

# Backpack-LoRa: Energy-Efficient Multi-Hop Networking for LoRaWANs

Daniel Szafranski  
Department of Informatics  
TU Clausthal, Germany  
daniel.szafranski@tu-clausthal.de

Moosa Sharafeldin  
Department of Informatics  
TU Clausthal, Germany  
mosh12@tu-clausthal.de

Andreas Reinhardt  
Department of Informatics  
TU Clausthal, Germany  
reinhardt@ieee.org

**Abstract**—The rising popularity of the Internet of Things and wireless sensing applications calls for energy-efficient and scalable communication protocols. LoRaWAN is one of today’s most prominent choices for this purpose, as it features long communication ranges and high energy-efficiency. However, a major drawback is its limitation to being operated in a star topology. Thus, data originating from nodes outside of the single-hop neighborhood of a gateway cannot be received. We overcome this limitation by presenting *Backpack-LoRa*, an extension to LoRaWAN which enables multi-hop uplink communication while being compatible with existing deployments. Instead of simply re-transmitting overheard frames entirely, however, *Backpack-LoRa* nodes extract and buffer only the relevant frame content for later decryption and append it to their own frames, which are transmitted during the next regular transmission slots. To address the limitation of most commodity transceivers – only being able to listen on one out of multiple available channels – we propose a pseudo-random channel allocation algorithm based on time and device address for *Backpack-LoRa*. We implement our approach in a real-world outdoor deployment consisting of four nodes and gateways. Subsequently, we analyze the improvements in network coverage, packet reception rate, and energy overhead. Our results show that *Backpack-LoRa* enables nodes to reach gateways which were unreachable for standard LoRaWAN and increase the packet reception rate by up to 54.9%. As compared to a simple re-transmission approach, *Backpack-LoRa* can reduce the energy overhead by 28.8%.

**Index Terms**—LoRa; LoRaWAN; multi-hop; energy-efficiency; packet forwarding

## I. INTRODUCTION

Long Range Wide Area Network (LoRaWAN) is a frequently used protocol for wireless communication, offering wide coverage up to several kilometers and high energy-efficiency. Its star network topology allows to simplify the Medium Access Control (MAC) layer implementation and to seamlessly integrate additional devices. However, this naturally comes at the limitation of single-hop communication between nodes and gateways. Neither LoRaWAN’s physical layer modulation scheme, Long Range (LoRa), nor the protocol itself offer multi-hop communication capabilities. In case a node is outside of a gateway’s reception range, its transmitted packets cannot be received, leading to packet loss and thus the waste of valuable resources on the energy-constrained end device. As the wireless link quality is known to be dependent on the ambient conditions, e.g., changing vegetation [1] or weather [2, 3], packet reception rates (PRRs) naturally also

fluctuate over time. Large temperature fluctuations are capable of temporarily completely disabling a previously reliable LoRa link [4]. The LoRaWAN specification [5] hence defines a mitigation strategy using acknowledgements (ACKs) to confirm successful reception on the gateway by sending downlinks, i.e., frames sent from the gateway to the end device. In absence of such a confirmation, an end device may try to re-transmit the lost frame. As the LoRaWAN header comes at a significant size of 12–13 bytes per uplink [5], the overhead is considerably large for re-transmissions, especially for the small payloads that are typically used in Internet of Things (IoT) applications and wireless sensor networks (WSNs), making it a very inefficient way to handle unreliable links. Therefore, various approaches were introduced to extend LoRa(WAN) by multi-hop networking, e.g., [6–9]. However, most related works come at the cost of high energy overheads, as they mainly focus on simply relaying transmitted frames. While these approaches can increase connectivity, the corresponding energy overhead is simply shifted to intermediate relay nodes.

In our paper, we address these limitations and present *Backpack-LoRa*, an extension to LoRaWAN which adds support for multi-hop uplinks focusing on energy-efficiency. Our approach is based on the fact, that LoRaWAN devices typically transmit their data, e.g., sensor readings or measurements of ambient conditions, in periodic intervals. Determining these transmission times and their periodicity allows us to foresee future transmission times for individual nodes. *Backpack-LoRa* is orchestrated by the application server, which leverages this fact and creates transmission profiles for individual nodes. These profiles are then transmitted to surrounding nodes, informing them about future transmissions of their neighbors. Since most end devices run a LoRa radio with only a single receive chain (allowing them to listen on only one of multiple defined channels [10]), we propose a pseudo-random algorithm for channel allocation, which allows to determine the future channel selection of neighbors and to overhear their transmissions. Instead of simply relaying the entire LoRaWAN frame, nodes only extract the relevant data for later decryption on the application server. During their next regular transmission slot, nodes append the extracted data to their own data and transmit it as part of their own frame. This allows *Backpack-LoRa* to significantly reduce the required airtimes and increase energy-efficiency.

