LoRaWAN Adaptive Data Rate With Flexible Link Margin

Gabriel Germino Martins de Jesus, Richard Demo Souza, *Senior Member, IEEE*, Carlos Montez, *Member, IEEE*, and Arliones Hoeller, Jr., *Member, IEEE*

***Abstract*—LoRaWAN is a remarkable network protocol to support the deployment of low-power wide-area network applications for the Internet of Things. Its modulation scheme, LoRa, uses different transmission parameters to allow long-range bidirectional communication by trading-off range for Time on Air (ToA). The LoRaWAN specification suggests using an adaptive data rate (ADR) algorithm to assign transmission parameters to nodes dynamically. The ADR algorithm used by The Things Network, a global open network, selects transmission parameters for communication without disconnection while lowering ToA and energy consumption. However, its often optimistic link quality estimate drives the assignment of inadequate parameters, especially in lossy channels. More precise channel quality estimates would improve resource allocation at the cost of increased complexity at the network server (NS). Through simulations, we observe that we can use the ADR link margin parameter to compensate for inaccurate link quality estimates, providing better service to nodes and maintaining low levels of energy consumption. In this work, we propose a modification to ADR that allows the selection of the link margin at runtime, increasing the network performance without the need of any prior knowledge and with no increase in the NS computational complexity.**

***Index Terms*—LoRaWAN, communications and networking for Internet of Things (IoT), low-power wide-area networks, machine-to-machine communications, sensor and actuator networks.**

I. Introduction

THE Internet of Things (IoT), in which electronics, machines, and everyday-objects sense their environment and share information with other devices and humans through the Internet, is boosted by the increasing number of connected devices. Such devices should surpass the number of smartphones and tablets already in 2020, creating, by 2024, an IoT industry responsible for a revenue of over $4 trillion [1].

Manuscript received July 20, 2020; revised September 23, 2020; accepted October 21, 2020. Date of publication October 27, 2020; date of current version March 24, 2021. This work was supported in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Print under Grant 698503P. *(Corresponding author: Richard Demo Souza.)*

Gabriel Germino Martins de Jesus and Richard Demo Souza are with the Department of Electrical and Electronics Engineering, Federal University of Santa Catarina, Florianópolis 88040-900, Brazil (e-mail: gabrielgmj3@gmail.com; richard.demo@ufsc.br).

Carlos Montez is with the Department of Automation and Systems, Federal University of Santa Catarina, Florianópolis 88040-900, Brazil (e-mail: montez@ieee.org).

Arliones Hoeller, Jr., is with the Department of Telecommunications, Federal Institute for Education, Science, and Technology of Santa Catarina, São José 88103-902, Brazil (e-mail: arliones.hoeller@ifsc.edu.br).

Digital Object Identifier 10.1109/JIOT.2020.3033797

One challenge in IoT is to support applications demanding a massive number of low-cost and long-life battery-powered devices communicating over long distances, in applications, such as smart metering and environmental monitoring [1]. While current technologies can reach long distances (e.g., cellular communication), operate with low-power consumption (e.g., Bluetooth low energy), and even support massive numbers of connected devices (e.g., massive MIMO), some applications demand all of these features in a single technology. The concept of low-power wide-area networks (LPWANs) guides the development of such technologies. A prominent LPWAN technology is LoRaWAN [2], an open network protocol. LoRaWAN has three main elements: 1) the nodes (or end devices, ED); 2) the gateways (GWs), which are agnostic to the content of received frames; and 3) the network server (NS), where lies most of the computational complexity.

LoRaWAN is based on LoRa [3], the Semtech's proprietary modulation technology that uses chirp spread spectrum, allowing the demodulation of messages with a low signal-to-noise ratio (SNR) at lower data rates. The data rates change by selecting one of the available bandwidths (BW) and spreading factors (SFs), the latter ranging from 7 to 12 (SF7-SF12). As the SF increases, the required SNR at the receiver decreases, improving sensitivity. Moreover, SF selection has a significant impact on time-on-air (ToA) of transmissions [1], impacting both energy consumption and the number of collisions [4].

The correct assignment of LoRa transmission parameters is vital to meet the LPWAN requirements of massive connectivity and long range. While assigning SF12 and maximum transmit power to all EDs would likely result in a successful connection even at great distances, this would increase collisions at the GW due to the excessive ToA [4]. Moreover, it would induce fewer transmissions by EDs because of duty-cycle restrictions imposed by regional regulations [5]. Besides that, also due to the excessive ToA, EDs would consume too much energy, thus reducing battery lifetime. For adequately allocating transmission parameters, the LoRaWAN protocol includes an adaptive data rate (ADR) mechanism to optimize transmission parameters and achieve long-range communication, low energy consumption, and scalability [2]. ADR runs asynchronously at the NS and at the EDs, behaving differently in each case. The ED-side ADR selects new transmission parameters if the ED seems to have lost connection to a GW. However, the LoRaWAN specification does not explicitly detail the NS-side of ADR, delegating the design of this part of the algorithm to the network operator.

Although ADR is not standardized, The Things Network (TTN) [6], a community-funded, world-wide, open LoRaWAN network, implements an updated version of the algorithm recommended by Semtech [7], which will be referred to, herein, as ADR-TTN. The algorithm assigns new transmission parameters based on channel estimates obtained from the SNR measure of the N most-recently received frames. ADR-TTN is capable of dynamically allocating reasonable transmission parameters, but several recent research works demonstrated that considerable improvements are possible.

A. Related Work

The literature contains several proposed algorithms to allocate resources in LoRaWAN, from small changes in the way ADR estimates the channel conditions to radically new approaches. Next, we review a few of these studies.

In order to distribute SFs among nodes, Cuomo *et al.* [8] exploited the received signal strength indicator (RSSI) of frames at the GW, assigning higher SF to nodes with lower RSSI values. To determine the percentage of nodes operating with each SF, they present the methods EXPLoRa-SF, where an equal number of nodes transmit with each SF, and EXPLoRa-AT, which allocates SF to equalize the activity levels in each SF, incurring in fewer nodes transmitting with higher SFs. Compared to ADR-TTN, EXPLoRa-SF supports more connected nodes at the expense of more frequent collisions, while the added complexity of EXPLoRa-AT pays off with fewer collisions and better performance.

Similar to EXPLoRa-AT, Abdelfadeel *et al.* [9] proposed FADR, which distributes SFs to equalize collision probability. Moreover, they propose a method for allocating transmit power that distributes LoRa discrete power values so that nodes have a similar probability of delivering their messages regardless of their distance to the GW. Simulations show that FADR outperforms EXPLoRa-SF in almost every scenario.

