



FioRa+: Empowering Energy Neutrality-aware Multicast Firmware Distributions in Energy-harvesting LoRa Networks

SUKANYA JEWSAKUL, Department of Computer and Electrical Engineering, Mid Sweden University, Sundsvall, Sweden

SEBASTIAN BADER*, Department of Computer and Electrical Engineering, Mid Sweden University, Sundsvall, Sweden

EDITH C. H. NGAI*, Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong, Hong Kong

Efficient firmware distributions in energy-harvesting (EH) LoRa networks require that EH LoRa sensors simultaneously receive data fragments from a server without facing power failures. This requirement is difficult to satisfy due to the impact of EH rates and LoRa transmission parameters on the efficiency of firmware distributions. We present FioRa+, a novel energy neutrality-aware multicast firmware distribution framework for EH LoRa networks. It gradually distributes a firmware image to EH LoRa sensors in an energy-neutral manner according to their future energy availability predicted using embedded machine learning models. Consequently, the need for additional firmware distributions caused by unsuccessful firmware image reconstructions is reduced. Through one-hop neighbor discovery, on-demand relay, flexible energy query, and coverage assessment mechanisms, FioRa+ ensures that all EH LoRa sensors can receive data fragments from the server at the scheduled time using high data rates. Equipped with a relay scheduling algorithm, it circumvents the collision of data fragments relayed by EH LoRa sensors using identical data rates. The experimental results show that FioRa+ renders up to 113× shorter distribution time and 22.7× less distribution overhead than the state of the art.

CCS Concepts: • **Networks** → **Network management**; • **Computer systems organization** → **Sensor networks**; • **Hardware** → **Renewable energy**.

Additional Key Words and Phrases: LoRa, multicast, over-the-air firmware update, energy harvesting, energy neutrality

1 Introduction

The prolonged lifetime of energy-harvesting (EH) LoRa networks necessitates the use of remote programming to make firmware updates available throughout the deployment. Nonetheless, performing firmware updates over the air (FUOTA) to completion in such networks is challenging because EH LoRa sensors are subject to power failures which bring about *intermittent operation* [15, 17, 22]. In EH LoRa networks that execute server-initiated multicast firmware distribution packages [7, 9], the intermittent operation causes EH LoRa sensors to fail to receive firmware data distributed by the server whenever they run out of harvested energy and temporarily shut down. Given EH LoRa sensors' harvested-energy availability varies across time and space [17, 18], the multicast

*S. Bader and E. C. H. Ngai are co-corresponding authors of this article.

This research was financially supported in part by the UGC General Research Fund No. 17203320 and No. 17209822, and Innovation and Technology Fund ITS/383/23FP from Hong Kong, and by the Knowledge Foundation under Grant No. 20180170 (NIIT).

Authors' Contact Information: S. Jewsakul and S. Bader (co-corresponding author), Department of Computer and Electrical Engineering, Mid Sweden University, Sundsvall, Sweden; emails: {sukanya.jewsakul, sebastian.bader}@miun.se; E. C. H. Ngai (co-corresponding author), Department of Electrical and Electronic Engineering, The University of Hong Kong, Pokfulam, Hong Kong; email: chngai@eee.hku.hk;



This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

© 2025 Copyright held by the owner/author(s).

ACM 1550-4867/2025/6-ART

<https://doi.org/10.1145/3744741>

firmware distributions are thus deemed inefficient and unreliable if EH LoRa sensors have inadequate energy to simultaneously receive firmware data from the server during a configured multicast session. Consequently, the server may be required to perform additional firmware distributions until all EH LoRa sensors in a multicast group can reconstruct firmware images, increasing the overhead of firmware distributions. Besides, using energy-inefficient LoRa transmission parameters during multicast firmware distributions exacerbates power failures at EH LoRa sensors and delays the completion time of firmware updates.

To alleviate such power failures, energy-saving techniques may be applied to minimize the energy consumption of EH LoRa sensors during the configured multicast session. For example, the server may use high data rates (DRs) to enable fast and energy-efficient multicast firmware distributions at the expense of firmware distribution coverage [1]. Additionally, it may employ differencing and compression algorithms [27, 31] to decrease the size of firmware data to be distributed. After receiving the firmware data, each EH LoRa sensor may shorten the time spent erasing and writing flash memory [22] to further minimize energy consumption. Despite being beneficial, the existing energy-saving techniques are, however, originally designed for battery-powered LoRa networks and/or neglect the spatio-temporal heterogeneity of EH rates among EH LoRa sensors during multicast firmware distributions. In addition, they cannot ascertain if EH LoRa sensors can definitely be active at the scheduled time. Enabling efficient multicast firmware distributions in the face of intermittent operation thus remains an open problem.

We propose *FioRa+*, a novel energy neutrality-aware multicast Firmware distribution framework for EH LoRa networks. As the name suggests, it is an extension of our previous work called *FioRa* [19]. Aiming to minimize both the completion time of firmware update and the overhead of firmware distributions, *FioRa+* allows the server to, at the highest possible DR, distribute firmware data (i.e., data fragments) in an energy-neutral manner according to sensors' predictive energy availability. Given EH LoRa sensors may not have enough energy to stay active and receive all data fragments simultaneously, the server spreads firmware distributions out across multiple multicast sessions. For each multicast session, the session duration and the size of the firmware data to be distributed are decided based on the sensors' predictive energy availability and the DR in use. Using the predictive (i.e., future) energy availability is beneficial because, following the specification [9], the multicast session is usually configured in advance [1]. To speed up the completion of firmware updates, the server uses the highest possible DR (i.e., high DRs feature large data payloads, short airtime, and low energy consumption) for multicast firmware distributions, so as to maximize firmware data received by EH LoRa sensors per each multicast session. Unlike *FioRa*, *FioRa+* further allows the server to adaptively schedule multicast firmware distributions at any time when a data reception rate is maximized.

Indeed, enabling energy neutrality-aware multicast firmware distributions in EH LoRa networks using high DR raises four main challenges. First, it is required that, despite using high DR, the multicast firmware distributions must cover all EH LoRa sensors in the multicast group. Second, to avoid power failures during multicast firmware distributions, it is essential for the server to obtain accurate information about EH LoRa sensors' future harvested energy availability at a low cost. Third, considering that EH LoRa sensors operate intermittently (i.e., they may not have adequate energy to be active at the same time), it is crucial for the server to figure out when the multicast firmware distributions should be scheduled. Fourth, given the heterogeneity of EH rates among EH LoRa sensors, the server must determine a session configuration that ensures the coverage of multicast firmware distributions and minimizes the completion time of firmware updates without causing power failures.

To tackle the first challenge, we equip *FioRa+* with two novel mechanisms called *on-demand relay* and *one-hop neighbor discovery*, both of which operate in an energy-neutral manner. Through the former mechanism, *FioRa+* allows EH LoRa sensors (aka relay sensors) to forward data fragments directly received from the server, provided that the multicast firmware distributions performed by the server do not reach all sensors in the multicast group. Assisted by the latter mechanism, *FioRa+* ensures that, after the former mechanism is performed, all the unreachable EH LoRa sensors are ultimately covered by the multicast firmware distributions. Given recent

Table 1. The Number of LoRa Packets and The Time Required to Distribute The Firmware Image of 50 kB Using Different DRs under 10% DC Restrictions in EU868 Frequency Plan [8]

DR	BW (kHz)	SF	Bit Rate (bps)	Payload Size (B)	Airtime (s)	Packets	Duration (h)
0	125	12	250	48	2.79	1042	8.1
1	125	11	440	48	1.48	1042	4.3
2	125	10	980	48	0.7	1042	2.0
3	125	9	1760	112	0.68	447	0.8
4	125	8	3125	239	0.7	210	0.4
5	125	7	5470	239	0.39	210	0.2

advances in embedded machine learning (ML), EH LoRa sensors start training/updating embedded ML models for accurate EH predictions [11, 26] and management [17, 18, 20] locally and never transmit EH information to the server. To address the second challenge and the third challenge, we thus supply FioRa+ with a *flexible energy query* mechanism. This mechanism enables the server to adaptively obtain information concerning the predictive harvested-energy availability and the good timing of firmware update from each EH LoRa sensor before configuring a multicast session. Considering an array of time-slotted predictive energy availability reported, the good timing of firmware update regards as the time when EH LoRa sensors' data reception rates can be maximized. As for the fourth challenge, we furnish FioRa+ with an *adaptive multicast session management* algorithm. This algorithm helps the server select the good timing, the relay sensors, and the DRs (i.e., the server and the relay sensors may use different DRs for multicast firmware distributions) that maximize the data reception rate while ensuring the full coverage of multicast firmware distributions. To alleviate signal collisions at out-of-range sensors (i.e., EH LoRa sensors that receive firmware data from relay sensors), we further provide the server with a relay scheduling algorithm. With this algorithm, the server can spread data fragments relayed using the same DR (i.e., the same spreading factor (SF) and bandwidth) over different time slots to avoid signal collisions.

Through trace-driven simulations, the experimental results show that FioRa+ minimizes the distribution time of firmware images by up to 113 \times and the overhead of firmware distributions by up to 22.7 \times compared with the state of the art. With the flexible energy query mechanism, it also reduces unsuccessful energy queries by up to 3 \times compared with FioRa.

In summary, we make three major contributions:

- We propose FioRa+, the first energy neutrality-aware multicast firmware distribution framework for EH LoRa networks. By selecting the update timing and the session configuration that maximize the firmware data distributed in an energy-neutral manner, FioRa+ speeds up the completion of firmware updates while minimizing the overhead of firmware distributions.
- We present novel one-hop neighbor discovery, flexible energy query, and time-slotted on-demand relay mechanisms as part of FioRa+. Together, these mechanisms ensure that the multicast group of EH LoRa sensors can successfully receive the maximum amount of firmware data at the scheduled time using high DRs.
- We investigate the impact of duty cycle regulations on the performance of multicast firmware distributions in EH LoRa networks. Compared to existing methods, the experimental results illustrate that FioRa+ can successfully distribute the larger sizes of firmware images regardless of duty cycle restrictions.

2 Background and Motivation

This section provides the background information necessary to understand FUOTA specifications [7, 9] for LoRa networks. After discussing the impact of intermittent operation on the efficiency of existing FUOTA solutions, we motivate the need to bring energy-neutral operation (ENO) to multicast firmware distributions in EH LoRa networks.

2.1 Background

The ability to perform FUOTA is the key to secure, up-to-date, and long-lived Internet-of-Things (IoT) systems. Aiming to activate this ability in LoRa networks, LoRa Alliance has rendered two specifications (i.e., *remote multicast setup* [9] and *fragmented data block transport* [7]), both of which run at the application layer of LoRa Wide Area Network (LoRaWAN) [6] protocol stack. Taken together, these specifications enable multicast firmware distributions, defeating the well-known limitations of LoRa networks (e.g., low DRs, duty cycle (DC) restrictions, and limited downlink capacity [1]). We provide an overview of LoRa networks and explain multicast firmware distributions in more detail below.

2.1.1 LoRa Networks. LoRa is a radio technology building on a proprietary modulation technique called chirp spread spectrum (CSS) [28, 32]. It enables energy-efficient, long-range, low data-rate wireless communications [2] for resource-constrained IoT systems. Aiming to trade DR and energy efficiency for communication range and interference immunity [33], LoRa incorporates different transmission parameters that can be fine-tuned. These parameters include transmission power, SF, channel frequency, bandwidth (BW), and coding rate [30]. Following a popular medium access control (MAC) protocol called LoRaWAN [6], SF and BW parameters are systematically organized to represent different DRs. Given a BW is fixed, an increase in SFs (i.e., LoRa supports six common SFs ranging from 7 to 12) lowers the DRs (aka bit rates) [21]. As shown in Table 1, given a BW of 125 kHz, SF7 (denoted as DR5) offers approximately $21\times$ higher bit rates when compared to SF12 (denoted as DR0). Thanks to the orthogonality of SFs, LoRa packets modulated using different SFs can occur simultaneously in the same channel without collisions [3].

A LoRaWAN-based network typically consists of three main components: LoRa sensors, a LoRa gateway, and a server [12]. Laid in a single-hop star topology [4], the LoRa sensors talk to the server via the LoRa gateway [5]. To trade energy consumption for downlink communication capability, LoRaWAN supports three operational modes: Class A, Class B, and Class C [2]. Operating in a Class A mode, a LoRa sensor (a so-called Class A device) spends most of the time asleep. As illustrated in Figure 1a, the server may use one of the two receive windows (i.e., RX1 and RX2) opened after each uplink transmission (TX) to initiate a downlink communication to the LoRa sensor. Unlike Class A sensor, Figure 1b shows that a Class C device/sensor continuously opens a second receive window (RX2) for possible downlink communications. Consequently, the server can reach Class C sensor anytime, provided that it is not transmitting, listening to the first receive window (RX1), or shutting down. A Class B sensor, on the other hand, opens extra receive window(s) at the scheduled time(s) periodically to increase the capability of downlink communications. We refer the reader to the existing LoRaWAN specification [6] for more details on each operational mode. It is worth noting that, due to the complexity of Class B operations, we only focus on Class A and Class C operations in this article.

Regardless of operational modes, in a region where DCs are regulated (e.g., Europe), a radio transmitter (i.e., a LoRa sensor and/or a LoRa gateway) has to wait for a T_{off} duration before being able to initiate the next transmission in the same frequency resource (e.g., the same sub-band) [10]. Equation 1 shows how T_{off} is calculated. Here, T_{on} , T_{off} , and DC denote the time on air (aka airtime), the off time, and the maximum percentage of time whereby the transmitter can use the same frequency resource [2], respectively.

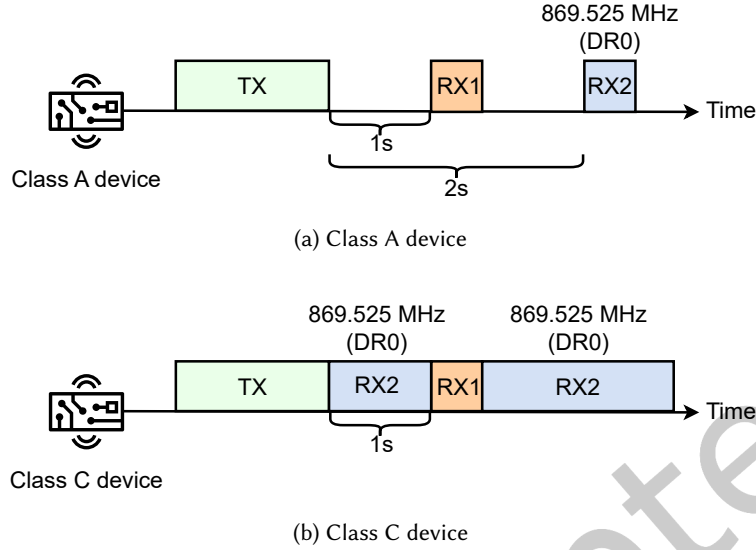


Fig. 1. Examples of how LoRaWAN Class A and Class C devices operate in EU868 frequency plan

$$T_{\text{off}} = \frac{T_{\text{on}}}{\text{DC}} - T_{\text{on}} \quad (1)$$

2.1.2 Multicast Firmware Distributions. To initiate FUOTA processes in LoRa networks, the server identifies a group of LoRa sensors to be updated and generates a firmware image accordingly. If the firmware image does not fit in an application payload (e.g., 51 bytes for DR2 in EU868 frequency plan [8]), the server breaks that firmware image into data fragments of equal size. After adding padding bytes, Table 1 exemplifies that the firmware image of 50 kB can be divided into 1042 data fragments of 48 bytes each (exclusive of a 3-byte-long fragment header [7]) when using DR2. With that, the server generates *coded* data fragments by using a forward error correction code to add redundancy [7]. In the face of packet losses, LoRa sensors can reconstruct the firmware image independently, provided they sufficiently receive the coded data fragments from the server.