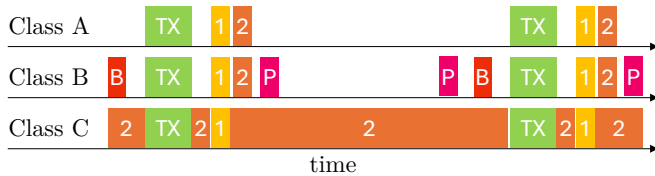


Fig. 1: Sample timing illustration of LoRaWAN device classes A, B and C [5]. Besides node-to-gateway uplink transmissions (marked TX, green), all devices have receive windows RX1 (marked 1, yellow) and RX2 (marked 2, orange) for gateway-to-node downlink traffic. Class B devices furthermore feature beacon (marked B, red) and ping (marked P, purple) slots.

## II. RELATED WORK

LoRa and LoRaWAN are intended to be used in a star topology, thus only allowing for single-hop communication, which naturally comes with reduced coverage and PRRs. In an attempt to overcome this limitation, numerous works have investigated and proposed concepts to enable multi-hop communication and mesh networking.

### A. Multi-hop Networking with LoRa

Already in 2017, Liao et al. investigated multi-hop networking in LoRa [11]. The authors proposed a concurrent transmission (CT)-based multi-hop scheme, where nodes simply forward received packets immediately. Their results show an average PRR of up to 99 % in low density deployments. In [12], a mesh-networking approach is presented, which implements a network discovery algorithm, such that nodes within reach can be recognized and used for routing packets towards a gateways. The authors evaluated their approach on a campus deployment and were able to increase the packet delivery ratio (PDR) from 58.7 % to 88.49 % as compared to standard LoRa. Abrardo and Pozzebon investigated the transmission range of LoRa in the *Bottini di Siena*, an underground system of medieval aqueducts, in [13]. They found that the communication range is significantly reduced to approximately 200 m, making multi-hop communication a necessity. The authors proposed a linear chain topology approach, where each node forwards data from the previous to the following one until the data finally reaches the gateway and introduced a synchronization algorithm to adjust the transmission, reception and sleep windows of neighboring nodes, reducing the power dissipation by up to 50 %. The authors in [14] present *RLMAC*, an alternative MAC protocol to enable Routing Protocol for Low power and Lossy Networks (RPL) multi-hop communication in LoRa networks. In order to handle the limitations of commodity transceiver chips used in end devices, i.e., only allowing to listen on a single Spreading Factor (SF) at a time, the authors propose a timeslot based SF allocation, which enables communication with neighboring nodes and allows to select the optimal SF for each link. Also open-source projects like *Meshtastic*<sup>1</sup> exist, which provide LoRa-based mesh networking implementations for common prototyping boards, including ESP32 and nRF52840. Further works on multi-hop networking for LoRa can be found in [7, 15–19].

<sup>1</sup>Available at <https://meshtastic.org> and <https://github.com/meshtastic>

### B. Extension to LoRaWAN

The LoRaWAN specification [5] introduces three device classes A, B and C, which are summarized in Figure 1. Class A devices open their downlink receive windows *RX1* and *RX2* after their own uplink transmission. Class B devices additionally use beacons for time synchronization and further receive windows (*ping* slots). And finally, Class C devices continuously open *RX2* receive windows. This further complicates LoRaWAN compatible multi-hop extensions on top of LoRa.