Park *et al.* [10] proposed EARN, which includes the dynamic selection of the available coding rates (CRs) in LoRa modulation. The method takes into account the probability of losing a packet by considering the link quality and SF occupancy, selecting a combination of SF, transmit power, and CR that should allow nodes to communicate with the server successfully. Additionally, a variable margin based on the standard deviation of SNR measurements can be added to the link quality estimation, compensating for the received power variation. When considering a low varying medium, EARN has similar performance to FADR in terms of data extraction rate (DER)—which is the ratio between successfully received frames and sent frames—and energy efficiency, with both outperforming ADR-TTN considerably in these metrics.

Kim and Yoo [11] proposed an algorithm to optimize throughput by exploiting the semiorthogonality of SFs to create new channels. Only SF7-SF9 are used to evaluate this method, as the authors assume that, in practice, nodes seldom use higher SFs in multigateway deployments. The proposed algorithm delivers higher throughput compared to schemes similar to EXPLoRa-SF and EXPLoRa-AT because it allocates most nodes to smaller SFs. While it increases throughput, it also decreases reliability due to disconnection. Besides that, the convergence time increases as more devices enter the network.

Marais *et al.* [12] evaluated Multitech's proprietary ADR algorithm in a testbed, showing the algorithm performance is worse compared to a deployment with manually assigned SF and transmit power. There is little information available about this algorithm, but it is different from ADR-TTN since almost all nodes used the fastest data rate, resulting in lower DER.

Finnegan *et al.* [13] assessed the performance of ADR-TTN using the NS-3 simulator and propose modifications to the algorithm. First, the NS estimates the DER and instructs nodes to lower their data rates if this estimate is below 80%. At the NS-side of the algorithm, after a data rate change, ADR-TTN commands can be anticipated if enough SNR measures indicate that they should. These proposed changes cause a significant improvement in both delivery ratio and convergence time compared to the standard ADR-TTN.

Li *et al.* [14] evaluated ADR-TTN focusing on convergence time, which for ADR-TTN differs at the NS and EDs, the former converging faster. Reducing the number of failed transmissions before stepping up the ED's SF results in faster convergence time, especially when the link quality is low. However, on the NS-side, changing the number of SNR measures needed to perform the ADR-TTN algorithm brings no significant improvement.

Slabicki *et al.* [15] introduced the OMNeT++-based simulator FLoRa and ADR+, a simple and efficient modification to the NS-side of the algorithm. Rather than taking the highest SNR measured from the most recently received N frames, ADR+ more conservative approach considers their average and outperforms ADR-TTN in both DER and energy efficiency.

Garg [16] performed an extensive experimental investigation of the ADR-TTN algorithm and proposes to decrease collisions by restricting the number of frequency channels a node can use, and to prevent disconnection between ED and GW by defining a threshold for the *Nstep* variable in ADR-TTN calculations. The threshold depends on the node experiencing or not spatial diversity by reaching multiple GWs. These changes improve DER significantly, especially because the NS becomes more conservative in resource allocation due to the threshold modification, as happens to ADR+. On the other hand, if the link quality is bad, the rather slow ED-side of the algorithm is almost solely responsible for improving performance, as around 99% of the time the network does not work actively to select stronger transmit power values.

B. Contribution

We use the FLoRa simulator [15] to analyze the impact of the overlooked ADR variable margin_*db*—an error margin used to compensate for ADR's optimistic channel condition estimate. The typical values of margin_*db* often result in low DER, as the algorithm considers that the majority of packet losses are due to interference and not disconnection caused by strong attenuation. However, we show that it is possible to

select values of margin_*db* that increase DER and energy efficiency at the same time. Furthermore, we propose ADRX, a new resource allocation algorithm for LORAWAN that selects the margins dynamically for each node to satisfy a reference DER value. We show that ADRX outperforms ADR-TTN and ADR+ in their default configurations, while also providing automatic margin_*db* selection to achieve target DERs without prior knowledge of the network deployment. For instance, a 200-node network where the average DER is below 40% with both ADR-TTN and ADR+, achieves 90% DER with the proposed ADRX algorithm.

The remainder of this article is as follows. Section II presents an overview of LORA and LORAWAN, and introduces the simulation model used in this work. Section III discusses the LORAWAN ADR and the ADR-TTN and ADR+ algorithms, and presents simulation results that motivate our work. We describe ADRX and present the comparative simulation results in Section IV. Finally, Section V summarizes the work and concludes this article.

II. LORA AND LORAWAN

LORAWAN is a network protocol developed by LORA Alliance for use with LORA at the physical layer. LORA characteristics are exploited by LORAWAN to achieve the LPWAN requirements [1]. In the following, we present these LORA aspects and a basic description of LORAWAN.

A. LORA

LORAWAN uses LORA modulation in the sub-GHz industrial, scientific, and medical (ISM) band, with BW of 125, 250, and 500 kHz. Error correction codes are available with rates CR varying from 4/5 to 4/8 [3]. LORA radios have discrete transmit power values (P_t), depending on the region [5]. In the EU868 band, P_t goes from 2 to 14 dBm, in 2-dB steps [13].

A key element of LORA modulation is the SF, which impacts the ToA of the transmitted signal. The larger the SF, the more robust the transmitted signal, which can be demodulated even below the noise floor [1]. LORA specifies six SFs [3], SF7–SF12. The SFs are semiorthogonal to each other, allowing the reception of multiple frames at the same time practically without collision, as long as they are transmitted using different SFs [4]. The SF choice has a significant impact on performance. The SNR required at the GW to successfully demodulate a frame is smaller in higher SFs, tolerating stronger signal attenuation and allowing longer ranges. On the other hand, the ToA increases with SF, resulting in a tradeoff between range and ToA, ultimately impacting energy consumption, data rate, and even packet delivery, since longer transmissions are more likely to collide [4].

LORA frames collide [4] when all of the following are true: the reception of a new frame starts while the last frame is still being received (time collision); the frames use the same SF (SF collision); and the frames occupy the same frequency channel (frequency collision). However, even if a collision happens, one of the frames may be decoded due to the capture effect, which makes it possible to decode the frame with a stronger signal (typically at least 6 dB above interference).

B. LORAWAN

The LORAWAN protocol [2] has a star topology such that each ED is connected to as many GWs as they can reach without the need for handovers. GWs simply forward the decoded frames to the NS through a standard Internet protocol (IP) connection. The NS removes replicated frames, schedules acknowledgment downlinks, and assigns, if requested by the ED, new transmission parameters for future uplinks. All data transmitted by EDs is encrypted and the payload contents are only visible at the application server.