As shown in Figure 2, to transmit the (coded) data fragments to the group of LoRa sensors at once, the server first sends two commands to, respectively, program multicast and fragmentation contexts on each LoRa sensor. Upon answering these commands, the LoRa sensor requests the current network time from the server for the sake of clock synchronization. After the LoRa sensor's clock is synchronized, the server sends a multicast session command to instruct the LoRa sensor to start operating in a receiving mode (e.g., a Class C mode) at the agreed time for at most a requested duration. Along with the previous instructions, the server also informs the LoRa sensor of the DR (i.e., the systematic representation of SF and BW) and the channel to listen for the transmitted data fragments. When the scheduled time is reached, the server starts transmitting/distributing the data fragments in a multicast manner while all LoRa sensors in the (multicast) group enter the receive mode accordingly. By sharing the same multicast address, security keys, and session configurations (e.g., DR, channel, and session duration), all LoRa sensors in the multicast group receive data fragments from the server simultaneously as if they are a single device. To conserve the available energy, LoRa sensors may revert to an energy-efficient mode (i.e., Class A) immediately after receiving sufficient data fragments (i.e., firmware data) for firmware image

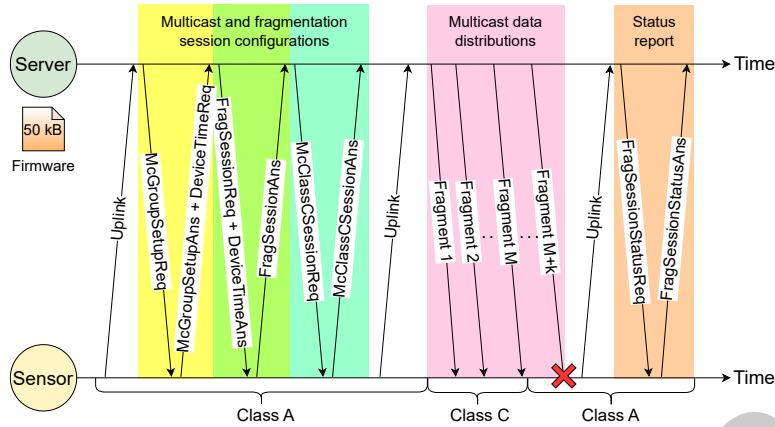


Fig. 2. An overview of multicast firmware distributions in LoRa networks

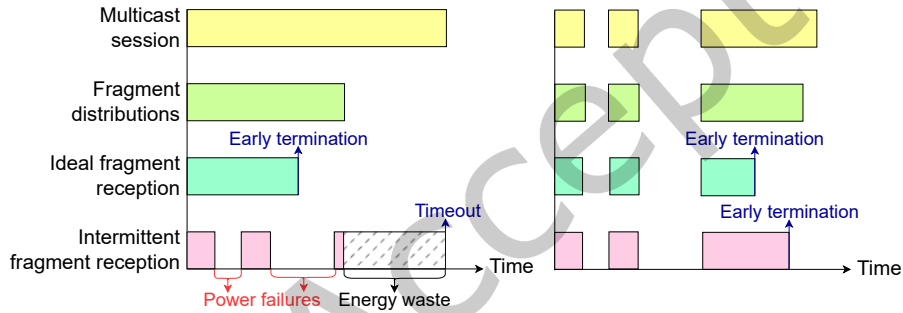


Fig. 3. The impact of power failures on the efficiency of multicast firmware distributions in EH LoRa networks

reconstructions. After multicast firmware distributions are over, the server may send a command to each LoRa sensor in a unicast manner to query its FUOTA status.

2.2 Motivation

Although existing work on FUOTA processes in LoRa networks has standardized application-layer protocols for multicast firmware distributions [7, 9], studied the impact of different transmission parameters on firmware distribution coverage/efficiency [1], or proposed sophisticated techniques to uplift the energy efficiency and reliability of FUOTA processes [27], enabling efficient multicast firmware distributions in EH LoRa networks still remains an open problem. Unlike traditional LoRa sensors that operate reliably until their single-use batteries are exhausted [16], EH LoRa sensors operate intermittently. Namely, they oscillate between functioning and shutting down thanks to the temporary power failures caused by the unpredictability of EH rates [22] and the limited capacity of rechargeable energy storage [18]. Given a mismatch between the harvested energy residing in energy storage and the energy consumed by FUOTA processes (i.e., FUOTA processes are energy consuming [22, 27]), EH LoRa sensors may suffer multiple power failures during multicast firmware distributions. In regions where DC restrictions are regulated, frequent power failures are likely unavoidable because of a relatively long multicast session whereby EH LoRa sensors are required to listen for data fragments. As exemplified in Table 1,

being compliant with 10% DC restrictions, the server takes roughly 8 hours to distribute the firmware image of 50 kB using the lowest DR (i.e., DR0). As exhibited on the left of Figure 3, if all (coded) data fragments are distributed within one long multicast session, EH LoRa sensors with insufficient energy will temporarily shut down and miss distributed data fragments. The insufficient reception of data fragments leads to a failure of firmware image reconstructions and additional firmware distributions that follow.

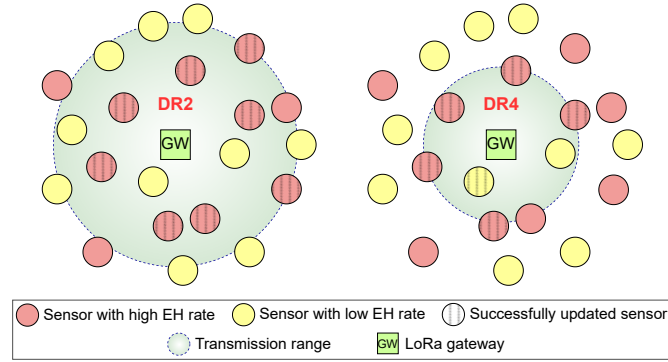
To demonstrate the necessity of our work, the right of Figure 3 exhibits the performance gain that can be obtained if multicast firmware distributions are performed in an energy-neutral manner. By spreading firmware distributions out across multiple multicast sessions, and for each multicast session, matching the energy consumption (i.e., occurrences) of firmware distributions to the harvested energy that EH LoRa sensors mutually have, all EH LoRa sensors can simultaneously receive data fragments from the server without encountering power failures. Consequently, the need for additional firmware distributions caused by insufficient firmware reception is declined. To circumvent a waste of harvested energy, the session duration should exactly accommodate firmware distributions performed in the multicast session. By doing so, EH LoRa sensors can immediately revert to Class A whenever the multicast session expires. Moreover, to make the most of energy harvested by EH LoRa sensors, the server should distribute data fragments using high DR to maximize the firmware data received by EH LoRa sensors per each multicast session. As illustrated in Table 1, using high DRs decreases the time required to distribute the entire firmware image. Therefore, performing multicast firmware distributions in an energy-neutral manner using high DR is essential to minimize the completion time of firmware updates and the overhead of firmware distributions.

3 Related Work

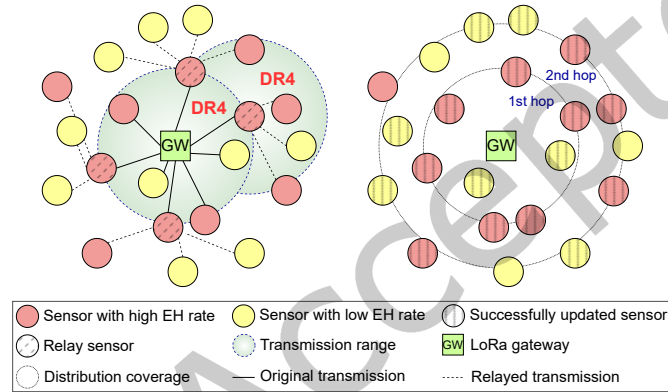
FUOTA processes usually refer to downloading a large file from a remote server to one or more embedded sensors over a wireless link [1, 22]. Nowadays, the file to be downloaded is not limited to a traditional firmware image that facilitates software updates or security patches. With the emergence of embedded ML, embedded sensors in smart IoT applications make use of FUOTA processes to download an embedded ML model [23] or model weights [31] trained on the (edge) server for the sake of incremental learning. Despite their advantages, FUOTA processes are deemed memory intensive, energy-consuming, and sensitive to packet errors/losses [27]. When the embedded sensors are powered by EH, performing FUOTA processes to completion becomes even more challenging because of the frequent power failures [22] that hinder firmware reception and updates [31].

Several FUOTA techniques have recently been proposed to tackle the above challenges. For example, [22] presented a method called *light flash write* to reduce the time spent erasing and writing a firmware image to an EH sensor's flash memory to minimize energy consumption. A *reinforcement* technique is also proposed to allow the EH sensor to rewrite the firmware image using a default flash erase/write configuration to increase data retention time. Regarding the over-the-air update of deep-learning weights on an EH sensor, [31] introduced three techniques called *delta encoding*, *data transmission optimization*, and *runtime support*. These techniques minimize the number of data transmissions while allowing the EH sensor to request a chunk of deep-learning weights based on the available energy and intermittently perform the model update. To enable energy-efficient and reliable FUOTA processes in (battery-powered) LoRa networks, [27] proposed a FUOTA system called FLoRa. It incorporates *delta scripting* and *channel coding* techniques to reduce the size of firmware data to be distributed and deal with packet losses and symbol errors. Moreover, as part of FLoRa, a *beamforming* technique enables the server/gateway to transmit data fragments to a smaller group of LoRa sensors in a unicast manner, provided that they do not sufficiently receive data fragments during multicast firmware distributions.

Unlike the existing work, our proposed framework enables energy neutrality-aware FUOTA in EH LoRa networks. By performing multicast firmware distributions at high DR(s) as per the harvested energy mutually



(a) The impact of DR2 and DR4 on multicast firmware distributions



(b) The impact of a relay mechanism using DR4 on multicast firmware distributions

Fig. 4. The impacts of different DRs and the one-hop relay mechanism on the coverage and the success of multicast firmware distributions

has by EH LoRa sensors, FioRa+ minimizes both the completion time of firmware update and the overhead of firmware distributions.

4 System Model

At the core of FioRa+ is server's ability to spread firmware distributions out across multiple multicast sessions. For each multicast session, the server performs multicast firmware distributions in an energy-neutral manner using high DR. With this ability, the server ensures that, despite having different EH rates, EH LoRa sensors can simultaneously receive as much firmware data as possible without encountering power failures. Nonetheless, using high DR for multicast firmware distributions raises concerns about a decrease in firmware distribution coverage. As exemplified in Figure 4a, despite high DR (i.e., DR4) helps bring the firmware reception/reconstruction at one of EH LoRa sensors with low EH rates to a success, it renders the limited coverage of multicast firmware distributions when compared to low DR (i.e., DR2). Although employing multiple LoRa gateways or a mobile LoRa gateway can increase both the coverage and DR of multicast firmware distributions [1], this solution literally

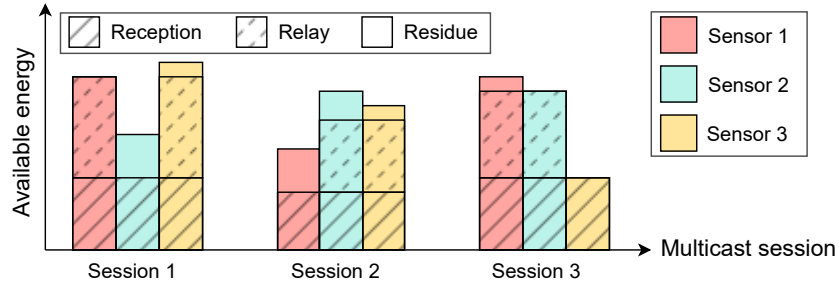


Fig. 5. An overview of energy management technique used for enabling energy neutrality-aware multicast firmware distributions

comes with the higher cost of deployment. To overcome this issue at low cost, we thus equip FioRa+ with a one-hop relay mechanism, so as to expand the coverage of multicast firmware distributions. As illustrated in Figure 4b, this mechanism enables two-hop communications from a LoRa gateway. In addition to expanding the distribution coverage, as also illustrated on the right of Figure 4b, the use of higher DR for relay-enabled multicast firmware distributions likely increases the success of firmware image reconstructions at EH LoRa sensors with low EH rates. Coupled such relay mechanism with careful energy management, FioRa+ makes sure that all the firmware data distributed by the server can be forwarded by relay sensors without incurring power failures. We detail the relay mechanism and the energy management technique below.

4.1 On-demand Relay Mechanism

In FioRa+, the relay mechanism is on demand. Aiming to keep configuration costs and complexity to a minimum, the *on-demand* relay mechanism is designed to enable two-hop multicast firmware distributions from the LoRa gateway. As the name suggests, this mechanism is activated only when the server cannot reach all EH LoRa sensors using a particular DR. We detail two more mechanisms proposed to ensure firmware distribution coverage and to select multicast session configurations (e.g., DRs and relay sensors) later in Section 5. For now, let us assume that high DR is used by both the server/gateway and the relay sensors. All EH LoRa sensors are within the firmware distribution coverage after the relay mechanism is performed. The selected relay sensors are those that offer the shortest completion time of firmware update. In other words, these configurations maximize both the firmware distribution coverage and the amount of distributed firmware data. As high DR renders short airtime and large application payload, EH LoRa sensors can consume less energy while receiving more firmware data (i.e., data fragments). Consequently, EH LoRa sensors with low EH rates may complete the firmware update, provided that they have sufficient energy and face less packet losses.

4.2 Energy Management

Given the server may employ multiple multicast sessions to distribute a firmware image, Figure 5 examples how FioRa+ manages energy consumption during three multicast sessions to achieve energy neutrality. Here, we assume that the relay mechanism is enabled during all three sessions. Any of Sensors 1, 2, and 3 can possibly be selected as relay sensors. During a multicast session, one of them manifests the lowest amount of harvested energy in an EH LoRa network; such network consists of more than three EH LoRa sensors. Assuming also that the server/gateway and the relay sensors use the same DR for multicast firmware distributions, the energy consumed by an EH LoRa sensor for transmitting/relaying a data fragment of a particular length is thus higher than the energy consumed for receiving the data fragment of the same length. The power profile of a real-world

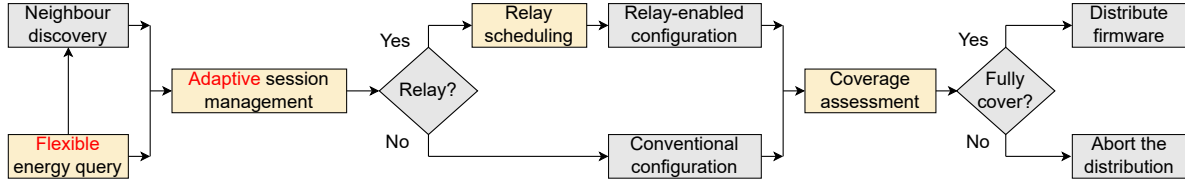


Fig. 6. Operational flow of FioRa+

EH LoRa sensor demonstrated in [18] validates this claim. For the sake of simplicity, we consider that sensors' harvested energy is known by the server. We elaborate how the server obtains this information later in Section 5.1. It is obvious that if the relay mechanism is not activated (i.e., the server uses a lower DR to reach all EH LoRa sensors), the server should perform multicast firmware distributions according to the maximum harvested energy that all EH LoRa sensors in the network (i.e., a multicast group) commonly have. This is to ensure that the energy neutrality is achievable. Otherwise, the server has to maximize the firmware data received by EH LoRa sensors while ensuring that the relay sensors can afford forwarding.

Let us take Session 1 in Figure 5 as an example. We assume that Sensors 1 and 3 are relay sensors and Sensor 2 features the minimum harvested energy. Here, the server cannot perform multicast firmware distributions according to Sensor 2's harvested energy (i.e., the maximum harvested energy that all EH LoRa sensors in the multicast group commonly have). This is because, if Sensors 1 and 3 spend such amount of harvested energy on receiving data fragments from the server, they will not have sufficient energy to relay all the received data fragments. Instead, the server needs to manage the energy consumed by receiving and relaying operations such that the amount of firmware data distributed in that session is maximized. When it comes to Session 3, Sensors 1 and 2 have sufficient energy to receive and relay firmware data according to Sensor 3's harvested energy. Consequently, the server fully utilizes Sensors 2 and 3's harvested energy in that session. In other words, Sensors 2 and 3 deplete their energy storage and have no energy residues after Session 3 expires. It is worth emphasizing that the energy neutrality is achievable as long as EH LoRa sensors face no power failures in the middle of multicast firmware distributions. Namely, the energy consumed during multicast firmware distributions is not greater than their energy availabilities. We detail how the server quantifies the amount of data fragments (i.e., firmware data) to be distributed in Section 5.3.