Dias and Grilo introduced a LoRaWAN compatible multi-hop extension for uplinks based on Destination-Sequenced Distance Vector (DSDV) routing in [20]. The authors differentiate between relay and leaf nodes and attach additional headers to the LoRaWAN frames for routing. Their results show a PRR of 91.6 % and a throughput of 30.26 bit/s in case of four relay nodes. The authors in [21] propose a LoRaWAN *range extender* which support both, up- as well as downlinks. Their downlink compatibility is enabled due to the presence of two receive windows per uplink, which allows relay nodes, called *e-Nodes*, to forward an uplink transmission, receive a potential downlink frame in its own *RX1* window and forward it to the leaf node in its *RX2* window. In [22], *LoRaHop* is presented by Tian et al.. Their concept is interoperable with standard LoRaWAN nodes and incorporates *LoRaDisC* [23], a CT-based collection and dissemination protocol. The authors were able to increase the PRR by up to 98.33 %. Ebi et al. present a time-synchronized multi-hop networking solution in [24]. Their approach divides the network into sub-networks and, opposed to LoRaWAN’s ALOHA-based MAC, time-division multiple access (TDMA) is used. Furthermore, *repeater nodes (RNs)*, which have connectivity with the gateway, forward data from *sensor nodes (SNs)* within the sub-networks. The authors evaluated their approach in underground infrastructure and were able to achieve a packet error rate (PER) of down to 2.2 %. The paper in [25] evaluates and compares the energy overhead of single and multi-hop communication in LoRaWAN. The authors, Aslam et al., show that the use of a two-hop communication can save up to 50 % energy, while increasing the network coverage by 35 %. Further works on extending LoRa(WAN) by multi-hop functionality can be found in [6, 8, 9, 26].

### C. Summary

Overall, various different approaches were proposed in related works. However, as LoRaWAN is the quasi-standard for IoT and WSN applications, approaches only focusing on LoRa [7, 11–19], are not directly compatible with most deployments. Furthermore, most approaches simply relay received packets [6–9, 20–22, 25, 26]. While this allows to simplify integration, it is sub-optimal in terms of energy-efficiency, as the entire packet is practically re-transmitted, including all LoRaWAN header fields, which come at a significant overhead, but are typically of minor relevance for the end application. *Backpack-LoRa* addresses these issues, by implementing a lightweight multi-hop approach which only forwards relevant data from selectively overheard transmissions.

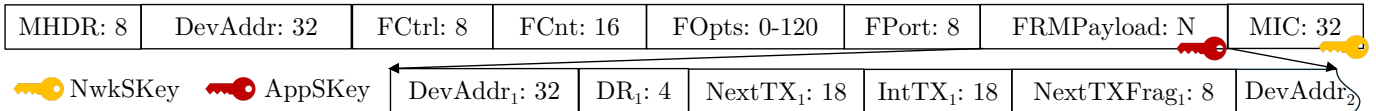


Fig. 2: *Backpack-LoRa* uses standard LoRaWAN downlink frames [5] to inform nodes about their neighbors (fields highlighted by different indices). This includes information about the device addresses, last used datarates, next expected uplinks and transmission intervals. Field sizes are given in bits.

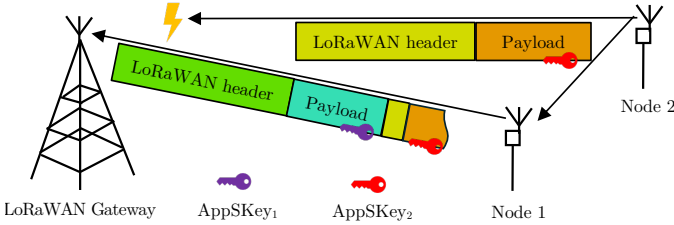


Fig. 3: Illustration of *Backpack-LoRa*. Nodes overhear neighbor transmissions, extract relevant data for later decryption and transmit it as part of their own frame during their next transmission slot.

### III. *Backpack-LoRa*: MULTI-HOP FOR LORAWANS

The high-level overview of *Backpack-LoRa* is illustrated in Figure 3. Nodes are orchestrated by the application server, which informs them about neighbors of interest and their transmission schedules regularly. The application can define and dynamically change nodes of interest, e.g., nodes which belong to the same application but have weak link quality or low PRRs. Nodes then selectively overhear the relevant neighboring uplink transmissions and only extract the data of the LoRaWAN frame, which is required for later decryption and payload extraction. This way, instead of simply forwarding the entire LoRaWAN frame, as mostly done by related work, *Backpack-LoRa* can decrease the overhead significantly, especially for small payload sizes. We present the details of our implementation in the following subsections, addressing the faced challenges and providing efficient solutions.

