatial distribution of the nodes and their SF for periods #11, #50, and #100. Finally, Table VI focuses on three system-wide (NS-wide) metrics, giving their mean and standard deviation over the last 10 simulated periods (i.e., #91 to #100).

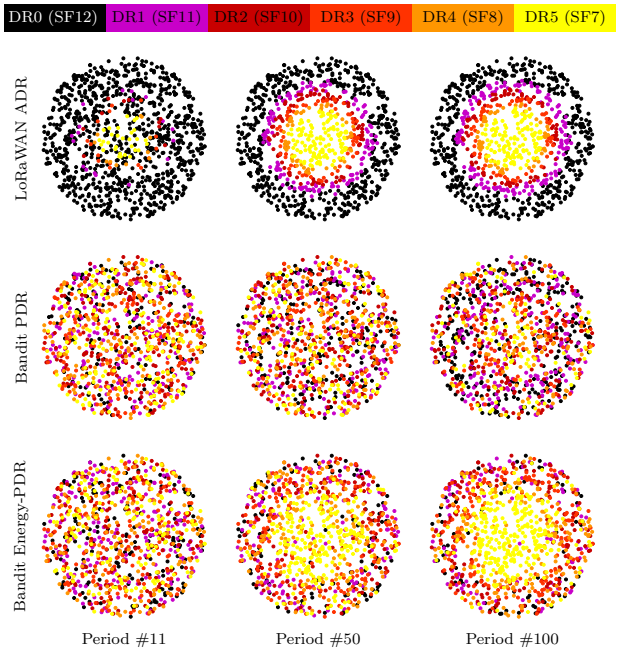

Fig. 4. Single-GW scenario, the spatial distribution of nodes, and Spreading Factor at periods #11, #50, and #100 for LoRaWAN ADR, Bandit with PDR rewards, and Bandit with Energy-PDR rewards.

1) PDR-centric Discussion: (a) LoRaWAN ADR achieves the highest system's PDR value of 81.8%, a stable value as indicated by its low standard deviation. Fig. 3 shows that the system "converges" (within 1.5%) to that PDR value around period #35 (i.e., fast). LoRaWAN ADR's periods #1 to #10 are marked with high interference due to all ED's using SF12, which has the largest time-on-air; in contrast periods #11 to around #35 are marked with heavy downlink traffic ("Lost because TX") as the NS is aggressively/actively trying to set the appropriate SF for each ED; in a real-world network–where not every node bootstraps at the same time–this will not be an issue.

(b) The Bandit PDR strategy, which aims at optimizing PDR, achieves a mean value of 63.2% for the last 10 periods. However, the value has not converged: the standard deviation over the last 10 periods is 0.014, and Fig. 3 shows a slight growing trend that is not over. We corroborated this trend with an ad-hoc simulation of 1000 periods, for which the PDR kept increasing (at a seemingly logarithmic growth), reaching a value of 78.4%. Note also that there are fewer "Lost Because TX" and "Interfered" packets than with LoRaWAN ADR.

(c) The Bandit Energy-PDR strategy reaches a value of 48.9%, with a decreasing trend. As seen in Fig. 4, the SF distribution resembles that of LoRaWAN ADR but with more weight towards the lower SFs. The energy-centric discussion will complement and help to understand this behavior; but is not surprising, as this bandit algorithm does not maximize PDR, but instead the quantity of information successfully transmitted per unit of energy consumed, hence the predominance of lower SFs. To understand the long-term behavior of

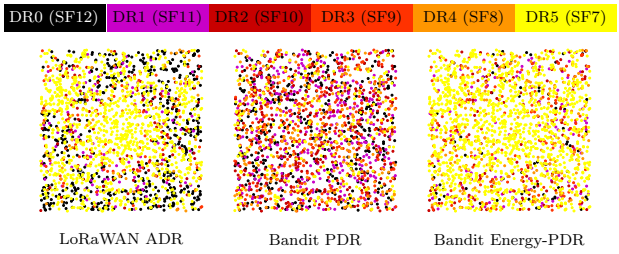

Fig. 5. Multi-GW scenario, the spatial distribution of ED's, and Spreading Factor at period #100 for LoRaWAN ADR, Bandit with PDR rewards, and Bandit with Energy-PDR rewards.