5 FioRa+

FioRa+ is an energy neutrality-aware multicast firmware distribution framework for EH LoRa networks. It is designed to minimize the completion time of firmware updates and the overhead of multicast firmware distributions. Figure 6 shows an operational flow of FioRa+. As an extension of our previous work called FioRa [19], FioRa+ is equipped with new/upgraded mechanisms. These mechanisms include flexible energy query, adaptive session management, time-slotted relay scheduling, and coverage assessment. Together, they enable the adaptive selection of the time and the session configurations that accelerate multicast firmware distributions at low cost. This section explains FioRa+'s system components in detail. After describing flexible energy query and one-hop neighbor discovery mechanisms, adaptive session management, time-slotted relay scheduling, and coverage assessment algorithms are elucidated.

5.1 Flexible Energy Information Acquisition

Unlike other wireless network technologies for IoT, LoRa networks support server-initiated multicast firmware distributions. By "server-initiated", it means a server decides when and how to distribute firmware data to a group of LoRa sensors. Consequently, LoRa sensors have less control over their energy consumption. To have LoRa

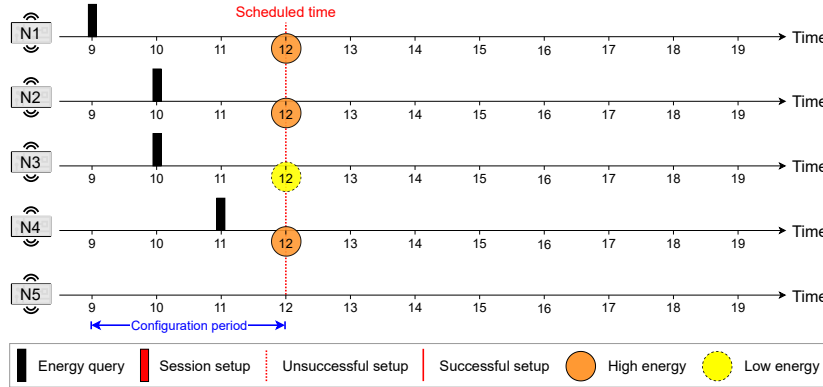


Fig. 7. Unsuccessful multicast session setup in FioRa

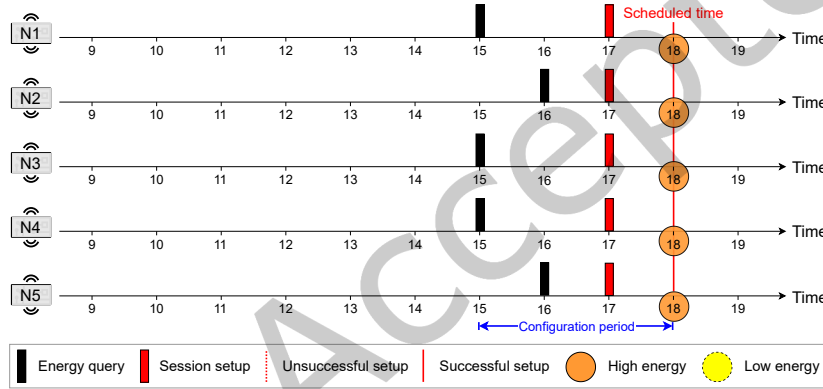


Fig. 8. Successful multicast session setup in FioRa

sensors receive firmware data simultaneously, the server needs to send a set of commands to each LoRa sensor in a unicast manner before multicast firmware distributions (see Section 2.1). In battery-powered LoRa networks, it may be adequate for the server to configure a single multicast session that accommodates the distribution of a firmware image. Namely, the server instructs LoRa sensors to enter a receive mode (e.g., Class C) for at most the time duration from the start of multicast firmware distributions until the entire firmware image is distributed. Nonetheless, such configuration may be inefficient for EH LoRa networks. Due to the heterogeneity of EH rates and the limited energy storage capacity, EH LoRa sensors may suffer power failures during multicast firmware distributions. The power failures cause EH LoRa sensors to miss the distributed firmware data (i.e., data fragments) when temporarily shutting down. Before configuring a multicast session, the server should thus acquire information about (future) energy availability from all EH LoRa sensors. With that, it configures the multicast session and performs multicast firmware distributions in an energy-neutral manner. This ensures that EH LoRa sensors receive firmware data simultaneously without encountering power failures. As a result, a firmware image may be fragmented and distributed across multiple multicast sessions. We detailed the design of energy neutrality-aware multicast firmware distributions in Section 2.2.

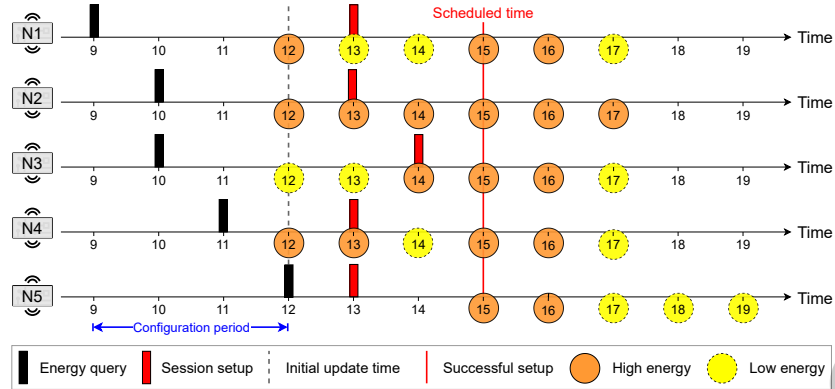


Fig. 9. A flexible energy query mechanism in FioRa+

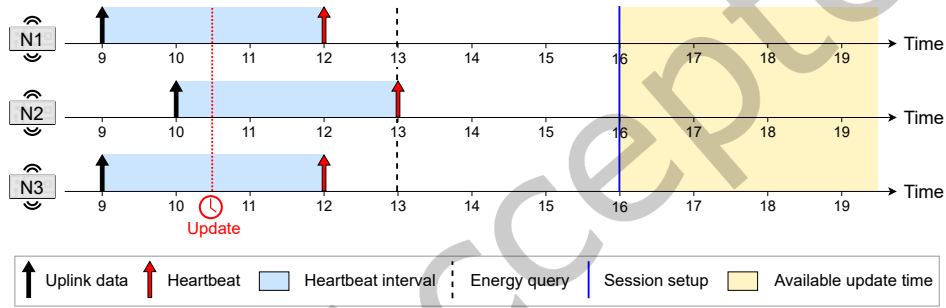


Fig. 10. The use of heartbeat interval for determining the initial time of firmware updates

A conventional means to obtain energy availability information is to have EH LoRa sensors regularly report harvested energy and/or battery readings to the server for further processing. With the emergence of embedded ML for predictive energy management [17, 18, 20], EH LoRa sensors start processing this information locally without transmitting it to the server. Because the multicast session needs to be configured beforehand, we thus rely on an existing embedded ML framework [17, 20] to acquire accurate information about future energy availability at low cost.

Precisely, FioRa+ allows the server to send an *energy query* to each EH LoRa sensor in a unicast manner before configuring a multicast session. The energy query contains a point in time (denoted as t) when multicast firmware distributions may be scheduled. Upon receiving the energy query, as opposed to FioRa where the predictive energy available at time t is reported (see Figures 7 and 8), a FioRa+ sensor reports an array of time-slotted predictive energy available from time t to $t+x$ (see Figure 9), where x is limited by the maximum size of the lowest DR's application payload (see Table 1). The time-slotted predictive energy is the amount of harvested energy possibly residing in EH LoRa sensor's energy storage at the beginning of a time-slot of equal length. Despite an increase in the size of energy-query response, the cost of reporting the array of predictive harvested energy can be trivial. In FioRa, as shown in Figure 7, all the successful energy queries/responses will be in vein if the server cannot determine session configurations and then configure a multicast session before the scheduled time of firmware distributions reaches. Consequently, the server has to reschedule the multicast firmware distributions

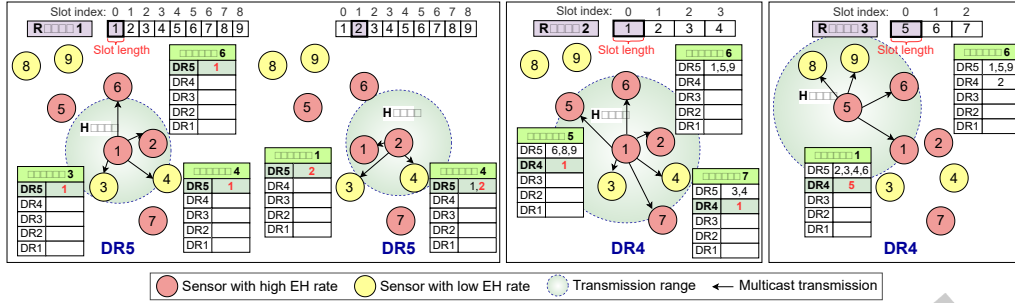


Fig. 11. Energy neutrality-aware multicast one-hop neighbor discovery

at the next time instant and send new energy queries to EH LoRa sensors (see Figure 8), increasing energy query overhead. Unlike FioRa, as illustrated in Figure 9, FioRa+ allows the server to reuse the time-slotted predictive energy successfully reported by EH LoRa sensors. If the (initial) scheduled time of firmware distributions reaches and energy-query responses from some EH LoRa sensors are still missing, the server will postpone the scheduled time by k time-slots. The number of k time-slots (i.e., the configuration period) must be sufficient for the server to successfully configure a multicast session before a new scheduled time reaches. With that, the server send energy queries containing the new (initial) scheduled time to EH LoRa sensors with no energy-query responses. The server will start over the energy query mechanism if, at any point in time, the remaining array of time-slotted predictive energy is shorter than $k + \gamma$. That is, it sends new energy queries containing a newly scheduled time to all EH LoRa sensors. Here, the number of extra time-slots (denoted by γ) is customizable. In Section 6, we consider $\gamma = k$, where k is set to 3. We explain the benefit of extra time-slots in Sections 5.2 and 5.3. Because of the flexible use of time-slotted predictive energy, we call FioRa+'s energy query mechanism a *flexible energy query* mechanism.

Aiming to synchronize the internal clock to network time, the EH LoRa sensor sends a command called DeviceTimeReq to the server periodically. The period of clock synchronization depends on the timing accuracy required by a system developer. Similar to [13], we allow the EH LoRa sensor to transmit a heartbeat message if it does not communicate with the server for an extended period of time. The server may use one of the two receive windows opened after such transmission to send the energy query to that EH LoRa sensor. As an example, Figure 10 illustrates the heartbeat intervals of three EH LoRa sensors denoted by N1, N2, and N3. By knowing roughly when such EH LoRa sensors communicate, the server can estimate the (initial) starting time of upcoming multicast firmware distributions. This time instant should be after the last McClassCSessionReq command (i.e., the last session setup command) sent to an EH LoRa sensor is responded.

According to Figure 2, the energy query can be sent at any time before McClassCSessionReq command is transmitted. If the entire firmware image is to be distributed across multiple multicast sessions, the energy query is sent per each multicast session until the firmware update is brought to an end. Once the energy availability information of all EH LoRa sensors in the multicast group is collected, the server executes an *adaptive session management* algorithm to select appropriate session configurations (see Section 5.3). With that, the server sends McClassCSessionReq command to each EH LoRa sensor after session configurations are determined. For each multicast session, a fresh transmission of McClassCSessionReq command is needed. Due to the simplicity of implementation [1], we consider multicast Class C sessions in this article.

5.2 One-hop Neighbor Discovery

As shown in Table 1, high DRs feature short airtime (i.e., low energy consumption) and large application payloads. Although these characteristics benefit FUOTA processes, such DRs, unfortunately, possess short transmission ranges. Considering this trade-off, FioRa+ leverages an on-demand relay mechanism to expand the coverage of multicast firmware distributions. However, relying on the relay mechanism alone is insufficient. Given EH LoRa sensors are geographically distributed, we need another mechanism to ensure that all EH LoRa sensors in the multicast group are covered by the relays. For this reason, we propose a mechanism called *one-hop neighbor discovery*. As illustrated in Figure 11, the one-hop neighbor discovery mechanism allows each EH LoRa sensor to announce its existence to all neighbors using different DRs. Tailored to EH LoRa networks, this mechanism operates in an energy-neutral manner. Namely, it prevents EH LoRa sensors from being unable to detect one-hop neighbors due to power failures.

We consider that each EH LoRa sensor is given a unique identifier (ID) at the time of deployment. To initiate the one-hop neighbor discovery mechanism, the server sends `McGroupSetupReq` command to all networked EH LoRa sensors in a unicast manner to create a multicast group. This multicast group is not necessarily similar to that of firmware update. By sharing the same multicast context (e.g., a multicast address), each EH LoRa sensor in the multicast group can broadcast a *hello message* containing its ID to neighbors. To have EH LoRa sensors listen for the hello message simultaneously, `McClassCSessionReq` command is sent to each EH LoRa sensor before the neighbor discovery takes place. The `McClassCSessionReq` command specifies, for example, the starting time of neighbor discovery, the session duration, and the DR for broadcasting/scanning the hello message. After overhearing the hello message, as colored in red, EH LoRa sensors store neighbor information (i.e., ID and corresponding DR) in a data structure. To introduce itself to one-hop neighbors located at different distances, the EH LoRa sensor broadcasts the hello message several times; each time a unique DR is employed. For example, Sensor 1 (i.e., sensor ID = 1) broadcasts hello messages using DR5 and DR4 in Figure 11. Upon receiving multiple hello messages from the same neighbor, an EH LoRa sensor (e.g., Sensor 6) only keeps the highest DR in the data structure and drops the others silently. Storing the highest DR is sufficient because it implies that the corresponding neighbor is also reachable using lower DR(s).

Given the heterogeneity of EH rates and the limited capacity of energy storage, EH LoRa sensors may not have sufficient energy to finish broadcasting and receiving hello messages in one multicast session. Namely, they cannot broadcast a hello message and then receive those from all other EH LoRa sensors using a configured DR. As a solution, we thus spread the neighbor discovery out across multiple multicast sessions in an energy-neutral manner. Before configuring a multicast session (i.e., prior to sending `McClassCSessionReq` command), the server sends energy queries (see Section 5.1) to all EH LoRa sensors to collect the energy availability information. Based on the arrays of predictive energy reported by EH LoRa sensors, the server selects the starting time of one-hop neighbor discovery (i.e., the starting time of multicast session) such that the number of exchanged hello messages is maximized. Among all the reported time-slotted predictive energy, the time-slot with the highest minimum predictive energy is selected. Subsequently, the server uses Equations 2 and 3 to calculate a session duration (SD) such that the energy consumption of neighbor discovery does not exceed the maximum energy sharing by all EH LoRa sensors (E_{max}). This energy consumption includes the energy consumed by an EH LoRa sensor for transmitting and receiving hello messages (N_{hello}). Indeed, the number of hello messages indicates how many EH LoRa sensors can broadcast their IDs in the multicast session. Essentially, the energy consumed during the transmission or reception of a hello message is the product of the transmitting (P_{tx}) or receiving (P_{rx}) power and the airtime (T_{hello}). Once the session duration is determined, the server sends `McClassCSessionReq` command to each EH LoRa sensor. Then, the server sends an instruction message to EH LoRa sensors that are selected to broadcast hello messages in that multicast session. This message contains a slot index. By knowing the slot indexes and the airtime of the hello message (i.e., slot length), the selected EH LoRa sensors can figure out when