Following the ALOHA [17] medium access control (MAC) protocol, nodes transmit every time they have data available unless restricted by duty-cycle. Nodes mostly work asynchronously to the NS and hop pseudorandomly over all available channels. LORAWAN defines three device classes [2] that target different application requirements. While all devices must support class *A* configuration, classes *B* and *C* are optional, and enable additional downlink opportunities, reducing downlink latency while increasing energy consumption.

C. LORAWAN *Simulation Model*

The simulations in this article use the FLORA simulation model for the OMNeT++ simulator [15]. We consider an application in which n class *A* EDs are randomly placed in a circular area with radius R. A LORAWAN GW is positioned at the center of the circular region. The GW is connected to the NS through a standard IP connection. All end devices comply to the same duty-cycle (DC), and start transmitting after a delay drawn from an exponential distribution with mean $1/\lambda_0$. In addition to the DC restriction, devices draw the time for their next transmission from exponential distributions with mean $1/\lambda_1$. All devices run the same application, the frame format being a p_a-symbols preamble and p_l-bytes payload. When no adaptive algorithm is used, the SF and P_t with which each node transmits are randomly selected from uniform distributions. However, when ADR is used, all nodes start with SF12 and $P_t = 14$ dBm. This way, the algorithms running on the NS and the EDs are responsible for the resource allocation.

The channel model includes path loss and i.i.d. Rayleigh fading h across time and space [17].[1] For each transmission, there is an instantaneous variation in the received power, P_r, such that $P_r = h^2\overline{P_r}$, where $\overline{P_r}$ is the average received power considering that particular position and transmit power. Moreover, $\overline{h^2} = 1$.

The average received power at a given distance d considering a particular P_t follows a standard log-distance path-loss model [17], i.e.,

$$\overline{P_r} = P_t - \overline{PL}(d_0) - 10\eta \log \frac{d}{d_0} \tag{1}$$

where $\overline{PL}(d_0)$ is the path loss for a reference distance d_0, and η is the path-loss exponent which determines how strong is the attenuation in the medium. In our model, an outage happens either if there is a collision at the GW (see Section II-A), or if

[1]The channel model does not consider shadowing because the results with shadowing are qualitatively the same once we analyze performance based on the average of several random deployments.

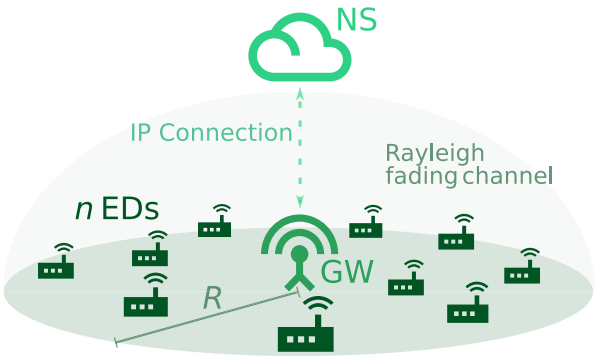

Fig. 1. Illustration of the system model with end devices, GW, and NS.

the ED is disconnected, i.e., P_r is below the radio sensitivity. Moreover, we consider an energy consumption model similar to [18]. Fig. 1 illustrates the physical aspects of the system model.

III. Adaptive Data Rate

To dynamically allocate SF and transmit power, LoRaWAN may use an ADR algorithm [2]—a distributed algorithm that runs at both the NS and the EDs. While the ED-side of the algorithm is specified in [2], the NS-side of ADR is not. Nevertheless, Semtech provides a reference algorithm, and ADR-TTN is an updated version of it.

At the ED-side, a counter monitors the device connectivity. For each new transmission, the counter is incremented, returning to 0 whenever a downlink is received. If the counter reaches a threshold (ADR_ACK_LIMIT), the following frames request a response from the NS. After a number of transmissions without response (ADR_ACK_DELAY), P_t is increased. If the node reaches the maximum P_t, the node increases the SF. This process is repeated every ADR_ACK_DELAY transmissions until a response is received or the node reaches the maximum SF and P_t. The values of ADR_ACK_LIMIT and ADR_ACK_DELAY are configurable and equal to 32 by default [2].

The way the NS evaluates the quality of the links is by SNR measurements performed at the GWs. The algorithm computes the number of P_t or SF adjustment steps required to reach an expected stable communication by taking into account the link quality, the sensitivity for each SF (SNR_{SF}) [19], and an error margin (margin_db). For each step, the NS instructs the ED to either decrease SF or P_t or to increase P_t. The channel estimation considers the maximum SNR value (SNR_m) from the last N transmissions. The number of steps N_{step} is

$$N_{\text{step}} = \lfloor \frac{\text{SNR}_m - \text{SNR}_{\text{SF}} - \text{margin\_}db}{3} \rfloor. \quad (2)$$

With this result, SF and P_t are modified following the flowchart presented in Fig. 2, which is the process currently implemented by TTN.

The default value for N is 20 transmissions, while the typical values for margin_db range from 5 to 15 dB [6], [7], [15]. Although SNR_m varies according to channel conditions and node distance to a GW, SNR_{SF} values are predetermined.

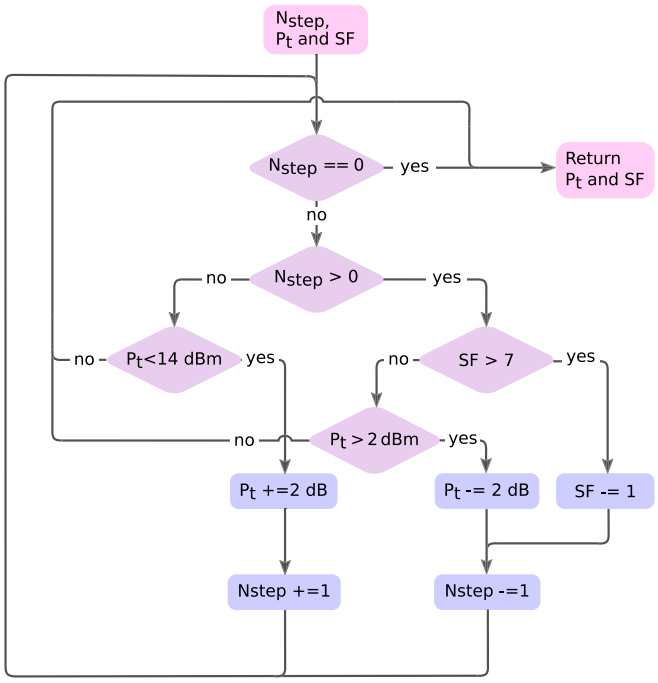

Fig. 2. Flowchart of the NS-side of ADR-TTN [6].