the PDR, we ran an ad-hoc simulation of 3000 periods: from an initial PDR value of 54.2% ($\sigma = 1.4\%$) the PDR kept slowly decreasing until around period #400 (PDR=47.9%), where the trend reverses and there is a slight upper trend; the average PDR is 56.3% ($\sigma = 1.2\%$) for periods #991-1000, and 58.5% ($\sigma = 0.9\%$) for periods #2991-3000. Our macroscopic analysis is that, in this setting, the Bandit Energy-PDR strategy has a first phase that aggressively converges to low SFs ($\approx$ period #400), and a second, stable, phase that starts to improve the PDR of each SF "region", by slowly increasing some SF choices at the frontiers of the concentric regions.

2) Energy-centric Discussion: We discuss here the energy-related metrics, to complement the pure-PDR analysis. Regarding total system energy consumption, the Bandit Energy-PDR has a value of 10.06Joules, which is half of the Bandit PDR at 21.95J, and almost one-third of LoRaWAN ADR at 28.18J. However, raw energy consumption is not that relevant, as it needs to take into consideration the PDR of the system. The ubiquitous NEC (uNEC) gives us this perspective; rephrasing its definition, uNEC is equivalent to the average energy spent per message divided by the average PDR, approximating the average amount of energy spent to successfully sent a packet. We want this value to be as low as possible (ideally, the lowest energy per message and PDR=1). As seen in Table VI, the uNEC values are for Bandit Energy-PDR 20.6mJ, while Bandit PDR and LoRAWAN ADR have similar values of 34.8mJ and 34.5mJ, respectively. **If good PDR performance pondered by energy consumption (i.e., uNEC) is a concern of the global IoT system, the Bandit Energy-PDR offers a significant advantage over the other strategies, with a 40% reduction of the average energy consumption per delivered packet**. In terms of total energy spent, the advantage is even greater ($\geq 50\%$), which can be relevant if the use case/IoT application does not need high PDR/reliability but instead crucially values battery life.

E. Multi-GW Results

Fig. 5 represents the spatial distribution of the EDs and their SF for period #100. In Table VII, we calculate three system-wide (NS-wide) metrics taking into account the last 10 simulated periods (i.e., #91 to #100).

1) PDR-centric Discussion: In contrast with the single-GW scenario where PDRs could differ by as much as 30 percentage points, here all strategies achieve closer values for the PDR, all above 70%. This can be because, even with attenuation from buildings, any node is at most 2 km away from the closest GW. On top of that, we focused on a square area around a central GW and do not include "peripheral" areas. Otherwise stated, we are simulating and evaluating the downtown area of a LoRaWAN-well-covered city, while the single-GW scenario is more representative of a rural region.

TABLE VII
RESULTS MULTI-GW SCENARIO:
METRICS FOR THE LAST 10 PERIODS (2000 PACKETS PER PERIOD)

SF selection strategy	Metrics (See Table IV)					
	Total Energy [J]		**PDR**		**Ubiquitous NEC [mJ]**	
	mean	σ	mean	σ	mean	σ
LoRaWAN ADR	30.03	0.09	**80.7%**	0.5%	18.61	0.14
Bandit PDR	35.34	0.75	75.3%	0.4%	23.46	0.42
Bandit Energy-PDR	**12.77**	0.36	70.2%	0.5%	**9.09**	0.25

2) Energy-centric Discussion: As seen in Table VII, the Bandit Energy-PDR achieves the best results for energy-aware metrics. The Bandit PDR is the worst performer in terms of Total Energy and uNEC values, with the LoRaWAN ADR having slightly lower values. Concretely, the Bandit Energy-PDR achieves a uNEC of 9.1mJ, while LoRaWAN ADR and Bandit PDR at least double this value with 18.6mJ and 23.5mJ, respectively.

V. CONCLUSIONS

This paper focuses on proving the usability of bandits on LoRAWAN networks in a realistic setting. For that, we relaxed several (unrealistic) assumptions made in the prior literature, regarding the management of several gateways and the limited number of downlink messages (that prevents having immediate feedback). We propose specific LoRaWAN MAC commands to obtain the necessary feedback to use reinforcement learning and implement our proposals on a realistic simulator (based on ns-3); with nodes adjusting their Spreading Factor either to maximize their Packet Delivery Ratio or to optimize their energy use (through a "transmitted frames per Joule" measure). **We also share the source code of our implementation** [35].