Algorithm 1: Adaptive multicast session management

```

1  Input:  $NB, DR^{sl}, E^r, E^x, D^{mc}, D^{gw}, A$ 
2  Initialize:  $E^t, F, CF, TS \leftarrow 0, 0, 0, 0$ 
3  Initialize:  $maxB \leftarrow 0$ 
4  for  $a^t \in A$  do
5       $E^t, F \leftarrow a^t, 0$ 
6       $f \leftarrow \text{SelectConfig}(\text{GetConfig}())$ 
7      if  $maxB < \text{FindMaxB}(f)$  then
8           $CF, TS \leftarrow f, t$ 
9           $maxB \leftarrow \text{FindMaxB}(f)$ 
10 function  $\text{GetConfig}()$ :
11     for  $i \in DR^{sl}$  do
12          $S^{rl}, S^{dr}, S^{nb}, C \leftarrow 0, 0, 0, 0$ 
13          $D^{nb} \leftarrow D^{mc} - D_i^{gw}$ 
14         for  $j \in D^{nb}$  do
15              $L \leftarrow \emptyset$ 
16              $R \leftarrow \text{GetIDs}(NB_j) \cap D_i^{gw}$ 
17              $u, v, N_{pkt} \leftarrow \text{FindConfig}(i, j, R, S^{rl}, S^{dr})$ 
18              $C_j \leftarrow \{u, v, N_{pkt}\}$ 
19             if  $u \in S^{rl}$  then
20                 if  $v < S_u^{dr}$  then
21                      $L \cup \{u\}$ 
22             else
23                  $S^{rl} \cup \{u\}$ 
24                  $S_u^{dr} \leftarrow v$ 
25                  $S_u^{nb} \cup \{j\}$ 
26             AdjustConfig( $i, L, C, S^{rl}, S^{dr}, S^{nb}$ )
27         for  $j \in D_i^{gw}$  do
28             if  $j \in S^{rl}$  then
29                  $dr \leftarrow S_j^{dr}$ 
30                  $N_{pkt} \leftarrow \lfloor E_j^t / E_{dr}^{rl} \rfloor$ 
31             else
32                  $dr \leftarrow i$ 
33                  $N_{pkt} \leftarrow \lfloor E_j^t / E_i^{rx} \rfloor$ 
34              $C_j \leftarrow \{0, dr, N_{pkt}\}$ 
35          $F_i \leftarrow \{S^{rl}, S^{nb}, C\}$ 
36     return  $F$ 
37 function  $\text{FindConfig}(i, j, R, S^{rl}, S^{dr})$ :
38      $N_{pkt} \leftarrow 1$ 
39     for  $k \in R$  do
40          $dr \leftarrow \text{GetDR}(k, NB_j)$ 
41         if  $k \in S^{rl}$  then
42              $dr \leftarrow \min(dr, S_k^{dr})$ 
43         if  $dr \geq i$  then
44              $p \leftarrow \min(\lfloor E_k^t / E_{dr}^{rl} \rfloor, \lfloor E_j^t / E_{dr}^{rx} \rfloor)$ 
45             if  $N_{pkt} < p$  then
46                  $N_{pkt} \leftarrow p$ 
47              $u, v \leftarrow k, dr$ 
48     return  $u, v, N_{pkt}$ 
49 function  $\text{AdjustConfig}(i, L, C, S^{rl}, S^{dr}, S^{nb})$ :
50     while  $L \neq \emptyset$  do
51          $m \leftarrow \text{GetRelay}(L)$ 
52          $del \leftarrow \emptyset$ 
53         for  $n \in S_m^{nb}$  do
54              $R \leftarrow \text{GetIDs}(NB_n) \cap D_i^{gw}$ 
55              $x, y, N_{pkt} \leftarrow \text{FindConfig}(i, n, R, S^{rl}, S^{dr})$ 
56             if  $x \neq m$  then
57                  $del \cup \{n\}$ 
58                 if  $x \notin S^{rl}$  then
59                      $S^{rl} \cup \{x\}$ 
60                 else
61                     if  $x \notin L$  then
62                          $L \cup \{x\}$ 
63                  $S_x^{dr} \leftarrow y$ 
64                  $S_x^{nb} \cup \{n\}$ 
65                  $C_n \leftarrow \{x, y, N_{pkt}\}$ 
66          $S_m^{nb} \leftarrow S_m^{nb} - del$ 
67         if  $S_m^{nb} = \emptyset$  then
68              $S^{rl} \leftarrow S^{rl} - \{m\}$ 
69              $S_m^{dr} \leftarrow \emptyset$ 
70              $L \leftarrow L - \{m\}$ 
71     return
    
```

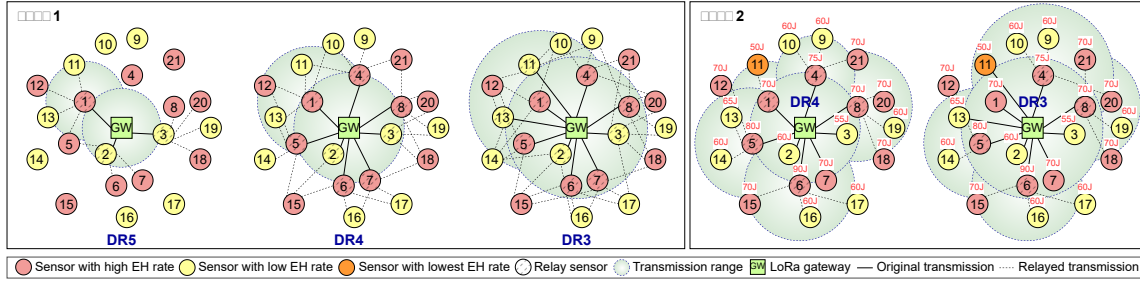


Fig. 12. The selection of data rates and EH LoRa sensors for relaying data fragments

to start broadcasting their IDs. Whether or not being selected, all EH LoRa sensors (if they are not broadcasting their hello messages) listen for hello messages from neighbors until the configured session duration expires. To have EH LoRa sensors broadcast hello messages using different DRs, the server configures multicast sessions using one DR after another.

$$N_{hello} = \max\left(\left\lfloor \frac{E_{max} - T_{hello} \times (P_{tx} - P_{rx})}{T_{hello} \times P_{rx}} \right\rfloor, 0\right) \quad (2)$$

$$SD = T_{hello} \times N_{hello} \quad (3)$$

By performing the neighbor discovery mechanism in an energy-neutral manner, it may takes several multicast sessions (i.e., rounds) to bring such mechanism to an end. Once the neighbor discovery mechanism is over, the server requests each EH LoRa sensor to report all the neighbor information (i.e., pairs of ID and DR). This information is then used by the adaptive session management algorithm (i.e., Algorithm 1) to ensure the firmware distribution coverage and select the appropriate session configurations (see Section 5.3). By design, the one-hop neighbor discovery mechanism is executed once during network deployment. Thereafter, it can be activated whenever the positions of EH LoRa sensors change and/or when new EH LoRa sensors are added to the existing networks. Consequently, the cost of executing this mechanism is deemed negligible because it occurs occasionally.

5.3 Adaptive Multicast Session Management

This section explains how FioRa+ adopts the mechanisms demonstrated in previous sections for adaptive multicast session management. From the server's perspective, the goals of multicast session management are threefold. The first is to activate the relay mechanism if necessary. The second is to ensure that firmware distributions cover the multicast group of EH LoRa sensors. The last is to determine the distribution timing and the session configurations that maximize the amount of firmware data distributed in an energy-neutral manner.

Over-the-air firmware update is an energy-intensive task [27]. In EH LoRa networks, the server should thus employ an energy-efficient DR (i.e., high DR) for multicast firmware distributions. To find out such DR, FioRa+ allows the server to keep track of the latest DRs used by EH LoRa sensors during uplink communications. In addition, the server stores EH LoRa sensors' one-hop neighbor information in a data structure (NB). As described in Section 5.2, this information is obtained by sending requests to EH LoRa sensors after completing the neighbor discovery mechanism. Once a multicast session for firmware distributions is to be configured, the server sends energy queries to all EH LoRa sensors in the multicast group (see Section 5.1). After their responses are recorded in a data structure, the server equalizes the length of the arrays of predictive energy reported by EH LoRa sensors (A), so as to ensure that they cover the same time-slots. As previously discussed in Section 5.1, the length of the arrays (i.e., the number of available time-slots) should be greater than or equal to the extra

time-slots (i.e., γ). To select the timing (TS) and configurations (CF) for multicast firmware distributions, the server then triggers the adaptive multicast session management algorithm (i.e., Algorithm 1). Given all EH LoRa sensors' predictive energy available at the time-slot t (denoted by E^t), as shown in Figure 12, the multicast session management algorithm is executed in two steps. First, the server determines which DRs ensure the coverage of firmware distributions (DR^{sl}). With that determination, it selects the session configurations (and relay sensors) that maximize data reception rate in an energy-neutral manner.

Precisely, to estimate the coverage of a particular DR, the server checks whether EH LoRa sensors can hear firmware data from a LoRa gateway and/or relay sensors using that DR. As explained in Section 4.1, the relay sensors are considered only when there remain some EH LoRa sensors not covered by the LoRa gateway. To determine if EH LoRa sensors are reachable by the LoRa gateway, the server searches its data structure for EH LoRa sensors' latest DRs. The EH LoRa sensors are considered in the LoRa gateway's vicinity (D^{gw}) if their latest DRs are not lower than the DR under consideration. As for EH LoRa sensors covered by relay sensors, the server uses the one-hop neighbor information stored in its data structure to determine. Here, the EH LoRa sensors are deemed reachable by the relay sensors if they overhear hello messages from those sensors using that or higher DRs.

After determining which DRs are usable (DR^{sl}), the server executes `GetConfig` module of Algorithm 1 to discover possible configurations (F) for firmware update for each time-slot. The configuration of each DR ($F_i; i \in DR^{sl}$) includes pairs of relay sensors (S^{rl}) and their neighbors (S^{nb}) as well as session configurations (C). The session configurations involve the DRs assigned to EH LoRa sensors for receiving (and relaying) data fragments. Among all the possible time-slots, through `SelectConfig` and `FindMaxB` modules, the server selects the time-slot with configurations that maximize the firmware data ($maxB$) received by the multicast group of EH LoRa sensors (D^{mc}) for multicast firmware distributions. With that, it configures a multicast session accordingly. The amount of firmware data is the product of the number of received data fragments (N_{pkt}) and the size of the application payload in bytes. Depending on the role of an EH LoRa sensor, the number of data fragments received and possibly relayed by the sensor is the ratio of its predictive harvested energy available at the time t (denoted by E^t) and the total energy consumed for receiving (E^{rx}) or relaying (E^{rl}) a data fragment. Here, E^{rl} concerns the energy consumed by the relay sensor for both receiving and relaying the data fragment, whereby different DRs may be used.

While listening to the LoRa gateway using a particular DR, the relay sensors can employ the same or higher DRs to relay the received data fragments to their neighbors. To select such DRs, the server leverages one-hop neighbor information to pair *out-of-range* sensors (i.e., EH LoRa sensors not covered by the LoRa gateway (D^{nb})) with the relay sensors using the highest possible DRs. Through `FindConfig` and `AdjustConfig` modules, the server keeps finding pairs of relay sensors and neighbors until they converge. Namely, each out-of-range sensor is assigned to a relay sensor that offers the maximum amount of distributed firmware data (see the right of Figure 12). To ensure the simultaneous and consistent reception of firmware data, the server employs the maximum number of data fragments received and possibly relayed by all EH LoRa sensors to calculate session duration. Basically, the session duration is the product of such number of data fragments and the time duration (i.e., airtime) for transmitting a data fragment using the DR selected by the server. If the relay mechanism is enabled, such airtime will be doubled to accommodate the relay. Whether or not the relay mechanism is activated, the airtime will be equal to the sum of T_{on} and T_{off} under DC regulations. To convey the session configuration to an EH LoRa sensor, `McClassCSessionReq` command and possibly a notification message are sent in a unicast manner to configure the multicast session and relay contexts. Essentially, the server uses the notification message to activate a relay mechanism and program a channel, DR, and time-slot for relaying firmware data on a relay sensor. We explain the relay slot assignment in Section 5.4.

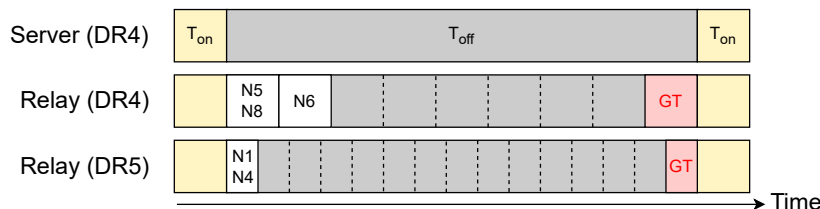


Fig. 13. The assignment of different time-slots to relay nodes given a 10% duty cycle

5.4 Time-slotted Relay Scheduling

Thanks to the variety of DRs possibly used by relay sensors, the session management algorithm (see Section 5.3) alleviates signal collisions to some extent. To take collision avoidance to the next level, the relay sensors may adopt an advanced carrier-sense multiple access technique for LoRa [14] if available. Otherwise, the server may instruct the relay sensors using the same DRs to forward data fragments at different points in time and/or in different channels. If DC restrictions are applied and/or the limited number of channels is of concern, adopting the former approach (i.e., time-slotted assignment) alone would be more efficient.

As shown in Figure 13, the server may divide the time duration (T_{off}) that the LoRa gateway cannot access a channel because of DC restrictions into different time slots. The length of each time slot is equal to the airtime (T_{on}) of a data fragment, which depends on the DR in use. Considering different DRs employed by the relay sensors, the server arranges time slots for each DR separately. Due to the orthogonality of SFs, the relay sensors using different DRs can forward data fragments concurrently without causing signal collisions. Aiming to check signal collisions caused by relay sensors using the same DRs, the server uses one-hop neighbor information collected from EH LoRa sensors (see Section 5.2). The signal collisions occur whenever the relay sensors can reach out-of-range sensors assigned to other sensors using the same DRs. To speed up firmware data distributions, the relay sensors using the same DRs are assigned to the same/earliest time-slots if they cause no collisions. If signal collisions are unavoidable, the server assigns the relay sensors to the time-slots causing the fewest signal collisions.

Through the notification messages discussed in Section 5.3, the server assigns time slots (i.e., slot indexes) to the relay sensors. By knowing the slot index and the airtime of the data fragment, each relay sensor figures out when to forward the received data fragment. Essentially, the time-slotted relay scheduling algorithm should allow the last relay sensors to complete forwarding a bit earlier before the LoRa gateway distributes the next data fragment. Namely, they cannot relay firmware data in guarded time-slots (GT).

5.5 Coverage Assessment

As exhibited in Figure 6, FioRa+ employs a coverage assessment algorithm to ensure that all EH LoRa sensors can receive firmware data at once. Through this algorithm, the server checks whether all EH LoRa sensors are successfully configured a multicast session. If such configuration is unsuccessful, the server aborts the scheduled distributions of firmware data. If the relay mechanism is enabled, the server further determines whether all notification messages sent to relay sensors are responded. If some relay sensors fail to response to such messages, the server estimates if their assigned out-of-range sensors can be covered by other relay sensors using the same DRs. During such estimation, the server again uses one-hop neighbor information collected from EH LoRa sensors (see Section 5.2). The server aborts the scheduled multicast firmware distributions if uncovered out-of-range sensors exist. Namely, there remain one or more out-of-range sensors uncovered by successfully-responding relay sensors using such configured DRs. Otherwise, the server proceeds with multicast firmware distributions

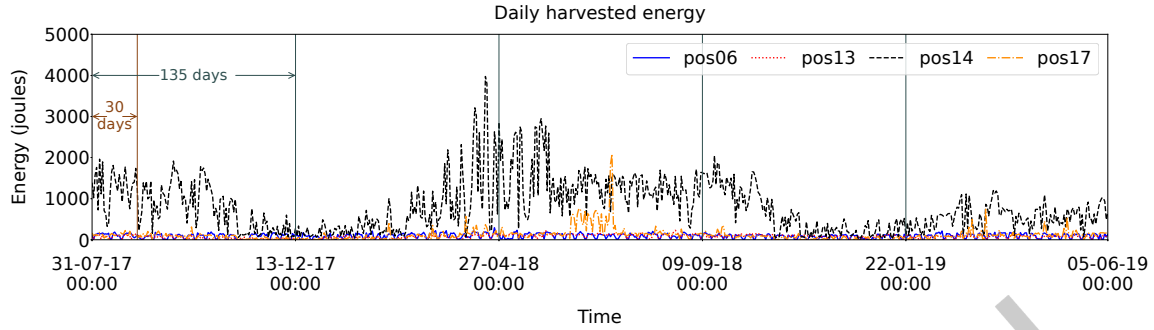


Fig. 14. Solar energy daily harvested by four sensors over 675 days [24, 25]

at the scheduled time. Indeed, aborting multicast firmware distributions is essential to keep traffic overhead to a minimum. To conserve energy storage, FioRa+ allows EH LoRa sensors to convert to a low-power mode whenever they do not detect any firmware data for a certain amount of time.

6 Simulation and Results

To the best of our knowledge, FioRa+ is the first framework to address multicast firmware distributions in EH LoRa networks. For that reason, this section validates its efficiency against the traditional solution in LoRaWAN [9]. To showcase the efficiency of new/upgraded mechanisms, we further compare the performance of FioRa+ against FioRa [19] in this article.