Table I presents values for SNR_{SF} according to the SF in use, considering a fixed BW of 125 kHz for the Semtech SX1272 LoRa radio [19], which is the radio modeled in FLoRa [15]. The sensitivity increases by about 3 dB for each SF increase. Finally, the convergence times for ADR in EDs and in the NS are different, as under the default configuration EDs take at least 64 transmissions to start changing parameters, while the NS only requires 20 received frames [1].

ADR-TTN, the version implemented by TTN, is the most known implementation in the current literature. It is interesting to note that this algorithm does not increase SF at the NS-side to prevent misuse of lower data rates [7], which result in longer ToA. If the node has a bad connection, the NS may only step up its P_t for the next transmissions. Increasing SF is exclusively done at the ED-side. Moreover, ADR-TTN is designed so that nodes fully explore the discrete P_t values when using SF7, and only use the maximum power $P_{t_{\text{max}}}$ with higher SFs. That is because using lower P_t with higher SF is less energy efficient from a connectivity viewpoint, while in terms of sensitivity being as effective as using lower SFs with higher P_t.

The ADR+ algorithm, introduced in [15], is similar to ADR-TTN, except for the channel estimation, since ADR+ considers that SNR_m takes the mean SNR value of the N most recent transmissions. ADR+ outperforms ADR-TTN in terms of DER and energy efficiency in most situations since it is more conservative in resource allocation.

A. ADR-TTN *and* ADR+ *Simulation Results*

In this section, we present simulation results for ADR-TTN and ADR+ in an extended version of FLoRa [15], modified to include the Rayleigh fading channel. The path-loss attenuation model utilizes real data from [20]. Table II presents the

TABLE I
SENSITIVITY SNR_{SF} OF SX1272 LORA RADIO ACCORDING TO SF FOR A FIXED BW = 125 KHZ [19]

SF	SNR_{SF} (dB)
7	-7.5
8	-10
9	-12.5
10	-15
11	-17.5
12	-20

TABLE II
APPLICATION AND PATH-LOSS PARAMETERS

Parameter	Value
n	200
R	1500, 2500, 3500, 4500, 5500 m
DC	0.1% referred to SF12
CR	4/8
BW	125 kHz
$1/\lambda_0$	100 s
$1/\lambda_1$	1200 s
p_a	8 symbols
p_l	20 bytes
$\overline{PL}(d_0)$	128.95 dB
η	2.32
d_0	1000 m

application and path-loss parameters chosen for the setups, where n is the number of nodes in the simulations, R is the maximum distance to the GW, DC is the duty cycle restriction, CR is the CR, BW is the BW, $1/\lambda_0$ and $1/\lambda_1$ are the means of the event generation exponential distributions, p_a is the number of symbols in a packet preamble, p_l is the number of bytes in the packet payload, $\overline{PL}(d_0)$ is the power attenuation for the reference distance d_0, and η is the attenuation coefficient. We performed ten independent simulations for each setup. Each simulation lasts for 20 simulated days, but statistics from the first ten days are considered as a warm-up period and discarded in the general results. In the graphs, lines represent the averages, and error bars indicate the 95% confidence intervals.

Initially, we set the parameter margin_*db* to 10 dB, following the typical value presented in [7]. Fig. 3 shows the average network DER when R varies from 1500 to 5500 m in steps of 1000 m, considering ADR-TTN, ADR+, and no ADR. In the "no ADR" setup, both SF and P_t are randomly and uniformly distributed among EDs. For shorter distances, ADR-TTN and ADR+ perform similarly and, overall, both are worse than the setup with no ADR. Interestingly, the DER of ADR+ improves as the distance increases since nodes are allowed to explore higher SFs without being hardly punished by collisions when the number of users is sufficiently low. However, this is insufficient to provide a reliable service.

The reason why ADR-TTN and ADR+ have worse DER performance than no ADR is the aggressive SF and P_t allocation performed by these algorithms. As shown in Fig. 4, in a deployment with $R = 1500$ m practically every node transmits using SF7 and the lowest P_t values available. ADR+ has a slightly better performance than ADR-TTN by having more transmissions using a higher P_t. The use of lower SF and P_t is expected as these algorithms, especially ADR-TTN, envision the minimization of the channel usage to increase battery life.

Using lower SF and P_t results in lower energy consumption by all nodes, as presented by the lower limit (LL), first quartile ($Q1$), mean, third quartile ($Q3$), upper limit (UL) of the

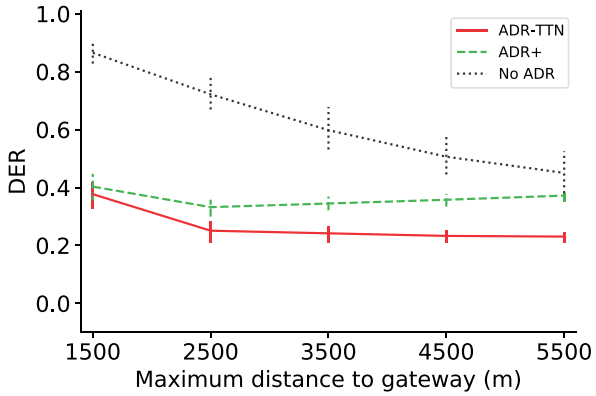

Fig. 3. Average DER using for margin_*db* = 10 dB, and varying the maximum distance of nodes to the GW.

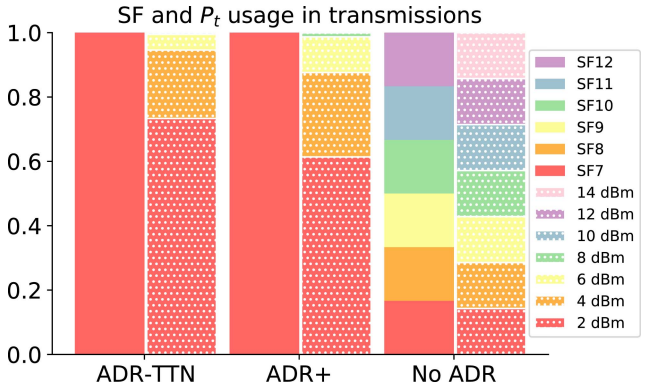

Fig. 4. Usage of SF and P_t when margin_*db* = 10 dB for ADR-TTN, ADR+, and no-ADR with $R = 1500$ m.