Our results show that the simple and lightweight bandit method used (Thompson Sampling with delayed feedback) offers good performance, in particular in terms of energy-related metrics, which are to be favored when dealing with battery-powered nodes. In particular, the energy consumption (number of Joules per successfully transmitted frame) is improved by at least 40% with respect to the currently implemented LoRAWAN's ADR algorithm.

The paper also opens several directions for future work. At first, one would be to **consider not only the Spreading Factor as a transmission decision** made by nodes, but also the channel (i.e., frequency) to use, the transmission power, and the coding rate. However, the number of configurations subject to reinforcement learning (i.e., bandit's arms) should be kept low to avoid a "slow" convergence of the learning

mechanisms, penalizing the real-world usability of the solution. Another direction of interest could be to investigate **nodes having different decision mechanisms**, like a proportion of nodes implementing legacy ADR while the others use (possibly different) reinforcement learning methods. One can also imagine nodes **changing objectives–hence the bandit algorithm–over time** (e.g., when the remaining battery energy gets below some threshold) but still benefit from the learning experience. Finally, our paper only considers static nodes, a situation with mobile nodes would impose **adding time considerations when making decisions**, as learned information will be only ephemeral.

ACKNOWLEDGMENTS

We would like to sincerely thank Alessandro Aimi for his help with the `.pcap` tracing capabilities. This work was funded by the French national research agency ANR, through the INTELLIGENTSIA project (grant: ANR-20-CE25-0011).

REFERENCES

[1] LoRa Alliance, "LoRaWAN what is it? A technical overview of LoRa and LoRaWAN LoRa alliance," 2015.

[2] M. C. Bor, U. Roedig, T. Voigt, and J. 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*. ACM, 2016, pp. 59–67.

[3] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the limits of LoRaWAN," *IEEE Communications magazine*, vol. 55, no. 9, pp. 34–40, 2017.

[4] C. Lehong, B. Isong, F. Lugayizi, and A. 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)*, 2020, pp. 1–9.

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

[6] S. Dawaliby, A. Bradai, and Y. Pousset, "Adaptive dynamic network slicing in LoRa networks," *Future Generation Computer Systems*, vol. 98, pp. 697–707, Sep. 2019.

[7] F. Z. Mardi, M. Bagaa, Y. Hadjadj-Aoul, and N. Benamar, "An efficient allocation system for centralized network slicing in lorawan," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*, 2022, pp. 806–811.

[8] Semtech, "Understanding adr: Developer portal." [Online]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/understanding-adr/

[9] R. Carvalho, F. Al-Tam, and N. Correia, "Q-learning adr agent for lorawan optimization," in *2021 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, 2021, pp. 104–108.

[10] S. A. Almarzoqi, A. Yahya, Z. Matar, and I. Gomaa, "Re-learning exp3 multi-armed bandit algorithm for enhancing the massive iot-lorawan network performance," *Sensors*, vol. 22, no. 4, 2022.

[11] R. Bonnefoi, L. Besson, C. Moy, E. Kaufmann, and J. Palicot, "Multi-armed bandit learning in iot networks: Learning helps even in non-stationary settings," in *Cognitive Radio Oriented Wireless Networks*. Cham: Springer International Publishing, 2018, pp. 173–185.

[12] Y. A. Al-Gumaei, N. Aslam, X. Chen, M. Raza, and R. I. Ansari, "Optimizing power allocation in lorawan iot applications," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3429–3442, 2021.

[13] R. Carvalho, F. Al-Tam, and N. Correia, "Q-learning adr agent for lorawan optimization," in *2021 IEEE international conference on industry 4.0, artificial intelligence, and communications technology (IAICT)*. IEEE, 2021, pp. 104–108.

[14] R. Kufakunesu, G. P. Hancke, and A. M. Abu-Mahfouz, "A survey on adaptive data rate optimization in lorawan: Recent solutions and major challenges," *Sensors*, vol. 20, no. 18, p. 5044, 2020.