6.1 Simulation Environment

6.1.1 Simulation Setup. As a proof of concept, we leveraged a publicly-available indoor solar harvesting dataset [25] to model our trace-driven simulations. Given battery voltage (V_{bat}) and current (I_{bat}) measurements contained in that dataset, we calculated the energy that four different EH nodes (sensors, for short) could concurrently harvest every hour. These sensors called pos06, pos13, pos14, and pos17 were deployed in different monitoring **positions** and harvested different amount of energy. As exhibited in Figure 14, Sensor pos14 harvested more energy than its counterparts. This is because it was located in a position where exposure to direct sunlight was expected. As opposed to other sensors, Sensor pos13, on average, harvested the least amount of energy because it was located in a relatively challenging position. More details of the deployment locations and the EH conditions of these real-world EH sensors can be found in [24].

To enable accurate trace-driven simulations, we modeled EH LoRa sensors (e.g., energy profiles) according to an off-the-shelf LoRa device. This LoRa device comprises an Arduino Portenta H7 Lite microcontroller and an add-on board called Arduino Portenta Vision Shield LoRa[®]. We considered each EH LoRa sensor to be powered by a 3.7V 30mAh LiPo rechargeable battery. To enable fruitful operations on our EH LoRa sensors, we modified the above EH dataset to support 5W solar panels (see Figure 14). Starting with 10% residual energy in energy storage, EH LoRa sensors periodically read hourly harvested energy from the dataset to estimate the amount of harvested energy stored in energy storage for energy consumption. Following a LoRaWAN regional parameter specification [8], we considered that a LoRa gateway can reach all EH LoRa sensors using DR2. In case DC restrictions are enabled, we consider an EH LoRa sensor and a LoRa gateway are subject to 1% (in an uplink direction) and 10% (in a downlink direction) DCs, respectively. In other words, the DCs restrictions follow EU868 frequency plan [8].

In our simulations, we considered that EH LoRa sensors regularly sent 10-byte sensing data to a server every hour. After clock synchronization, multicast group setup, fragmentation session setup, and multicast session configuration/management mechanisms were performed, firmware distributions were activated periodically during the daytime (i.e., at 8 am, 1 pm, and 6 pm) until all EH LoRa sensors received firmware images in full. No sensing data transmissions ever occurred whenever multicast firmware distributions took place. Given the limited amount of energy harvested by EH LoRa sensors, several multicast sessions might be required to transmit varied sizes of firmware images. For each multicast session, control messages related to session configuration and/or session management were transmitted separately in a unicast manner before firmware distributions took place. Regardless of the maximum amount of firmware data remaining to be received by EH LoRa sensors, we allowed the server to continuously transmit firmware data for one hour at most (i.e., a multicast session duration was limited to one hour). In other words, the session duration could be shorter than one hour, provided that a few firmware data remaining to be distributed. This time limitation ensures that EH LoRa sensors could harvest some energy between any two consecutive multicast sessions and made progress. Throughout the simulations, packet losses occurred when EH LoRa sensors did not have sufficient energy to communicate. If DC restrictions were enabled, we also considered signal collisions caused by relay sensors using the same DRs. In the face of unsuccessful session configuration and/or management due to power failures, the server would wait until all sensors completed such mechanisms before proceeding with the next stage. As a result, the server might be unable to distribute firmware data regularly at the aforementioned times of the day. For the simplicity of implementation, we allowed the server to (re)schedule a new multicast session whenever the (current) scheduled time reached or the previous multicast session expired.

We divided the above dataset into five smaller datasets containing time-series data of 135 days each. Considering each divided dataset contained four different sensors, we employed them to model 20 EH LoRa sensors (i.e., Sensors 1–20) deployed in different EH conditions as exemplified in Figure 12. These EH LoRa sensors joined a network as Class A devices and converted to Class C devices during a configured multicast session. To investigate the impact of DC restrictions on the efficiency of multicast firmware distributions in EH LoRa networks, we conducted two experiments whereby DC restrictions are disabled and enabled, respectively. In the first experiment, we estimated the performance of different frameworks when distributing the varied sizes of firmware images in the range of 10–100 kB. Similar to our previous work [19], we considered packet losses occurring only when EH LoRa sensors had inadequate energy for communications. As the activation of DC restrictions and the limited size of EH datasets prevent the multicast firmware distributions of some baselines from success, we limited the size of firmware images distributed in the second experiment to 20 kB. Specifically, we made sure that all frameworks considered in the second experiment could complete multicast firmware distributions within 105 days. Thanks to the activation of relay signal collisions in a 10% duty-cycled downlink channel, we excluded the performance evaluation of FioRa from the second experiment. For a fair comparison, we mainly validated the efficiency of FioRa+ (i.e., FioRa+ was equipped with the time-slotted relay scheduling mechanism to avoid relay signal collisions) against other baselines whereby relay signal collisions did not exist. From out-of-range sensor's perspective, relay signal collisions occurred whenever there were data fragments concurrently transmitted by different relay sensors in its vicinity using the same configured DR. To point out the outstanding performance of FioRa+, we further conducted the second experiment separately. In other words, we investigated how the flexible energy-query mechanism affects the efficiency of FioRa+ in the third experiment. In this experiment, we allowed the server to distribute the varied sizes of firmware images in the range of 10–90 kB.

Each simulation terminated when all EH LoRa sensors reported that they had sufficiently received firmware data for firmware image reconstructions. If a relay mechanism was enabled, we instructed a relay sensor to keep receiving and forwarding firmware data even though it has completely received them. As for an out-of-range sensor, we enabled it to receive firmware data from any relay sensors, provided that they used the same configured DR and were in its vicinity. If the (flexible) energy query mechanism was enabled, we allowed the server to

launch such mechanism 3 hours ($k = 3$) prior to the (initial) scheduled time. Aiming to minimize the traffic overhead, we programmed the server not to send any energy query if the current time-slot is one hour/time-slot away from the (initial) scheduled time. This is to ensure that all EH LoRa sensors could successfully configure a multicast session (and possibly relay contexts) before firmware distributions started. If firmware distributions were aborted, we enabled EH LoRa sensors successfully configuring a multicast session to scan a downlink channel for one minute before resuming their application tasks. After the (original) firmware distributions were completed, the server communicated a command to each EH LoRa sensor individually, requesting its FUOTA status. If the additional firmware distributions were needed, the amount of firmware data distributed was limited to the maximum amount of firmware data (inclusive of padding) remaining to be received by any EH LoRa sensor before successfully reconstructing a firmware image. Furthermore, to facilitate neighbor discovery and emulate an actual EH LoRa network, we allowed the server to start multicast firmware distribution processes (i.e., to initiate clock synchronization) after 30 days of deployment.

6.1.2 Baselines. We compared the performance of FioRa+ against different baselines. LoRaWAN is a traditional solution [9] where the server has no knowledge of EH LoRa sensors' harvested energy during session configuration. Consequently, the server distributed firmware data to EH LoRa sensors according to the maximum number of firmware data remaining to be received, provided that the airtime was not over an hour. ENO (abbreviated from energy-neutral operation) is a variant of FioRa where the server only distributed firmware data in an energy-neutral manner without enabling a relay mechanism. Unlike ENO, ENO+ is a variant of FioRa+; it is equipped with a flexible energy query mechanism. To further investigate the impact of the flexible energy query mechanism, we divided FioRa+ into two variants whereby the flexible energy query mechanism is enabled and disabled (NF), respectively. Here, NF stands for "no flexibility". By disabling the flexible energy query mechanism, FioRa+ server is unable to reuse the arrays of time-slotted predictive energy reported by any EH LoRa sensors. Unlike FioRa (EML), FioRa+ (EML), and FioRa+ (EML-NF) where embedded ML (EML) proposed in [17, 20] was employed for EH predictions, LoRaWAN, FioRa (OPT), FioRa+ (OPT), FioRa+ (OPT-NF), ENO, and ENO+ used the actual future harvested energy to illustrate optimal (OPT) performance. Using actual energy availability leads to flawless energy prediction for session management.

6.1.3 Performance Metrics. We used the following performance metrics to evaluate the efficiency of FioRa+ against the baselines.

- Success rate is the ratio of the number of EH LoRa sensors that successfully receive original firmware images to the total number of sensors in the network.
- Completion time is the time duration from the start of multicast firmware distributions until all EH LoRa sensors receive sufficient data for reconstructing firmware images.
- Number of multicast session is the number of multicast session(s) required until all EH LoRa sensors successfully receive firmware images.
- Distribution duration is the total airtime of multicast firmware distributions.
- Distribution overhead is the total bytes of firmware data distributed by the server until all EH LoRa sensors are able to reconstruct their firmware images.
- Traffic overhead is the total bytes of firmware data and control messages exchanged between the server and EH LoRa sensors, including packet headers.
- Energy query overhead is the number of energy queries with unique IDs transmitted by the server.
- Unsuccessful energy requests are the number of energy queries with unique IDs transmitted by the server, but fail to have multicast firmware distributions take place.
- The number of colliding sensors is the number of out-of-range sensors concurrently hearing data fragments from multiple relay sensors using the same DRs.

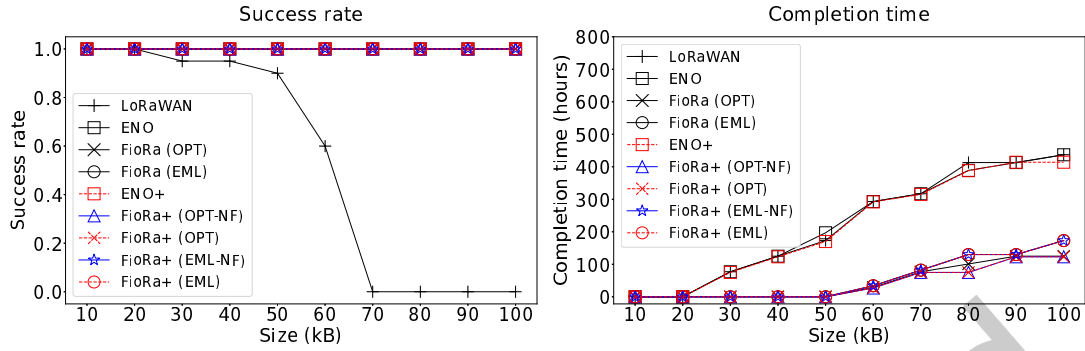


Fig. 15. The success rate and the completion time of multicast firmware distributions in EH LoRa networks with no DCs regulated

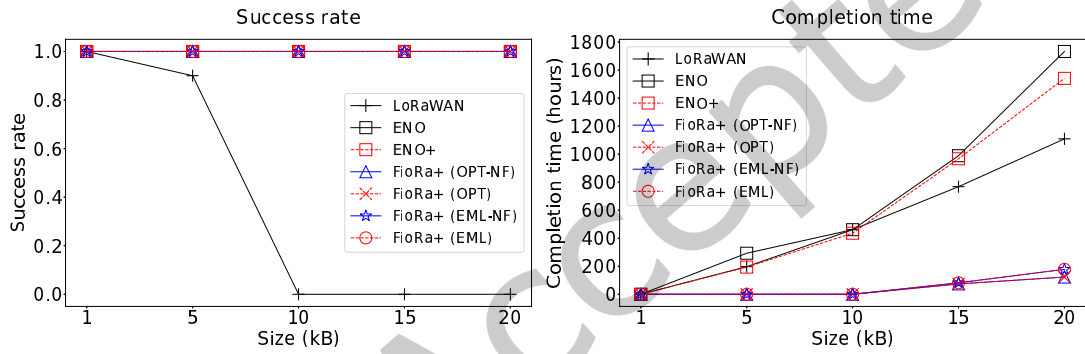


Fig. 16. The success rate and the completion time of multicast firmware distributions in EH LoRa networks with DCs regulated

- The number of collided bytes is the average number of unsuccessfully received firmware data per colliding sensor.
- The minimum number of relay sensors is the smallest amount of EH LoRa sensors selected as relay sensors per simulation.
- The maximum number of relay sensors is the greatest amount of EH LoRa sensors selected as relay sensors per simulation.

6.2 Results

6.2.1 Success Rate. As shown on the left of Figures 15, 16, and 17, all frameworks enabling energy neutrality-aware multicast firmware distributions achieve the highest success rate. This is because, by following the concept of energy-neutral operation, the server gradually distributed firmware data to all EH LoRa sensors according to their energy availability. Without facing power failures, all EH LoRa sensors implementing ENO, FioRa, and FioRa+ variants could simultaneously receive original firmware data distributed by the server. As a result, they eliminated the need for the server to trigger additional multicast firmware distributions. Compared to previous frameworks, LoRaWAN renders the lowest success rate. Without the knowledge of energy availability, LoRaWAN server kept distributing firmware data to EH LoRa sensors even though they ran out of energy and temporarily

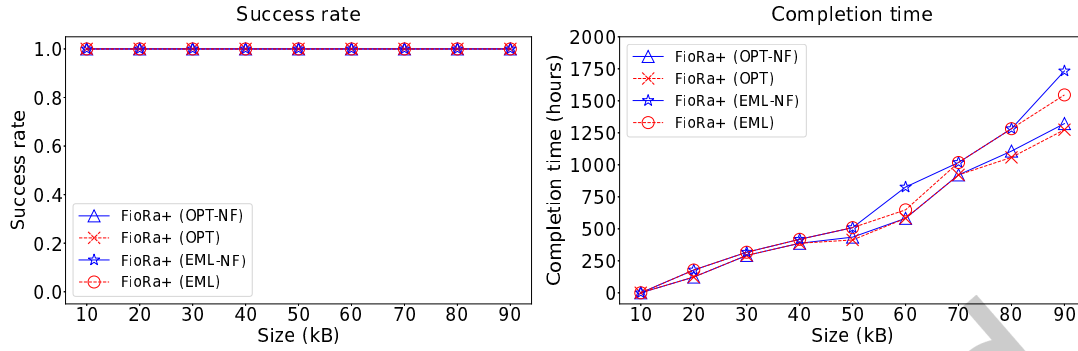


Fig. 17. The success rate and the completion time of multicast firmware distributions performed using different FioRa+ variants in duty-cycled EH LoRa networks

shut down. Consequently, the success rate of LoRaWAN drops sharply (with the lowest success rate equals to 0) when distributing the large sizes of firmware images.

When DCs were regulated (see the left of Figures 16 and 17), EH LoRa sensors likely encountered power failures during (original) multicast firmware distributions. Namely, the energy consumed during each T_{off} duration (i.e., EH LoRa sensors waited for T_{off} duration before receiving the next data fragment from the server/gateway) accelerated the depletion of EH LoRa sensors' energy storage. As a function of airtime (i.e., T_{on}), T_{off} duration lasts longer (i.e., EH LoRa sensors waste more energy on scanning a downlink channel) when the server uses low DR for multicast firmware distributions. Given EH LoRa sensors unlikely sustained an hour-long multicast session (i.e., the maximum session duration used in our experiments), performing multicast firmware distributions in an energy-neutral manner thus became even more beneficial. Compared with LoRaWAN, energy neutrality-aware frameworks like ENO, ENO+, FioRa, and FioRa+ thus consistently manifest the highest success rates (i.e., their success rates equal to 1) in all the experiments.

6.2.2 Completion Time. As exhibited on the right of Figures 15 and 16, EH LoRa sensors implementing FioRa+ and FioRa variants complete receiving firmware images faster than ENO variants and LoRaWAN. Additionally, FioRa+ also renders slightly shorter completion time than FioRa. For example, given a firmware image of 80 kB (see the right of Figure 15), FioRa+ (OPT) server completed multicast firmware distributions approximately 26 hours earlier than FioRa (OPT) server. Because of the use of higher DRs, FioRa+ and FioRa sensors consumed less energy to receive more firmware data from the server or the relay sensors. Consequently, despite having the limited amounts of harvested energy, low EH-rate sensors could still make sufficient progress in an energy-neutral manner when running FioRa+ and FioRa variants. Compared with optimal FioRa+ and FioRa variants, FioRa+ and FioRa variants using EML render slightly higher completion time due to the underestimation of predictive energy availability. Through the flexible energy query mechanism, the right of Figure 17 suggests that, apart from maximizing data reception rates, FioRa+ (OPT) and FioRa+ (EML) facilitate the success of energy queries. As soon as the (flexible) energy query mechanism was completed, the server could immediately proceed with the multicast session setup. Accordingly, FioRa+ (OPT) and FioRa+ (EML) took shorter time to complete the distributions of large firmware images than FioRa+ (OPT-NF) and FioRa+ (EML-NF). As illustrated on the right of Figures 15 and 16, the completion times of ENO+ and ENO also align with this finding. Being equipped with the flexible energy query mechanism, ENO+ completed multicast firmware distributions faster than ENO.