#### A. General Preconditions and Considerations

First, as LoRa operates in the license-free Industrial, Scientific and Medical (ISM) band, national regulations and *LoRaWAN regional parameters* apply [10]. More specifically, e.g., in the *EU868* band, this means to restrict the *duty cycle* to typically 1%, i.e., an end device must hold back transmissions for at least 99s after it sent an uplink with an airtime of 1s. To fulfill this requirement, *Backpack-LoRa* does not simply relay overheard packets, but rather appends them to their own payload and sends them with its own next planned uplink transmission. Furthermore, up to 16 uplink channels and six SFs, in the range from 7 to 12 are defined for the *EU868* band [10]. Channel selection shall happen *pseudo-randomly* for every uplink transmission, to increase frequency diversity and robustness to interference [5]. This represents a major challenge, because often used commodity LoRa transceivers, e.g., Semtech's SX1276 [27], only feature a single receive chain, allowing to only listening on one channel at a time. Thus, in order to practically enable multi-hop communication with commonly used commodity hardware, nodes must know in advance on which channel and SF their neighbors will

transmit their data. Moreover, in order to enable energy-efficient operation, LoRa radios should stay in sleep mode for as long as possible to reduce energy consumption. Therefore, the expected time of transmission should be known too, to only sporadically switch the radio to receive mode.

#### B. Orchestration

To address these challenges, *Backpack-LoRa* is orchestrated by the application server. First, the server analyzes the transmission behavior of nodes within reach and checks for periodicities. As most LoRaWAN devices transmit data periodically, e.g., sensor readings every ten minutes, the devices' next transmission slots can be easily determined. This even works for low PRRs, as we extrapolate the next transmission slot  $t_{next}$  based on the last received time  $t_1$  and frame counter  $fcnt_1$  in relation to the previous reception of  $t_0$  and  $fcnt_0$  as shown in Equation (1).

$$t_{next} = t_1 + \frac{t_1 - t_0}{fcnt_1 - fcnt_0} \quad (1)$$

Nodes typically also do not change their SF frequently, but rather try to minimize their energy requirements by selecting the minimal SF that still suffices the link quality requirements. LoRaWAN's built-in feature, Adaptive data rate (ADR), automates this procedure by periodically checking the link margin and commanding the most suitable SF to the end device, which ensures sufficient link margin but minimizes airtime [5]. As a result, the SF typically stays constant throughout the deployment time and barely changes unless significant link quality changes occur. Based on these assumptions, the server regularly, i.e., once every 100 uplink frames, sends downlink messages to all nodes within reach via the gateways, to inform them about their neighbors and their transmission behavior. Regular LoRaWAN frames are used to this end, as visualized in Figure 2, where the information about neighboring nodes is embedded in the *FRMPayload* field. For each neighbor, it contains the 32-bits long device address (DevAddr), 4 bits for the last used data rate (DR), i.e., SF and Bandwidth (BW) setting, and 18 bits for both, the next expected transmission slot (NextTX) and the transmission interval (IntTX) in seconds. Additionally, an 8-bits long field (NextTXFrag) is used for the fragments of a second for the next expected transmission slot, allowing for finer quantization of approximately 3.9ms. In total, 80 bits are used per device. The timestamp for the NextTX is relative and calculated based on the time of a node's uplink transmission. For example, if a node sends an uplink frame at  $t_0$ , and another node is expected to send an uplink frame at  $t_1$ , the server sets the value for NextTX to  $t_1 - t_0$ . This comes with two advantages. First, the size

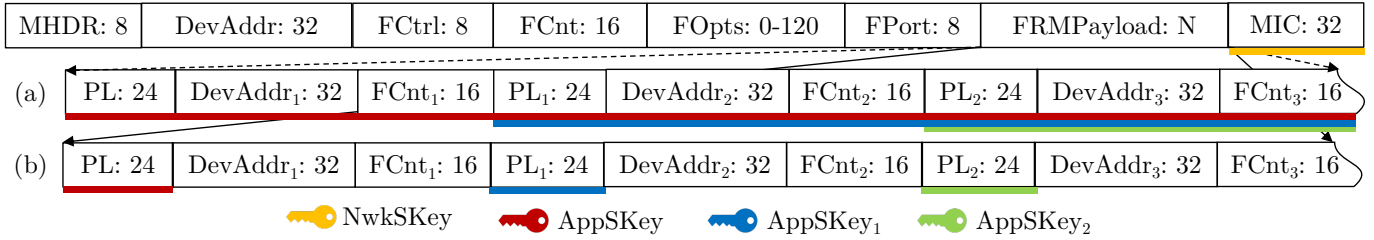


Fig. 4: *Backpack-LoRa* nodes embed overheard LoRaWAN frames [5] in their own *FRMPayload* field. Instead of encrypting the entire *FRMPayload* (subfigure (a)), only their own payload is encrypted, keeping the original encryption of the received frames but leaving the device addresses and frame counters unencrypted (subfigure (b)). The visualization is shown exemplary for a payload size of 24 bits.