TABLE III
ENERGY CONSUMPTION DISTRIBUTION WHEN margin_*db* = 10 DB IN A $R = 1500$ M DEPLOYMENT

Algorithm	LL (J)	Q1 (J)	Mean (J)	Q3 (J)	UL (J)	DER (%)
ADR-TTN	57.07	60.17	61.24	62.28	65.39	37.70
ADR+	56.89	60.17	61.27	62.39	65.61	40.32
No ADR	57.06	66.16	78.82	120.01	200.71	64.35

distributions of the three methods in Table III. The last column of the table presents the final DER achieved with each algorithm. However, this lower energy consumption is not necessarily translated into energy efficiency because the DER is relatively low. By adopting a more conservative approach in estimating the channel conditions, ADR+ has a better SNR margin to work with than ADR-TTN. This extra SNR margin is the motivation to modify the margin_*db* parameter in the algorithm, aiming for better DER results while maintaining low energy consumption.

The margin_*db* variable in ADR impacts the allocation of the transmission parameters. Fig. 5 shows how varying margin_*db* (same for all nodes) between 5 and 40 dB impacts the average DER of a network with $R = 1500$ m. By optimizing this parameter, it is possible to achieve higher DER than with no ADR while maintaining a lower energy consumption. For instance, with the optimum margin_*db* that enables a 90% DER, the distribution of the energy consumed by the nodes is similar to the one presented earlier, except for few outliers.

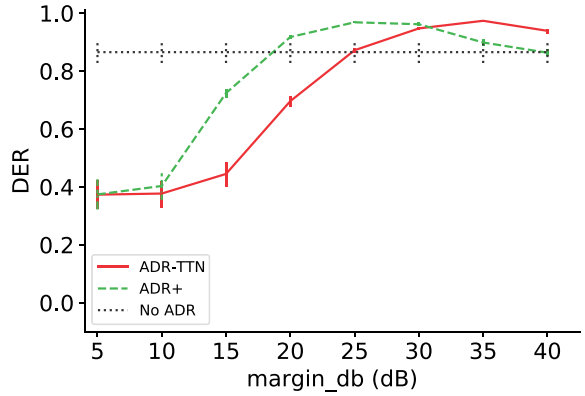

Fig. 5. Impact on average DER when varying margin_*db*.

Compared to ADR-TTN, ADR+ achieves the target 90% DER at a lower margin_*db*. Since ADR+ naturally considers smaller values of SNR to perform its calculations, the additional margin needed for the algorithm to reliably estimate the channel condition is lower. Even though choosing inadvertently high margin_*db* values for the algorithm will likely provide good DER results, this strategy might backfire, as nodes will not transmit in lower SFs, increasing energy consumption and collisions, which is clear for margin_*db* values greater than 30 dB for ADR+ and 35 dB for ADR-TTN.

IV. ADRX

After an extensive simulation campaign with the ADR+ algorithm, we noticed that the margin_*db* parameter can be tuned to achieve a desired performance level. This parameter compensates for the inaccuracy of the channel quality estimate. If a node fails to deliver a considerable amount of frames due to outages, the NS may use biased SNR values, as frames with RSSI lower than the radio sensitivity were not received and are not considered in the estimate. It is possible to accurately estimate the RSSI distribution of a biased set of measurements through curve fitting [21]. However, this can add too much complexity, while being model-dependent. Furthermore, this optimization requires previous knowledge of the network, which may not be available. Also, choosing the same margin_*db* value for all nodes may result in overestimated parameters for nodes near the GWs. To solve both problems, we propose ADRX.

Instead of manually selecting the value of margin_*db* for each node at their deployment, in ADRX, all nodes start with the same margin_*db* value and adapt it every time the NS computes the ADR (i.e., at every N successfully received uplinks). A reference DER (DER_{ref}) is defined and, if the DER achieved between ADR calculations (DER_{inst}) is lower than DER_{ref}, then a new margin_*db* is computed by summing 5 dB to its current value, with an UL of 30 dB. As the DER_{inst} metric does not always accurately represent the actual DER of the device, a safety margin is considered before reducing margin_*db*. If DER_{inst} is greater than $1.15\times\text{DER}_{\text{ref}}$, then ADRX subtracts 2.5 dB from margin_*db* to prevent nodes from saturating at high margin_*db* values while ensuring a satisfactory performance.[2] The LL of margin_*db* is 5 dB. The rest of the algorithm is identical to ADR+.

Algorithm 1 ADRX

Input Variables: $FCnt$, $margin\_db$
Initialization: $I \leftarrow 0$
1: $FCntUp[I] \leftarrow FCnt$
2: $I = I + 1$
3: **if** $I < N$ **then**
4: $\quad margin\_db \leftarrow margin\_db$
5: **else**
6: $\quad$ I = 0
7: $\quad DER_{inst} \leftarrow N/(FCntUp[N-1] - FCntUp[0])$
8: $\quad$ **if** $DER_{inst} < DER_{ref}$ **and** $margin\_db < 30$ **then**
9: $\quad\quad margin\_db \leftarrow margin\_db + 5$
10: $\quad$ **else if** $DER_{inst} > 1.15 \times DER_{ref}$ **and** $margin\_db > 5$ **then**
11: $\quad\quad margin\_db \leftarrow margin\_db - 2.5$
12: $\quad$ **else**
13: $\quad\quad margin\_db \leftarrow margin\_db$
14: $\quad$ **end if**
15: **end if**

The ADRX-specific part of the ADR algorithm is presented in Algorithm 1. The algorithm uses as input variables the current margin_*db* of the transmitting device and the frame counter ($FCnt$) of the received frame. Initially (line 1), the algorithm stores the frame counter in the corresponding position of $FCntUp$, which is a vector to store the frame counters of the N most recently received LoRaWAN frames from each device. After increasing the I index variable (line 2), ADRX checks if N frames have already been received (line 3). If the server has not received N frames yet, then the margin_*db* keeps its current value (line 4) and the algorithm ends. Otherwise, when the server reaches N received frames (line 5), the algorithm checks if margin_*db* needs update. First, I is reset (line 6) to indicate that after this check the server needs to wait for N new frames before checking margin_*db* again. Second, the algorithm computes the instantaneous DER (DER_{inst}, line 7), i.e., the ratio of N and the difference between the most and least recently stored frame counters, which indicate the number of transmissions the node made to successfully deliver N frames. If DER_{inst} is below the target DER (DER_{ref}) and margin_*db* still can be increased (line 8), then margin_*db* is incremented by 5 (line 9). Otherwise, if DER_{inst} is sufficiently above the reference and margin_*db* is at least 5 (line 10), then it is decremented by 2.5 (line 11). Finally, if DER_{inst} is in the desired range (line 12), then margin_*db* remains unchanged (line 13). It is important to note that ADR algorithms must have low complexity because they run very often at the NS.