Thanks to the use of low DR, ENO, ENO+, and LoRaWAN manifest comparable completion times when DCs are not regulated. However, when DC restrictions are present (see the right of Figure 16), LoRaWAN tends to complete

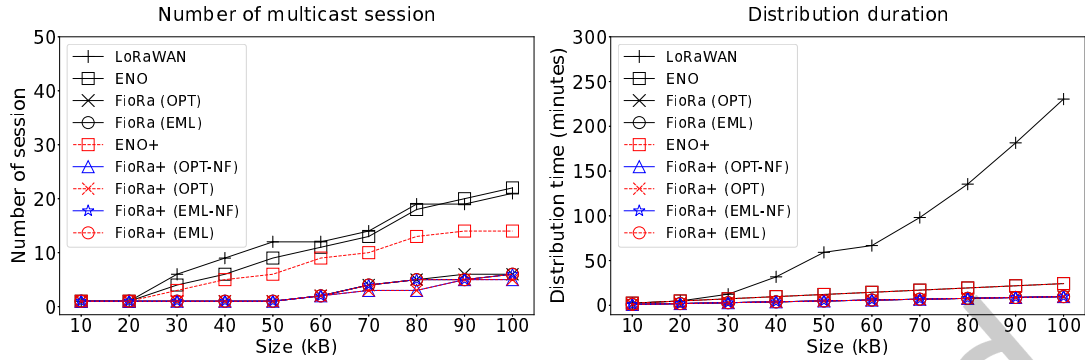


Fig. 18. The number of multicast session and the distribution duration when performing multicast firmware distributions in EH LoRa networks with no DCs regulated

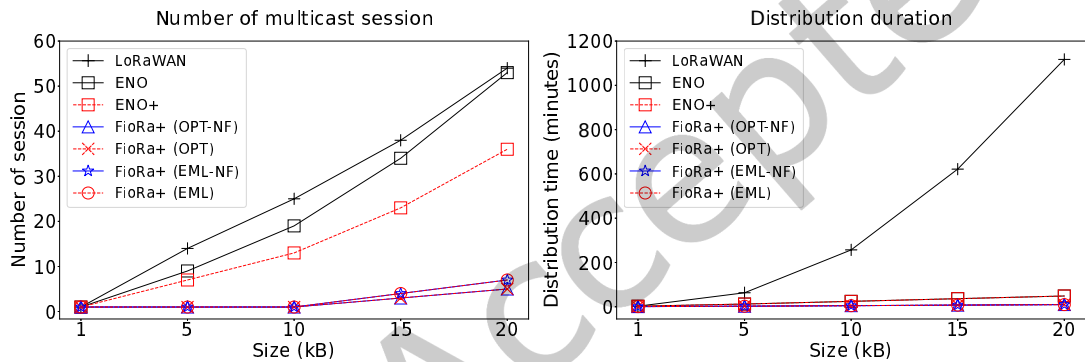


Fig. 19. The number of multicast session and the distribution duration when performing multicast firmware distributions in EH LoRa networks with DCs regulated

the distributions of firmware images faster than ENO and ENO+. Because the significant amounts of energy were depleted during gateway's off times (i.e., T_{off} lasts longer when using lower DR), ENO and ENO+ sensors took longer time to recover from power failures under challenging EH conditions. Being unable to configure multicast sessions, ENO and ENO+ thus completed multicast firmware distributions later than LoRaWAN.

6.2.3 Number of Multicast Session. As illustrated on the left of Figures 18 and 19, FioRa+ and FioRa variants employ the smaller number of multicast sessions to complete distributing the varied sizes of firmware images compared to LoRaWAN and ENO variants. Thanks to the use of high DRs, FioRa+ and FioRa variants could distribute more firmware data per LoRa packet, while consuming harvested energy efficiently. Being able to maximize firmware data distributed per each multicast session, the server thus required a few multicast sessions to distribute firmware images. When the size of firmware images were relatively small, FioRa+ and FioRa variants required only one multicast session to complete the distributions of such firmware images. However, when the size of firmware images grew and/or when DCs were regulated, FioRa+ and FioRa variants might require more multicast sessions. Nonetheless, such increasing number of multicast sessions were still minor when compared to LoRaWAN and ENO variants. Being equipped with the flexible energy query mechanism, the left of Figures 18, 19 and 20 further reveals that FioRa+ (OPT), FioRa+ (EML), and ENO+ require less multicast sessions than their

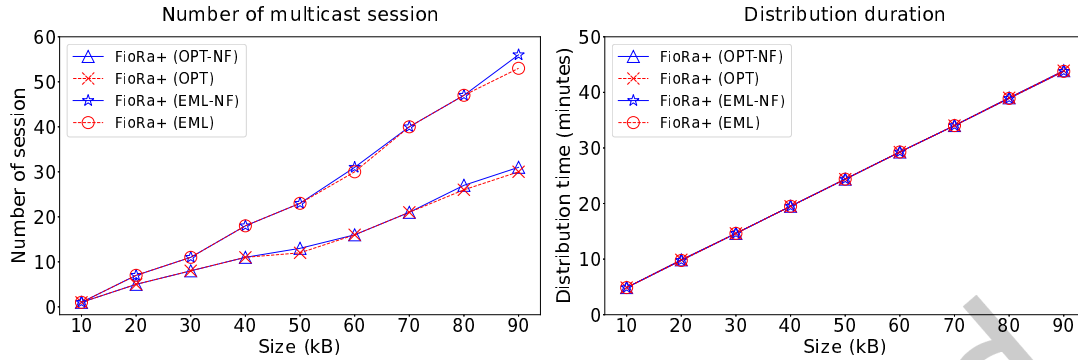


Fig. 20. The number of multicast session and the distribution duration when performing multicast firmware distributions using different FioRa+ variants in duty-cycled EH LoRa networks

counterparts. Through the arrays of time-slotted predictive energy reported by EH LoRa sensors, the server could adaptively select the distribution timings and the session configurations that maximized firmware data distributed. Such adaptive selection enabled multicast firmware distributions to take place at the time other than the initial time requested by the server, provided that more firmware data could be distributed. Compared with FioRa (OPT), the left of Figure 18 also shows that FioRa+ (OPT) manifests up to $1.7\times$ fewer number of multicast sessions when distributing the firmware images of more than 60 kB.

Owing to the use of low DR, ENO variants and LoRaWAN employed the significant numbers of multicast sessions to complete the distributions of firmware images. When DCs were regulated, their performances were drastically limited by the length of EH datasets in use. Despite rendering longer completion time (see the second diagram of Figure 16), the left of Figure 19 exhibits that ENO and ENO+ require the fewer number of multicast sessions than LoRaWAN. Complying with energy-neutral operation, ENO, ENO+, FioRa, and FioRa+ thus performed multicast firmware distributions only when the simultaneous reception of firmware data was ascertained.

6.2.4 Distribution Duration. As shown on the right of Figures 18 and 19, due to the use of high DRs during multicast firmware distributions, FioRa+ and FioRa variants offer the shortest distribution duration/time (i.e., the smallest total airtime) compared to other frameworks. Unlike the completion time (see Section 6.2.2), the distribution duration is not influenced by the update times, which were 8 am, 1 pm, and 6 pm in our simulations (see Section 6.1.1). Given a LoRa gateway is usually half-duplex (i.e., it stops receiving any LoRa packets while transmitting) [29, 34], the short distribution durations of FioRa+ and FioRa variants enable the server/gateway to complete multicast firmware distributions and return to normal operations rapidly, reducing packet losses in an uplink direction. Through energy neutrality-aware multicast firmware distributions, the right of Figures 18, 19, and 20 further reveals that the distribution durations of ENO, ENO+, FioRa, and FioRa+ increase linearly as the size of firmware images increases.

Despite using low DR, ENO and ENO+ render relatively short distribution durations. Following the concept of energy-neutral operation, they were as well free from the additional distributions of firmware data. Regardless of long completion time (see the right of Figures 15 and 16), their actual airtime was fairly short, yet longer than FioRa and FioRa+. Unlike all other frameworks, LoRaWAN manifests the longest distribution duration. This is because, without the knowledge of energy availability, LoRaWAN keeps the number of firmware distributions to the maximum per multicast session. Performing additional multicast firmware distributions using low DR further increases the distribution duration of LoRaWAN. Compared with LoRaWAN (see the right of Figures 18 and 19),

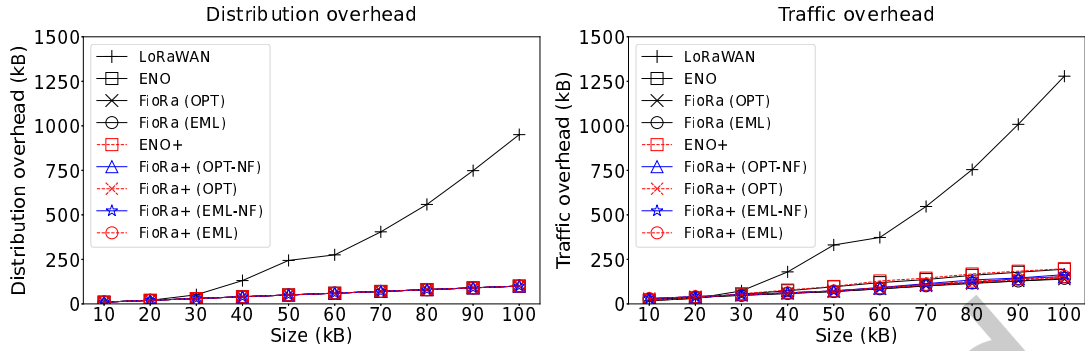


Fig. 21. The distribution overhead and the traffic overhead when performing multicast firmware distributions in EH LoRa networks with no DCs regulated

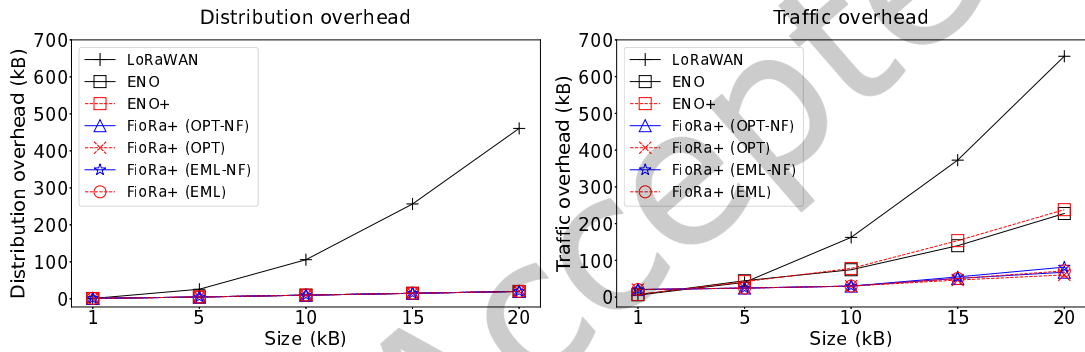


Fig. 22. The distribution overhead and the traffic overhead when performing multicast firmware distributions in EH LoRa networks with DCs regulated

FioRa+ (OPT) renders up to $23.7\times$ shorter distribution duration in a duty cycle-free EH LoRa network and up to $113\times$ shorter distribution duration in a duty-cycled EH LoRa network, respectively.

6.2.5 Distribution Overhead. As illustrated on the left of Figures 21 and 22, LoRaWAN exhibits the highest firmware distribution overhead when compared to all other frameworks following energy neutrality-aware multicast firmware distributions. Without knowing EH LoRa sensors' available energy, LoRaWAN kept distributing firmware data to the multicast group of EH LoRa sensors even though some of them did not have enough energy to be active and simultaneously receive those firmware data. Because of the scarce harvested energy, EH LoRa sensors with low EH rates missed a lot of firmware data distributed by the server when running LoRaWAN. As a result, LoRaWAN server had to perform additional multicast firmware distributions until all EH LoRa sensor being able to successfully reconstruct firmware images. This drastically increases the overhead of firmware distributions. On the other hand, ENO, FioRa, and FioRa+ variants render comparable distribution overheads. By following energy-neutral objectives, these frameworks only distributed original firmware data to EH LoRa sensors and triggered no additional firmware distributions. As validated through the first diagrams of Figures 21, 22, and 23, the firmware distribution overheads of ENO, ENO+, FioRa, and FioRa+ thus increase linearly as the size of firmware images increases. Compared with LoRaWAN, the left of Figures 21 and 22 demonstrates that

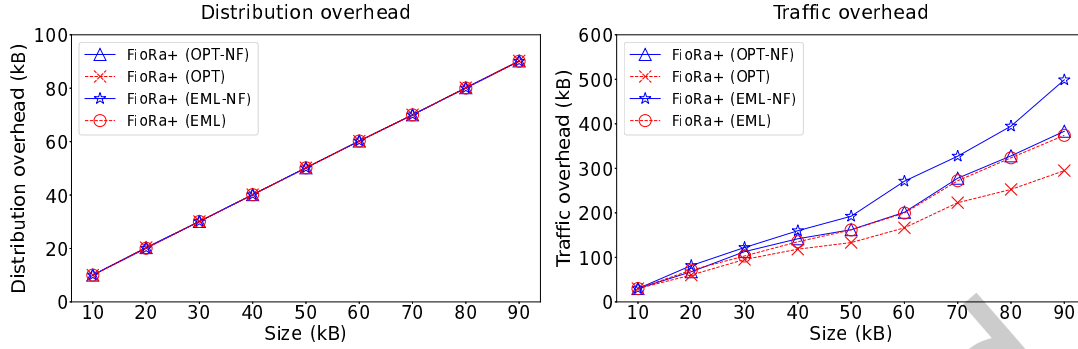


Fig. 23. The distribution overhead and the traffic overhead when performing multicast firmware distributions using different FioRa+ variants in duty-cycled EH LoRa networks

FioRa+ (OPT) minimizes firmware distribution overheads by up to 9.5 \times in a duty cycle-less EH LoRa network and by up to 22.7 \times in a duty-cycled EH LoRa network, respectively.

6.2.6 Traffic Overhead. The right of Figures 21, 22, and 23 exhibits the traffic overhead of different frameworks. Unlike the distribution overhead, the traffic overhead additionally includes the overhead of control messages exchanged between the server and EH LoRa sensors (see Section 6.1.3). Although FioRa+, FioRa, and ENO variants introduce the control traffic overhead related to energy query, relay, and one-hop neighbor discovery mechanisms (i.e., the last two mechanisms are only applicable for FioRa+ and FioRa variants) to EH LoRa networks, such overhead is deemed negligible when compared with the additional firmware distribution overhead that can be reduced. In other words, the amount of distributed firmware data dominates the traffic overhead.

As illustrated on the right of Figures 21 and 22, the traffic overhead of all frameworks increases as the size of firmware images increases. Here, an increase in the size of firmware image necessitates the employment of more multicast sessions (see the left of Figures 18 and 19), which in turn increases the control messages exchanged. Without the knowledge of energy availability, LoRaWAN server kept distributing (additional) firmware data to EH LoRa sensors until they could successfully reconstruct their firmware images. To configure any additional multicast session, the server had to communicate a session setup command to each unsuccessful EH sensor individually, increasing the control traffic overhead. Consequently, LoRaWAN's traffic overhead overshoots all the other frameworks. It has, for example, up to 9 \times and 11 \times higher traffic overheads than FioRa+ (OPT) in duty cycle-free and duty-cycled EH LoRa networks, respectively. Compared with FioRa+ and FioRa variants, ENO variants render slightly higher traffic overhead. Thanks to the use of low DR, the server running ENO or ENO+ could distribute less firmware data per multicast session. Namely, it required the larger number of multicast sessions to distribute a firmware image. When DCs were regulated, even more multicast sessions were needed because EH LoRa sensors' capability to receive firmware data reduced (i.e., they wasted the significant amounts of energy during gateway's off time). Consequently, the traffic overheads of ENO and ENO+ increased as more control messages were exchanged. Apart from an increase in the number of multicast sessions, the failure to configure multicast sessions also induced more control traffic overhead. Compared with FioRa+ (OPT), the right of Figures 21 and 22 illustrates that ENO variants (i.e., ENO and ENO+) render up to 1.5 \times and 4 \times higher traffic overheads in duty cycle-less and duty-cycled EH LoRa networks, respectively.