[15] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[16] O. Chapelle and L. Li, "An empirical evaluation of Thompson sampling," *Advances in neural information processing systems*, vol. 24, pp. 2249–2257, 2011.

[17] S. Agrawal and N. Goyal, "Analysis of Thompson sampling for the multi-armed bandit problem," in *Conference on learning theory*. Journal of Machine Learning Research, 2012.

[18] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, pp. 235–256, 2002.

[19] P. Joulani, "Multi-armed bandit problems under delayed feedback," Master's thesis, Dep. of Computing Science, University of Alberta, 2012.

[20] P. Joulani, A. Gyorgy, and C. Szepesvári, "Online learning under delayed feedback," in *International Conference on Machine Learning*. PMLR, 2013, pp. 1453–1461.

[21] T. Desautels, A. Krause, and J. W. Burdick, "Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization," *Journal of Machine Learning Research*, vol. 15, pp. 3873–3923, 2014.

[22] T. Mandel, Y.-E. Liu, E. Brunskill, and Z. Popović, "The queue method: Handling delay, heuristics, prior data, and evaluation in bandits," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[23] C. Vernade, O. Cappé, and V. Perchet, "Stochastic Bandit Models for Delayed Conversions," in *Conference on Uncertainty in Artificial Intelligence*, Sydney, Australia, Aug. 2017.

[24] O. Avner and S. Mannor, "Concurrent bandits and cognitive radio networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 66–81.

[25] J. Rosenski, O. Shamir, and L. Szlak, "Multi-player bandits–a musical chairs approach," in *International Conference on Machine Learning*. PMLR, 2016, pp. 155–163.

[26] R. Kerkouche, R. Alami, R. Féraud, N. Varsier, and P. Maillé, "Node-based optimization of LoRa transmissions with Multi-Armed Bandit algorithms," in *2018 25th International Conference on Telecommunications (ICT)*. IEEE, 2018, pp. 521–526.

[27] R. Bonnefoi, L. Besson, C. Moy, E. Kaufmann, and J. Palicot, "Multi-armed bandit learning in IoT networks: Learning helps even in non-stationary settings," in *International Conference on Cognitive Radio Oriented Wireless Networks*. Springer, 2017, pp. 173–185.

[28] L. Besson, "Multi-players bandit algorithms for internet of things networks," Ph.D. dissertation, CentraleSupélec, 2019.

[29] C. Moy, L. Besson, G. Delbarre, and L. Toutain, "Decentralized spectrum learning for radio collision mitigation in ultra-dense IoT networks: LoRaWAN case study and experiments," *Annals of Telecommunications*, vol. 75, no. 11, pp. 711–727, 2020.

[30] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, and I. Tinnirello, "Impact of LoRa imperfect orthogonality: Analysis of link-level performance," *IEEE Communications Letters*, vol. 22, no. 4, pp. 796–799, 2018.

[31] D. Magrin, M. Capuzzo, and A. Zanella, "A thorough study of LoRaWAN performance under different parameter settings," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 116–127, 2019.

[32] B. Reynders, Q. Wang, P. Tuset-Peiro, X. Vilajosana, and S. Pollin, "Improving reliability and scalability of lorawans through lightweight scheduling," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1830–1842, 2018.

[33] V. Hauser and T. Hégr, "Proposal of adaptive data rate algorithm for LoRaWAN-based infrastructure," in *2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*. IEEE, 2017, pp. 85–90.

[34] R. M. Sandoval, A.-J. Garcia-Sanchez, and J. Garcia-Haro, "Optimizing and updating LoRa communication parameters: A machine learning approach," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 884–895, 2019.

[35] R. E. Navas, "Energy-Aware Spreading Factor Selection in LoRaWAN Using Delayed-Feedback Bandits: Code and Data." Apr. 2023. [Online]. Available: https://doi.org/10.5281/zenodo.7876244

[36] E. Bargiacchi, D. M. Roijers, and A. Nowé, "AI-Toolbox: A C++ library for Reinforcement Learning and Planning (with Python Bindings)," *Journal of Machine Learning Research*, vol. 21, no. 102, pp. 1–12, 2020.