From Algorithm 1, ADRX is $\mathcal{O}(1)$, i.e., it has constant time and space complexities for constant N, and therefore,

[2]In our extensive simulations, we noticed that the increase of margin_*db* by 5 dB steps is sufficient to provide a significant improvement in DER, and margin_*db* values greater than 30 dB have no positive impact in the final result. Similarly, a margin of 15% in DER_{ref} is found to be sufficient to start decreasing margin_*db* by 2.5 dB, which results in a slight deterioration of DER that can be corrected in further calculations if necessary.

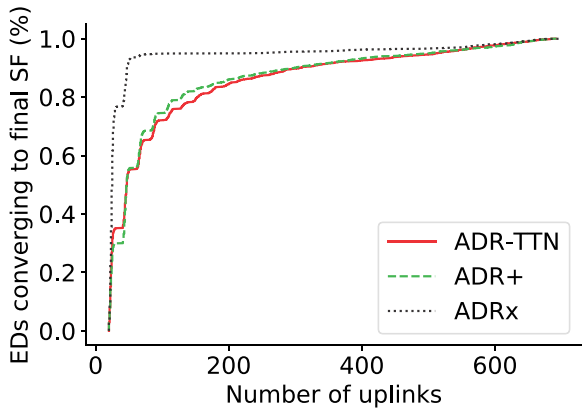

Fig. 6. Percentage of nodes that require a certain number of uplinks to converge to their final SF, when considering a DER_{ref} of 90%, in a R = 1500 m deployment.

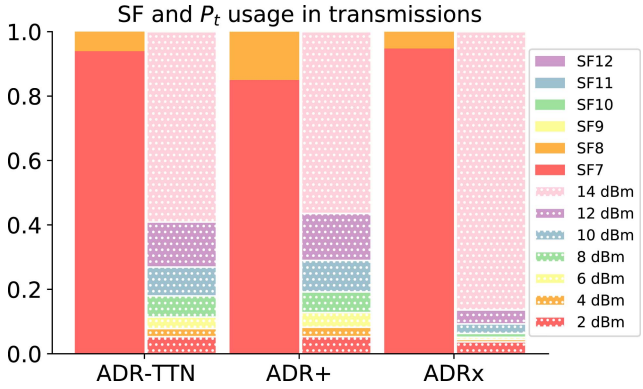

Fig. 7. Usage of SF and P_t for a target DER of 90% when margin_*db* is optimized offline for ADR-TTN and ADR+, and selected dynamically for each node with ADRx. R = 1500 m.

implementing it in the NS is simple. Moreover, note that the proposed method can be directly applied to multiple GWs scenarios, as it is also the case for ADR-TTN and ADR+. However, in this article, we focus on the single GW scenario, leaving the multiple GW scenario for future work.

A. ADRx *Simulation Results*

The results in Section III-A highlight that it is possible to improve performance by adequately tuning the margin_*db* parameter. For the same deployment with R = 1500 m, we select DER_{ref} = 90% for ADRx. For the other algorithms, we performed multiple simulations varying margin_*db*, and ended selecting constant margin_*db* values of 26 and 19 dB for ADR-TTN and ADR+, respectively, as in average, these margins resulted in a final DER of 90%. We performed simulations lasting for 20 simulated days considering the new margin_*db* values to determine a good warm-up period. In Fig. 6, we plot the number of uplink transmissions required for each algorithm to converge its SF allocation. As in Section III, we decided to keep a ten days warm-up (approx. 350 uplinks), as by then the algorithms are sufficiently distant from the transient period and closer to convergence. Interestingly, in this case, ADRx converges faster than the other algorithms. As lower SFs are required and the slowest side of ADR (at the ED) does not need to take any action to increase the SF, ADRx starting in a less conservative way compared to the others comes in as an advantage. With lower margin_*db* values, nodes that would require lower transmit parameters to achieve the desired performance are assigned such parameters earlier on, while further SF changes only take place for those nodes with poorer link quality. On the other hand, ADR-TTN and ADR+ have higher margins from the beginning, tending to hold to overestimated parameters, requiring higher link quality to effectively change SF to a lower, more suitable one.

Fig. 7 presents the SF and P_t usage by each algorithm, while Table IV presents the consequent energy consumption. All algorithms have similar energy consumption, as the main difference in the resource allocation lies in P_t usage, which has less impact on consumption compared to SF. It is remarkable that the proposed method, ADRx, succeeds in dynamically allocating SF and P_t while achieving the same performance level that ADR and ADR+ would only obtain after an offline optimization.

TABLE IV
ENERGY CONSUMPTION DISTRIBUTION FOR A DER_{ref} = 90% IN ADRx, AND margin_*db* OPTIMIZED OFFLINE FOR ADR-TTN AND ADR+, WITH R = 1500 M

Algorithm	LL (J)	Q1 (J)	Mean (J)	Q3 (J)	UL (J)	DER (%)
ADR-TTN	57.92	61.89	63.22	64.55	68.53	89.31
ADR+	58.35	62.18	63.49	65.01	69.17	89.82
ADRx	58.61	62.22	63.38	64.64	68.22	90.61
No ADR	57.06	66.16	78.82	120.01	200.71	64.35

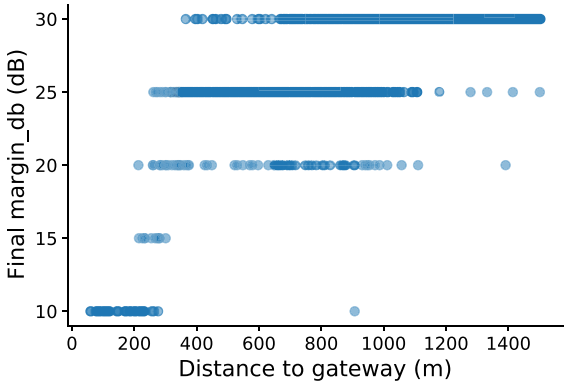

Fig. 8. Scatter plot of the final value of margin_*db* for DER_{ref} = 90% with ADRx. R = 1500 m.