Furthermore, the second diagrams of Figures 21, 22 and 23 demonstrate that the overhead of the flexible energy query mechanism can be minor, especially when the size of firmware images and/or the number of multicast sessions increases. Through the arrays of future harvested energy reported by EH LoRa sensors, the server could

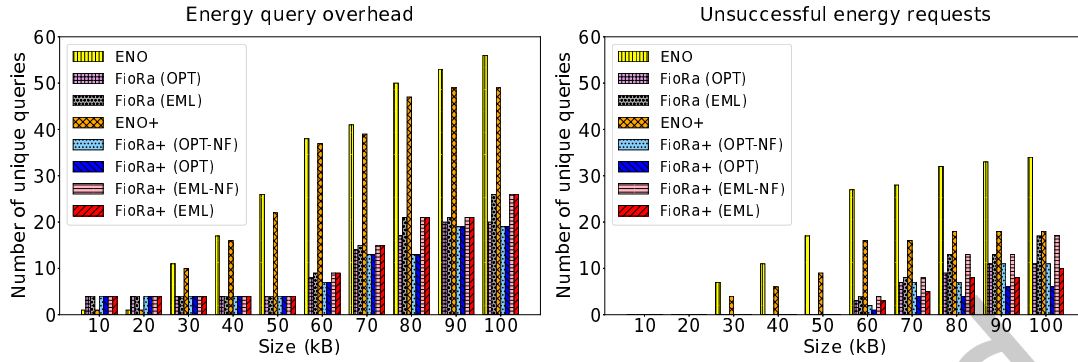


Fig. 24. Energy query overhead and the number of unsuccessful energy requests when performing multicast firmware distributions in EH LoRa networks with no DCs regulated

perform multicast firmware distributions using time-slots and session configurations that eventually minimized the completion time and the total number of multicast sessions required (see Sections 6.2.2 and 6.2.3). When some EH LoRa sensors failed to complete the flexible energy query mechanism, the server could only transmit new energy queries to such sensors, provided that the information previously reported by other EH LoRa sensors satisfied the minimum requirement (see Sections 5.1 and 5.3). As shown on the right of Figure 23, the ability to reuse the energy information (i.e., the energy available from the next $k = 3$ time-slots onward) of FioRa+ (OPT) and FioRa+ (EML) thus results in the lower traffic overhead when compared with FioRa+ (OPT-NF) and FioRa+ (EML-NF). When DC restrictions and relay-signal collisions were disabled, the right of Figure 21 suggests that FioRa+'s traffic overhead was comparable to FioRa's. Despite an increase in size of predictive harvested energy reported, the flexible energy query mechanism of FioRa+ enabled the server to complete multicast firmware distributions (and/or neighbor discovery) faster, using the fewer number of multicast sessions than FioRa. Again, the shorter completion time helped the (half-duplex) LoRa gateway return to regular operations quicker. The fewer number of multicast sessions, on the other hand, reduced the number of energy queries/responses exchanged. In a real-world EH LoRa network, we envision that the traffic overhead of FioRa+ can be remarkably lower than FioRa. Specifically, the ability to reuse the reported predictive energy information makes FioRa+ less sensitive to packet loss and/or the spatio-temporal heterogeneity of EH rates.

Given the overhead of firmware distributions (aka the distribution overhead) dominated the traffic overhead, the coverage assessment mechanism further helped the server suppress the unavailing multicast firmware distributions. Namely, it ensured that all EH LoRa sensors were able to receive the firmware data simultaneously. For a fair comparison, we allowed all frameworks to abort firmware distributions if a multicast session was not successfully configured. Unlike FioRa, FioRa+ further suspended the scheduled multicast firmware distributions if there existed some out-of-range sensors unreachable by the successfully-instructed relays.

6.2.7 Energy Query Overhead. We defined the energy query overhead as the number of energy queries with unique IDs transmitted by the server (see Section 6.1.3). In our simulations, the IDs of energy queries were sequentially increased by 1 whenever the (initial) scheduled time reached. Here, the left of Figures 24, 25, and 26 demonstrates that the energy query overheads of all frameworks (i.e., the frameworks capable of querying EH LoRa sensors' future energy availabilities) increase as the sizes of firmware images increase. As discussed previously, multiple multicast sessions might be required when large firmware images were distributed, low DRs were used for firmware distributions, and/or DCs were regulated. Being newly sent per each multicast session, the overhead of energy queries thus grows as the number of multicast sessions grows. Apart from the number of

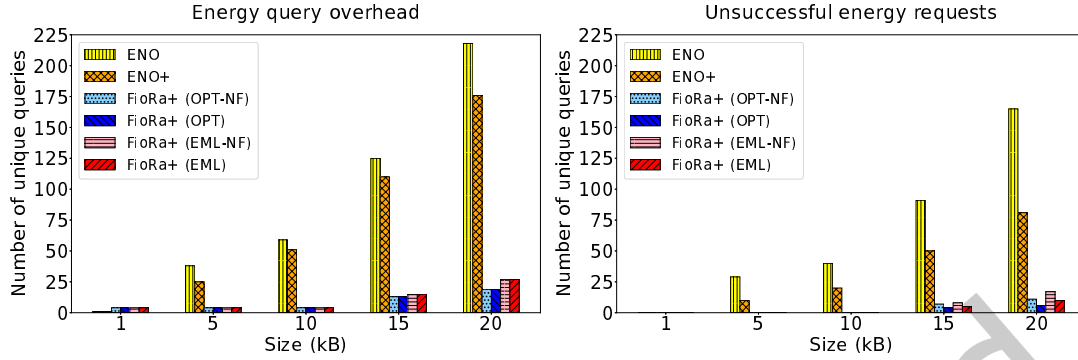


Fig. 25. Energy query overhead and the number of unsuccessful energy requests when performing multicast firmware distributions in EH LoRa networks with DCs regulated

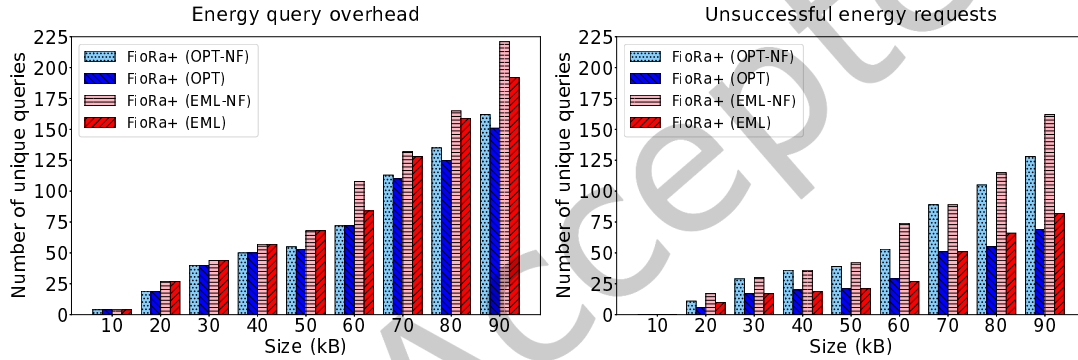


Fig. 26. Energy query overhead and the number of unsuccessful energy requests when performing multicast firmware distributions using FioRa+ variants in EH LoRa networks with DCs regulated

multicast sessions, the rescheduling of multicast firmware distributions caused by unsuccessful energy queries (see Section 6.2.8) also contributed to increasing energy query overhead.

By using the larger number of multicast sessions (see the left of Figures 18 and 19), the energy query overheads of ENO and ENO+ overshoot FioRa+ and FioRa variants. For example, considering the distributions of the same size of firmware images in a duty cycle-less EH LoRa network (see the left of Figure 24), the energy query overheads of ENO and ENO+ could be up to $6.5\times$ and $5.5\times$ higher than FioRa+ (OPT)'s, respectively. When it comes to a duty-cycled EH LoRa network (see the left of Figure 25), the energy query overhead of ENO and ENO+ could, respectively, be up to $15\times$ and $13\times$ higher than FioRa+ (OPT)'s. Compared with FioRa variants, the left of Figure 24 also shows that FioRa+ variants render smaller (if not comparable) energy query overheads. Given again the distributions of the same size of firmware images in the duty cycle-less EH LoRa network, FioRa (OPT) could, for instance, have up to $1.3\times$ higher energy query overhead than FioRa+ (OPT). In addition to the use of the larger number of multicast sessions (see the left of Figure 18), the server executing FioRa (OPT) or FioRa (EML) had to communicate new energy queries to all EH LoRa sensors whenever the energy query mechanism failed (e.g., at least one EH LoRa sensor had insufficient predictive energy to receive any firmware data at the requested time). The server's inability to successfully schedule multicast firmware distributions thus exacerbated FioRa's

energy query overhead. Unlike FioRa, the server executing FioRa+ could make use of the arrays of time-slotted predictive energy reported by EH LoRa sensors to schedule multicast firmware distributions on or after the initial schedule time, provided that the distributed firmware data were maximized. As a result, FioRa+ server limited the transmissions of (new) energy queries, reducing downlink traffic in EH LoRa networks. As discussed previously, minimizing downlink traffic is crucial for enhancing half-duplex LoRa gateway's ability to receive uplink LoRa packets. When the flexible energy mechanism was enabled, the first diagrams of Figures 24, 25 and 26 show that ENO+, FioRa+ (OPT), and FioRa+ (EML) have lower energy query overheads than their counterparts. Being able to reuse the time-slotted predictive energy successfully reported by EH LoRa sensors, the server running ENO+, FioRa+ (OPT), or FioRa+(EML) could only transmit energy queries to any EH LoRa sensor facing the energy query failure. Consequently, the transmissions of (new) energy queries were further reduced.

Thanks to the use of energy queries during one-hop neighbor discovery, the energy query overheads of FioRa+ and FioRa variants are higher than ENO variants when distributing the small size of firmware images. In a real-world scenario, the energy query overhead of one-hop neighbor discovery can be minor because the one-hop neighbor discover is performed once at the time of deployment or when the network topology changes.

6.2.8 Unsuccessful Energy Requests. The right of Figures 24, 25, and 26 exhibits the extent of unsuccessful energy requests of different frameworks capable of querying predictive energy from EH LoRa sensors. Throughout our simulations, the unsuccessful energy requests occurred mainly due to two different causes. First, one or more EH LoRa sensors did not have sufficient energy to receive and/or respond to energy queries. Second, the predictive energy reported by any EH LoRa sensor was inadequate for receiving any firmware data. Although failing to configure a multicast session (w.r.t. the overestimation of predictive energy) contributes to the unsuccessful energy requests, we do not observe such failure in our simulations.

As discussed previously, distributing a large firmware image in an energy-neutral manner might require multiple multicast sessions. The number of multicast sessions could become larger when DCs were regulated. In the second diagrams of Figures 24, 25 and 26, the unsuccessful energy requests of all frameworks thus show upward trends as a result of increases in the numbers of multicast sessions and corresponding energy queries in use. When distributing the small sizes of firmware images, FioRa+ and FioRa variants experienced no unsuccessful energy requests. This is because they only required a single multicast session to accommodate the distributions of each firmware image. However, when the sizes of firmware images increased, EH LoRa sensors implementing any framework had to sustain multiple multicast sessions to receive firmware images in full. During each multicast session, some EH LoRa sensors might (nearly) deplete their energy storage, so as to maximize the firmware data received (see Section 4.2). Located in challenging EH conditions, EH LoRa sensors with low EH rates might inadequately recharge their energy storage, failing to complete the energy query mechanism w.r.t. the subsequent multicast session. Given the longer period of multicast firmware distributions, the changes in EH conditions also impacted the success of energy queries.

Compared to the frameworks employing the flexible energy query mechanism, the frameworks using no flexible energy query mechanism render higher unsuccessful energy requests. By having EH LoRa sensors report the energy available at a particular time in the future (i.e., the scheduled time), the right of Figures 24 and 25 shows that ENO and FioRa variants suffer the larger number of unsuccessful energy requests than their counterparts. Given some EH LoRa sensors could not complete the energy query mechanism on time and/or had inadequate (predictive) energy to receive any firmware data at the scheduled time, the server executing ENO, FioRa (OPT), or FioRa (EML) had to drop the predictive energy information successfully reported by EH LoRa sensors as it became out of date. For example, the server could not use the energy available at 8 a.m. to equate the energy available at 1 p.m. (or 6 p.m.) because EH conditions changed dynamically. The inability to adaptively use the predictive energy information received from EH LoRa sensors thus induced unsuccessful energy requests.

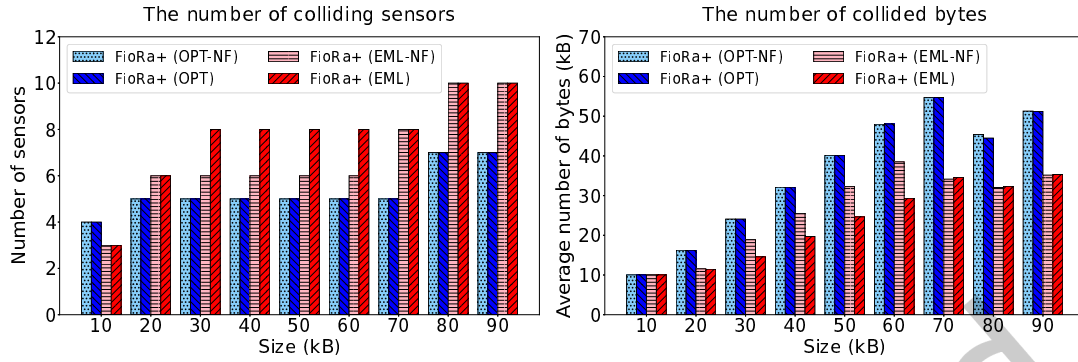


Fig. 27. The number of colliding sensors and the average number of collided bytes per colliding sensor before enabling the time-slotted relay scheduling mechanism in duty-cycled EH LoRa networks

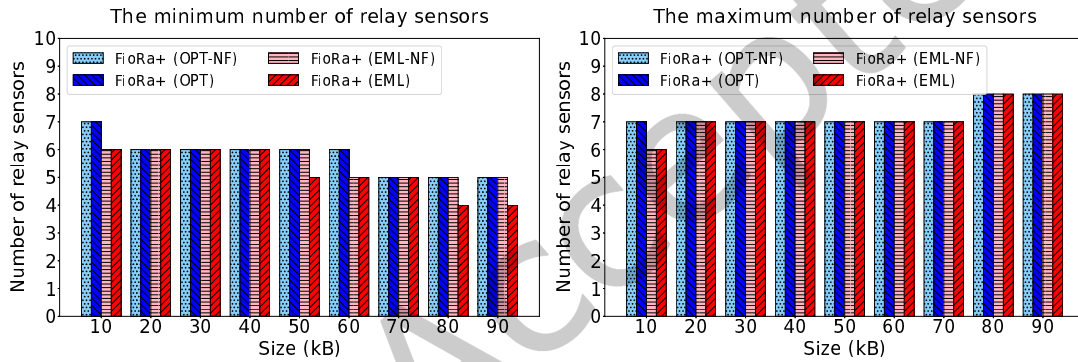


Fig. 28. The minimum and maximum numbers of EH LoRa sensors selected as relay sensors in duty-cycled EH LoRa networks

As opposed to previous frameworks, ENO+ and FioRa+ variants enabled the server to defer multicast firmware distributions. By leveraging the arrays of time-slotted predictive energy (i.e., the energy available at the scheduled time onward) reported by EH LoRa sensors, the server could configure a multicast session at the time other than the (initial) scheduled time, provided that all EH LoRa sensors could simultaneously receive firmware data. In other words, the server selected the time-slot maximizing the data reception rate for multicast firmware distributions. Otherwise, it sent new energy queries to all EH LoRa sensors at least 3 hours (i.e., the configuration period employed in our simulations) prior to the next (initial) scheduled time. Despite the larger size of energy-query responses, reporting the array of time-slotted predictive energy effectively helped ENO+ and FioRa+ variants limit the unsuccessful transmission of energy requests. In addition to deferring multicast firmware distributions, ENO+, FioRa+ (OPT) and FioRa+ (EML) further dealt with the absence of energy-query responses from EH LoRa sensors. Through the flexible energy query mechanism, the server could dynamically postpone the scheduled time and reuse the time-slotted predictive energy successfully reported by EH LoRa sensors for session management, provided that the minimum requirement was met (see Section 5.1). Then, it only transmitted new energy queries to EH LoRa sensors failing to receive and/or respond to the past energy queries. As a result, the amounts of unsuccessful energy requests of both frameworks were further reduced.

Table 2. The Number of Time-slots Available when The Server and Relay Sensors Use Different DRs for Multicast Firmware Distributions given EU868 Frequency Plan [8] and 10% DC Regulations

Original DR	Payload (B)	T_{on} (s)	T_{off} (s)	Number of Relay Slots					
				DR5	DR4	DR3	DR2	DR1	DR0
DR5	239	0.39	3.55	8	-	-	-	-	-
DR4	239	0.7	6.27	14	8	-	-	-	-
DR3	112	0.68	6.09	27	15	8	-	-	-
DR2	48	0.7	6.29	52	29	16	8	-	-
DR1	48	1.48	13.31	111	63	35	18	8	-
DR0	48	2.79	25.14	212	121	67	35	16	8

Considering the distributions of the same size of firmware images in a duty cycle-less EH LoRa network, the right of Figure 24 reveals that ENO, ENO+, FioRa (OPT), and FioRa+ (OPT-NF) render up to 27 \times , 16 \times , 3 \times , and 2 \times larger unsuccessful energy requests than FioRa+(OPT), respectively. When it comes to a duty-cycled EH LoRa network, the right of Figures 25 and 26 (the latter figure is applicable to FioRa+ (OPT-NF)) demonstrates that ENO, ENO+, and FioRa+ (OPT-NF), respectively, have up to 40 \times , 20 \times , and 2 \times larger unsuccessful energy queries than FioRa.

6.2.9 Relay Signal Collisions. In our simulations, the relay signal collisions occurred when any EH LoRa sensor (so-called out-of-range sensor) simultaneously heard data fragments transmitted by two or more relay sensors using the same DR. Considering severe packet losses that might occur, we thus excluded FioRa (i.e., FioRa did not support relay collision avoidance) from the second and the third experiments for the sake of fair performance comparisons (see Section 6.1.1). In such experiments, the relay signal collisions were activated in a 10% duty-cycled downlink channel. Unlike FioRa and FioRa+, the frameworks including LoRaWAN, ENO, and ENO+ did not support a relay mechanism. Consequently, they did not experience any relay signal collisions in our experiments.

Aiming to investigate the efficiency of time-slotted relay scheduling algorithm, we mainly focused on evaluating the performances of FioRa+ variants in a duty-cycled EH LoRa network (i.e., the third experiment). Here, the first and the second diagrams of Figure 27 show, respectively, the number of out-of-range sensors facing relay signal collisions (aka colliding sensors) and the average number of collided bytes per colliding sensor before the relay scheduling algorithm was applied. Among 20 EH LoRa sensors, the left of Figure 27 illustrates that the number of colliding sensors per simulation was in the range of [3, 10]. In the worst case, each of them could fail to receive almost all the relayed firmware data if the relay scheduling algorithm was not enabled. For example, the right of Figure 27 exhibits that each of 3–4 colliding sensors could fail to receive approximately 10 kB of firmware data when a 10-kB firmware image was distributed. Basically, the severe relay signal collisions occurred because, thanks to the use of high DRs, FioRa+ employed a single multicast session (i.e., the relay sensors were selected once) for distributing such small firmware image.

Through the time-slotted relay scheduling algorithm, the relay sensors causing relay signal collisions were assigned different time-slots for transmitting firmware data. Out of 20 EH LoRa sensors, Figure 28 shows that the minimum (left) and the maximum (right) numbers of relay sensors per simulation were in the range of [4, 8]. Given the DRs used by the server and the relay sensors were not slower than DR2 (see Section 6.1.1), Table 2 demonstrates that the numbers of time-slots (aka relay slots) available for assignments were in the range of 8 to 52 slots. Because the numbers of relay slots sufficed for accommodating all the relay sensors, the relay signal collisions did not occur in our simulations. In other words, relay signal collisions were completely addressed after the relay scheduling algorithm was executed. It is worth emphasizing that any relay slot can accommodate

multiple relay sensors using the same DR, provided that they cause no (or the fewest) relay signal collisions. Because the session management algorithm selected the relay sensors and the DRs for multicast firmware distributions dynamically, the assignment of relay slots could be different for each multicast session. Therefore, the relay scheduling algorithm does not cause systematic relay signal collisions in EH LoRa networks.

7 Conclusion

This article presented FioRa+, the first energy neutrality-aware multicast firmware distribution framework for EH LoRa networks. Unlike FioRa, FioRa+ allows the server to schedule multicast firmware distributions at any time when the data reception rate is maximized. Despite having different EH rates, it ensures that a multicast group of EH LoRa sensors can simultaneously receive firmware data from the server without encountering power failures. To minimize the completion time of firmware updates and the overhead of firmware distributions, FioRa+ spreads firmware distributions over multiple multicast sessions. Each multicast session employs high DR(s) to perform multicast firmware distributions in an energy-neutral manner according to the maximum energy harvested by the EH LoRa sensors in the multicast group. To alleviate the impact of high DRs on the coverage of multicast firmware distributions, FioRa+ employs energy-neutral one-hop neighbor discovery and on-demand relay mechanisms to ensure that all EH LoRa sensors in the multicast group are at most two hops away from a LoRa gateway. By leveraging the arrays of predictive harvested energy reported by the EH LoRa sensors, FioRa+ enables the server to adaptively select the timing of multicast firmware distributions and session configurations to minimize the firmware distribution time. Coupling flexible energy query with adaptive session management, FioRa+ minimizes the unsuccess of energy queries. Time-slotted relay scheduling also prevents the collisions of data fragments relayed using identical DRs. Assisted by coverage assessment mechanism, unnecessary multicast firmware distributions are eliminated, limiting the overhead of firmware distributions. Validated through trace-driven simulations, experimental results show that FioRa+ outperforms all the existing methods, especially when DCs are regulated. Compared with FioRa, FioRa+ reduces unsuccessful energy requests and minimizes the completion time of firmware updates in EH LoRa networks.

References

- [1] Khaled Abdelfadeel, Tom Farrell, David McDonald, and Dirk Pesch. 2020. How to Make Firmware Updates over LoRaWAN Possible. In *2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. 16–25. doi:10.1109/WoWMoM49955.2020.00018
- [2] Ferran Adelantado, Xavier Vilajosana, Pere Tuset-Peiro, Borja Martinez, Joan Melia-Segui, and Thomas Watteyne. 2017. Understanding the Limits of LoRaWAN. *IEEE Communications Magazine* 55, 9 (2017), 34–40. doi:10.1109/MCOM.2017.1600613
- [3] Muhammad Ali Lodhi, Mohammad S. Obaidat, Lei Wang, Khalid Mahmood, Khalid Ibrahim Qureshi, Jenhui Chen, and Kuei-Fang Hsiao. 2024. Tiny Machine Learning for Efficient Channel Selection in LoRaWAN. *IEEE Internet of Things Journal* 11, 19 (2024), 30714–30724. doi:10.1109/JIOT.2024.3413585
- [4] Martin Bor, John Vidler, and Utz Roedig. 2016. LoRa for the Internet of Things. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks (Graz, Austria) (EWSN '16)*. Junction Publishing, USA, 361–366.
- [5] Marco Centenaro, Lorenzo Vangelista, Andrea Zanella, and Michele Zorzi. 2016. Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios. *IEEE Wireless Communications* 23, 5 (2016), 60–67. doi:10.1109/MWC.2016.7721743
- [6] LoRa Alliance Technical Committee. 2020. LoRaWAN® L2 1.0.4 Specification (TS001-1.0.4). *LoRa Alliance: Fremont, CA, USA* (Oct. 2020).
- [7] LoRa Alliance Technical Committee. 2022. LoRaWAN® Fragmented Data Block Transport Specification TS004-2.0.0. *LoRa Alliance: Fremont, CA, USA* (April 2022).
- [8] LoRa Alliance Technical Committee. 2022. LoRaWAN® Regional Parameters RP002-1.0.4. *LoRa Alliance: Fremont, CA, USA* (Sept. 2022).
- [9] LoRa Alliance Technical Committee. 2022. LoRaWAN® Remote Multicast Setup Specification TS005-2.0.0. *LoRa Alliance: Fremont, CA, USA* (April 2022).
- [10] Valentina Di Vincenzo, Martin Heusse, and Bernard Tourancheau. 2019. Improving Downlink Scalability in LoRaWAN. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. 1–7. doi:10.1109/ICC.2019.8761157
- [11] Koustabh Dolui, Ashok Samraj Thangarajan, Thibo Claes, Sam Michiels, and Danny Hughes. 2022. Towards On-Board Learning for Harvested Energy Prediction. In *Proceedings of the 6th International Workshop on Embedded and Mobile Deep Learning* (Portland, Oregon)

- (EMDL '22). Association for Computing Machinery, New York, NY, USA, 7–12. doi:10.1145/3539491.3539593
- [12] Sezana Fahmida, Venkata Prashant Modekurthy, Dali Ismail, Aakriti Jain, and Abusayeed Saifullah. 2022. Real-Time Communication over LoRa Networks. In *2022 IEEE/ACM Seventh International Conference on Internet-of-Things Design and Implementation (IoTDI)*. 14–27. doi:10.1109/IoTDI54339.2022.00019
- [13] Sezana Fahmida, Venkata P Modekurthy, Mahbubur Rahman, Abusayeed Saifullah, and Marco Brocanelli. 2020. Long-Lived LoRa: Prolonging the Lifetime of a LoRa Network. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. 1–12. doi:10.1109/ICNP49622.2020.9259375
- [14] Amalinda Gamage, Jansen Christian Liando, Chaojie Gu, Rui Tan, and Mo Li. 2020. LMAC: efficient carrier-sense multiple access for LoRa. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (London, United Kingdom) (MobiCom '20)*. Association for Computing Machinery, New York, NY, USA, Article 43, 13 pages. doi:10.1145/3372224.3419200
- [15] Kai Geissdoerfer and Marco Zimmerling. 2022. Learning to Communicate Effectively Between Battery-free Devices. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 419–435. <https://www.usenix.org/conference/nsdi22/presentation/geissdoerfer>
- [16] Neal Jackson, Joshua Adkins, and Prabal Dutta. 2019. Capacity over Capacitance for Reliable Energy Harvesting Sensors. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks (Montreal, Quebec, Canada) (IPSN '19)*. Association for Computing Machinery, New York, NY, USA, 193–204. doi:10.1145/3302506.3310400
- [17] Sukanya Jewsakul and Edith C.H. Ngai. 2023. EmbientLoRa: Embedded Intelligence for Predictive Energy Harvesting and Management in LoRa Networks. In *Proceedings of the 2023 International Conference on Embedded Wireless Systems and Networks (Rende, Italy) (EWSN '23)*. Association for Computing Machinery, New York, NY, USA, 231–236.
- [18] Sukanya Jewsakul and Edith C. H. Ngai. 2023. ENORA: Empowering Energy-Neutral Operation in LoRa Networks via Embedded Intelligence. *IEEE Network* 37, 4 (2023), 127–134. doi:10.1109/MNET.010.2200662
- [19] Sukanya Jewsakul and Edith C. H. Ngai. 2024. FioRa: Energy Neutrality-aware Multicast Firmware Distributions in Energy-harvesting LoRa Networks. In *Proceedings of the 11th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (Hangzhou, China) (BuildSys '24)*. Association for Computing Machinery, New York, NY, USA, 88–98. doi:10.1145/3671127.3698174
- [20] Sukanya Jewsakul and Edith C. H. Ngai. 2025. RACEME: Embedded Intelligence for Correlation-driven Predictive Energy-harvesting Management in LoRa Networks. *ACM Trans. Sen. Netw.* 21, 2, Article 14 (March 2025), 38 pages. doi:10.1145/3715129
- [21] Jansen C. Liando, Amalinda Gamage, Agustinus W. Tengourtius, and Mo Li. 2019. Known and Unknown Facts of LoRa: Experiences from a Large-Scale Measurement Study. *ACM Trans. Sen. Netw.* 15, 2, Article 16 (Feb. 2019), 35 pages. doi:10.1145/3293534
- [22] Songran Liu, Mingsong Lv, Wei Zhang, Xu Jiang, Chuancai Gu, Tao Yang, Wang Yi, and Nan Guan. 2023. Light Flash Write for Efficient Firmware Update on Energy-harvesting IoT Devices. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 1–6. doi:10.23919/DATES6975.2023.10136990
- [23] Chollet Nicolas, Bouchemal Naila, and Ramdane-Cherif Amar. 2022. Energy efficient Firmware Over The Air Update for TinyML models in LoRaWAN agricultural networks. In *2022 32nd International Telecommunication Networks and Applications Conference (ITNAC)*. 21–27. doi:10.1109/ITNAC55475.2022.9998338
- [24] Lukas Sigrist, Andres Gomez, and Lothar Thiele. 2019. Dataset: Tracing Indoor Solar Harvesting. In *Proceedings of the 2nd Workshop on Data Acquisition To Analysis (New York, NY, USA) (DATA'19)*. Association for Computing Machinery, New York, NY, USA, 47–50. doi:10.1145/3359427.3361910
- [25] Lukas Sigrist, Andres Gomez, and Lothar Thiele. 2019. Long-Term Tracing of Indoor Solar Harvesting. doi:10.5281/zenodo.3363925
- [26] Naomi Stricker and Lothar Thiele. 2022. Accurate Onboard Predictions for Indoor Energy Harvesting using Random Forests. In *2022 11th Mediterranean Conference on Embedded Computing (MECO)*. 1–6. doi:10.1109/MECO55406.2022.9797188
- [27] Zehua Sun, Tao Ni, Huanqi Yang, Kai Liu, Yu Zhang, Tao Gu, and Weitao Xu. 2023. FLoRa: Energy-Efficient, Reliable, and Beamforming-Assisted Over-The-Air Firmware Update in LoRa Networks. In *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks (San Antonio, TX, USA) (IPSN '23)*. Association for Computing Machinery, New York, NY, USA, 14–26. doi:10.1145/3583120.3586963
- [28] Zehua Sun, Huanqi Yang, Kai Liu, Zhimeng Yin, Zhenjiang Li, and Weitao Xu. 2022. Recent Advances in LoRa: A Comprehensive Survey. *ACM Trans. Sen. Netw.* 18, 4, Article 67 (Nov. 2022), 44 pages. doi:10.1145/3543856
- [29] Anastasios Valkanis, Georgia A. Beletsoti, Konstantinos F. Kantelis, Petros Nicosopolitidis, and Georgios I. Papadimitriou. 2024. Ensuring Reliability for LoRa Networks Using a Reinforcement-Learning-Assisted Time Division Duplex Protocol. *IEEE Internet of Things Journal* 11, 3 (2024), 4179–4190. doi:10.1109/JIOT.2023.3300541
- [30] Thiemo Voigt, Martin Bor, Utz Roedig, and Juan Alonso. 2017. Mitigating Inter-network Interference in LoRa Networks. In *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks (Uppsala, Sweden) (EWSN '17)*. Junction Publishing, USA, 323–328.
- [31] Wei Wei, Sahidul Islam, Jishnu Banerjee, Shanglin Zhou, Chen Pan, Caiwen Ding, and Mimi Xie. 2022. An Intermittent OTA Approach to Update the DL Weights on Energy Harvesting Devices. In *2022 23rd International Symposium on Quality Electronic Design (ISQED)*. 1–6. doi:10.1109/ISQED54688.2022.9806295

- [32] Xianjin Xia, Qianwu Chen, Ningning Hou, Yuanqing Zheng, and Mo Li. 2023. XCopy: Boosting Weak Links for Reliable LoRa Communication. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking (Madrid, Spain) (ACM MobiCom '23)*. Association for Computing Machinery, New York, NY, USA, Article 14, 15 pages. doi:10.1145/3570361.3592516
- [33] Cheuk-Wang Yau, Sukanya Jewsakul, Man-Ho Luk, Angela P. Y. Lee, Yun-Hin Chan, Edith C. H. Ngai, Philip W. T. Pong, King-Shan Lui, and Jiangchuan Liu. 2022. NB-IoT Coverage and Sensor Node Connectivity in Dense Urban Environments: An Empirical Study. *ACM Trans. Sen. Netw.* 18, 3, Article 49 (Sept. 2022), 36 pages. doi:10.1145/3536424
- [34] Shiming Yu, Xianjin Xia, Ziyue Zhang, Ningning Hou, and Yuanqing Zheng. 2024. FDLora: Tackling Downlink-Uplink Asymmetry with Full-duplex LoRa Gateways. In *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems (Hangzhou, China) (SenSys '24)*. Association for Computing Machinery, New York, NY, USA, 281–294. doi:10.1145/3666025.3699338

Received 28 February 2025; revised 15 May 2025; accepted 6 June 2025

Just Accepted