Even though ADRx uses the average SNR of the N most recent transmissions, some nodes have margin_*db* values different than 19 dB (the optimized margin in ADR+), as Fig. 8 shows. Moreover, Fig. 8 also shows that in ADRx, the margin_*db* parameter increases with the distance to the GW. However, since the fading model utilized in this simulation is the same for all nodes, the statistical variation around the mean RSSI should also be the same to all nodes, and therefore they should converge to the same margin_*db* values. This apparent contradiction can be explained because, as discussed in Section IV, the RSSI measurements are biased due to the radio sensitivity, thus the need for higher margin_*db* values as nodes get further away from the GW. The increase of the margin_*db* of nodes far from the GW ensures not only that

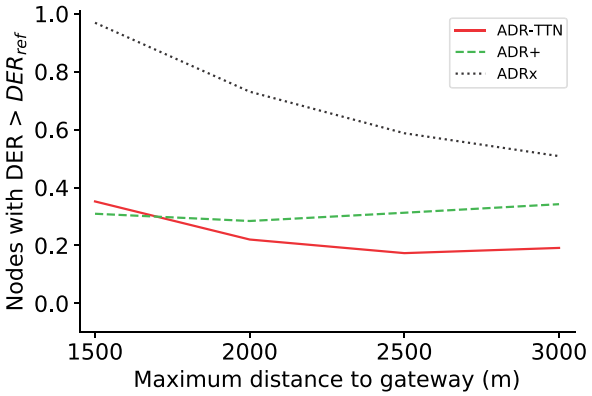

Fig. 9. Percentage of nodes that achieve a final DER higher than the reference of 80% using all three algorithms.

nodes select better transmission parameters but also maintain these parameters even if the estimate improves.

To further evaluate the performance of ADRX, we select different values for DER_{ref} and the maximum distance R. Since the distances to the GW are longer than $R = 1500$ m, we select a less rigorous DER_{ref} of 80%. After multiple simulations with different margin_*db* values, we select 22 and 16 dB for ADR-TTN and ADR+, respectively, as with these values, the majority of simulations lead to an average DER of 80% in a reference deployment with $R = 1500$ m.

Fig. 9 presents a line graph that shows the percentage of nodes achieving DERs higher than the reference when varying the maximum distance to the GW. In distances as short as 1500 m, both ADR-TTN and ADR+ fail to deliver a reliable service to a reasonable amount of nodes. As distance increases, similarly to the first result in the previous section, in higher distances, ADR+ allows nodes to explore higher SF and P_t, but it still fails to keep the majority of nodes above the reference. Differently, with ADRX, almost all nodes achieve a satisfactory result in shorter networks ($R = 1500$ m). Naturally, this portion decreases as distance increases, although remaining considerably higher than the results of other algorithms. The manual, offline optimization in ADR-TTN and ADR+ considers the average DER of all nodes after multiple tests, and is the same for all nodes, whereas the online margin selection of ADRX is at runtime and device specific, ensuring better individual results.

In Table V, we present the energy consumption and average DER of nodes in the $R = 3000$ m deployment, also comparing the algorithms to the case without ADR. The increase in energy consumption is expected, as the maximum distance to the GW increases, therefore, higher SF and P_t are required. ADR+ and ADR-TTN achieve satisfactory DERs and have similar energy consumption. While ADR-TTN fails to achieve the reference DER, it consumes less energy compared to the case without ADR, in which not only the energy consumption is higher, but the DER is significantly worse than in the other cases.

Next, we select DER_{ref} of 70%, 80%, and 90% while varying the distance R for ADRX. The average DER of nodes is presented in Fig. 10. For less demanding references, nodes with ADRX provide consistent DER in longer ranges. Even

TABLE V
ENERGY CONSUMPTION DISTRIBUTION FOR A $\text{DER}_{\text{ref}} = 80\%$ IN ADRX, AND margin_*db* OPTIMIZED OFFLINE FOR ADR-TTN AND ADR+, WITH $R = 3000$ M

Algorithm	LL (J)	Q1 (J)	Mean (J)	Q3 (J)	UL (J)	DER (%)
ADR-TTN	57.35	63.32	66.50	71.28	83.10	75.61
ADR+	57.35	63.38	67.95	75.93	94.13	78.73
ADRX	57.88	63.84	68.03	74.68	90.90	79.29
No ADR	57.42	66.31	79.25	119.01	197.67	64.35

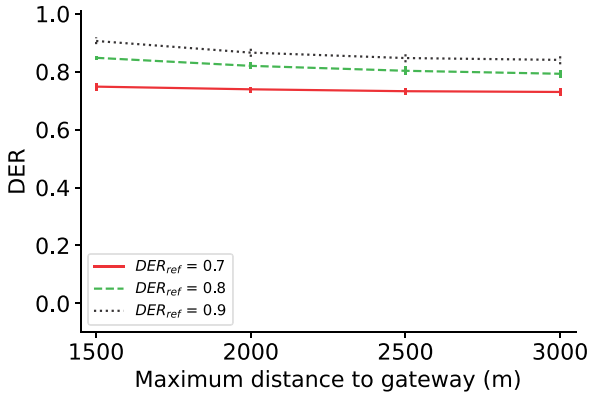

Fig. 10. Average DER of nodes when varying R for different DER_{ref} using ADRX.

when using a strict reference of 90%, the performance of the network remains close to the desired, with average DERs of 84.72% and 84.08% when the network radius is of 2500 and 3000 m, respectively. This demonstrates ADRX robustness to distance variation as long as the required DER is feasible.

Finally, although we have performed simulations for larger number of nodes (up to 1000), the differences in results are not expressive (degradation of less than 2% in the final DER) and were omitted for the sake of brevity.

V. CONCLUSION

In this article, we investigated the impact of a largely neglected parameter in LORAWAN ADR, margin_*db*. This parameter serves as a safety margin on the link quality estimate performed by the ADR algorithm. We showed that the typical values of margin_*db* are not suited when the medium is too lossy, as the margin will not compensate for the optimistic estimate, causing ADR-TTN (and, to a lesser extent, ADR+) to assign transmission parameters which cannot provide reliable communication in most cases. By optimizing the margin_*db* value, it is possible to achieve better performance while maintaining a low energy consumption and longer battery lifetime. However, such optimization may not be possible in many applications, as it requires prior knowledge of the network to be both effective in service and energy efficiency.

To solve these problems, we proposed ADRX, in which the margin_*db* parameter is selected online and to each node individually by considering the number of lost frames within ADR executions. Through experiments using the FLORA simulator, we verified that ADRX achieves better performance than ADR-TTN and ADR+ present after an offline optimization of the margin_*db* parameter, which ADRX does not require since it selects margin_*db* online.

REFERENCES

[1] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 855–873, 2nd Quart., 2017.

[2] *LoRaWAN™ 1.0.3 Specifications*, LoRa Alliance, Fremont, CA, USA, Jul 2018.

[3] *AN120.22 LoRa Modulation Basics*, Semtech Coorp., Camarillo, CA, USA, Mar. 2015.

[4] M. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa low-power wide-area networks scale?" in *Proc. 19th ACM Int. Conf. Model. Anal. Simulat. Wireless Mobile Syst.*, 2016, pp. 59–67.

[5] *RP002-1.0.0 LoRaWAN Regional Parameters*, LoRa Alliance, Fremont, CA, USA, 2019.

[6] *LoRaWAN Adaptive Data Rate*, Things Netw., Amsterdam, The Netherlands, Oct. 2019. [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/adaptive-data-rate.html

[7] *LoRaWAN—Simple Rate Adaptation Recommended Algorithm, Rev. 1.0.*, Semtech, Camarillo, Ca, USA, Oct. 2016.

[8] 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 *Proc. IEEE 13th Int. Conf. Wireless Mobile Comput. Netw. Commun.*, Rome, Italy, 2017, pp. 1–8.

[9] K. Q. Abdelfadeel, V. Cionca, and D. Pesch, "Fair adaptive data rate allocation and power control in LoRaWAN," in *Proc. IEEE 19th Int. Symp. World Wireless Mobile Multimedia Netw.*, Chania, Greece, Jun. 2018, pp. 14–15.

[10] J. Park, K. Park, H. Bae, and C. Kim, "EARN: Enhanced ADR with coding rate adaptation in LoRaWAN," *IEEE Internet Things J.*, early access, Jun. 30, 2020, doi: 10.1109/JIOT.2020.3005881.

[11] S. Kim and Y. Yoo, "Contention-aware adaptive data rate for throughput optimization in LoRaWAN," *Sensors*, vol. 18, no. 6, p. 1716, May 2018.

[12] J. M. Marais, R. Malekian, and A. M. Abu-Mahfouz, "Evaluating the LoRaWAN protocol using a permanent outdoor testbed," *IEEE Sensors J.*, vol. 19, no. 12, pp. 4726–4733, Jun. 2019.

[13] J. Finnegan, R. Farrell, and S. Brown, "Analysis and enhancement of the LoRaWAN adaptive data rate scheme," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7171–7180, Aug. 2020.

[14] S. Li, U. Raza, and A. Khan, "How agile is the adaptive data rate mechanism of LoRaWAN?" in *Proc. IEEE Global Commun. Conf.*, Abu Dhabi, UAE, 2018, pp. 206–212.

[15] M. Slabicki, G. Premsankar, and M. D. Francesco, "Adaptive configuration of LoRa networks for dense IoT deployments," in *Proc. IEEE/IFIP Netw. Oper. Manag. Symp. (NOMS)*, Taipei, Taiwan, Apr. 2018, pp. 1–9.

[16] P. Garg, "Performance evaluation of a LoRaWAN towards development of an optimised ADR (adaptive data rate) model," M.S. thesis, Dept. Embedded Syst., Delft Univ. Technol., Delft, The Netherlands, 2019.

[17] A. Goldsmith, *Wireless Communications*. New York, NY, USA: Cambridge Univ. Press, 2005.

[18] L. Casals, B. Mir, R. Vidal, and C. Gomez, "Modeling the energy performance of LoRaWAN," *Sensors*, vol. 17, no. 10, p. 2364, 2017.

[19] *SX1272/73 Datasheet, Rev. 3.1.*, Semtech, Camarillo, CA, USA, Mar. 2017,

[20] J. Petäjäjärvi, K. Mikhaylov, M. Hämäläinen, and J. Iinatti, "Evaluation of LoRa LPWAN technology for remote health and wellbeing monitoring," in *Proc. 10th Int. Symp. Med. Inf. Commun. Technol.*, Worcester, MA, USA, Mar. 2016, pp. 1–5.

[21] M. R. Kielgast, A. C. Rasmussen, M. H. Laursen, J. J. Nielsen, P. Popovski, and R. Krigslund, "Estimation of received signal strength distribution for smart meters with biased measurement data set," *IEEE Wireless Commun. Lett.*, vol. 6, no. 1, pp. 2–5, Feb. 2017.

Gabriel Germino Martins de Jesus was born in Florianópolis, Brazil. He is currently pursuing the B.Sc. degree in electrical engineering with the Federal University of Santa Catarina, Florianópolis.

Richard Demo Souza (Senior Member, IEEE) received the D.Sc. degree in electrical engineering from the Federal University of Santa Catarina (UFSC), Florianópolis, Brazil, in 2003.

From 2004 to 2016, he was with the Federal University of Technology—Paraná (UTFPR), Curítiba, Brazil. Since 2017, he has been with UFSC, where he is a Professor. His research interests are in the areas of wireless communications and signal processing.

Prof. Souza is a co-recipient of the 2014 IEEE/IFIP Wireless Days Conference Best Paper Award, the supervisor of the awarded Best Ph.D. Thesis in Electrical Engineering in Brazil in 2014, and the 2016 Research Award from the Cuban Academy of Sciences. He has served as an Editor or an Associate Editor for the *SBrT Journal of Communications* and *Information Systems*, the IEEE COMMUNICATIONS LETTERS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and the IEEE INTERNET OF THINGS JOURNAL.

Carlos Montez (Member, IEEE) received the B.Sc. degree in computer science from the Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil, in 1988, and the M.Sc. and D.Eng. degrees from University of Santa Catarina (UFSC), Florianópolis, Brazil, in 1995 and 1999, respectively.

He is a Full Professor and a Researcher with the Automation and System Department, UFSC. Since 2005, he has advised and co-advised 23 Masters and eleven Ph.D. students. He has authored or coauthored more than 130 papers in the area of wireless sensor networks and real-time communication. His research interests include wireless sensor networks, industrial communication protocols, and big data sensing.

Arliones Hoeller, Jr. (Member, IEEE) was born in Blumenau, Brazil. He received the B.Sc. and M.Sc. degrees in computer science from the Federal University of Santa Catarina, Florianópolis, Brazil, in 2004 and 2007, respectively, where he is currently pursuing the D.Sc. degree in electrical engineering.

From 2007 to 2013, he was a Software Engineer and a Researcher in Brazil, working on telecom and embedded systems projects. Since 2013, he has been a Lecturer with the Federal Institute for Education, Science, and Technology of Santa Catarina, São José, Brazil. From 2018 to 2020, he was a Visiting Researcher with the Centre for Wireless Communications, University of Oulu, Oulu, Finland. His research interests are wireless communications and networks, and distributed embedded real-time systems.