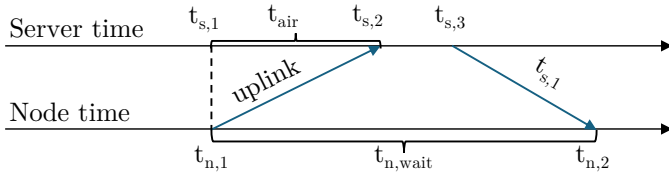


Fig. 5: *Backpack-LoRa*'s application layer time synchronization protocol.

of relative timestamps is typically significant smaller as for absolute timestamps, e.g., Coordinated Universal Time (UTC), and second, no precise knowledge of the absolute time is required. Using this information, nodes can foresee when their neighbors will send their uplinks and which SF and BW they will most likely use.

### C. Pseudo-random Channel Selection

To address the limitations of commodity transceivers, of only being capable to listen on a single channel, we propose a pseudo-random channel allocation algorithm. The algorithm is identical between all devices and its seed value is computed from the epoch time  $e$  and device address  $d$ , as calculated in Equation (2).

$$seed(d, e) = \left( d + \left\lfloor \frac{e}{60000} \right\rfloor \right)^2 \quad (2)$$

As the epoch is given in milliseconds, the seed will be changed every minute and the usage of the unique device address reduces the chance of collisions between nodes. Since the number of channels is limited, this will not guarantee a collision-free operation, but reduce the risks as compared to only using the epoch time as input. Starting from version 1.0.3, LoRaWAN offers specific MAC commands, i.e., *DeviceTimeReq* (device time request) and *DeviceTimeAns* (device time answer), which allow to request the GPS epoch time with a worst case accuracy of  $\pm 100$  ms [5]. However, during first evaluations, we found that the accuracy highly depends on the proper configuration of the gateways and accurate internal clocks. This is problematic in public LoRaWAN networks, where no control about contributing gateways existing. If the internal clock of a gateway is incorrect, naturally also the end device will not be able to properly synchronize to a correct absolute time. Even if only a single one out of multiple gateways has an incorrect time, proper time synchronization cannot be guaranteed, as the network server will determine the gateway answering the device time request. Thus, we implemented our

TABLE I: LoRaWAN *FRMPayload* encryption/decryption Block(s).

Offset in [bytes]	0	1	5	6	10	14	15
Size in [bytes]	1	4	1	4	4	1	1
Value	0x01	0x00000000	Dir*	DevAddr	FCnt**	0x00	i

\* 0x00 for uplink, 0x01 for downlink. \*\* Uplink FCnt for uplink, else downlink FCnt.

own time synchronization protocol on application layer, which works analogously to the Network Time Protocol (NTP) and LoRaWAN's implementation. As visualized in Figure 5, an end device and the server store their local time reference at the beginning of an uplink transmission. The server can derive the time based on the reception timestamp, reduced by the airtime of the packet. Regularly, every 1000 uplink frames, the server issues downlink frames via a gateway, marked by a specific application port (*FPort*) value, with its local time reference at  $t_{s,1}$  to the nodes. If a node received this downlink in one of the two receive windows (cf. Figure 1), it corrects its own local time to the received time reference as  $t_{n,2} = t_{s,1} + t_{n,wait}$ . This ensures the same time reference on all end devices and allows to foresee the neighbor's future channel selections.

### D. Data Embedding and Extraction

If a node overhears a neighbor's transmission, it parses the received bytes according to the LoRaWAN header as visualized in the upper part of Figure 4. In the first step it checks whether the device address was previously communicated by the application server as a node of interest. If not, the packet will be discarded. Otherwise, the *FCnt* and *FRMPayload* fields will be extracted. Together with the device address, these fields are then appended to the node's own payload in its *FRMPayload* field and the frame is scheduled for the next transmission slot. The inclusion of these fields is crucial, as the LoRaWAN *FRMPayload* decryption scheme, in addition to the Application Session Key (AppSKey) (which is only known by the application server and the corresponding node), also requires these values as input parameters for the decryption block(s) as visualized in Table I [5]. Our approach also works recursively, meaning that if a node received a frame which already has appended data from a third node, it appends this data, too. Therefore, an adaption in the standard encryption process of LoRaWAN frames is required, because if a node would append overheard payloads directly in its *FRMPayload* field and encrypt the entire field, a third node would no longer be able to detect the appended frame, as it would appear encrypted (cf. Figure 4 (a)). Thus, *Backpack-LoRa* first encrypts

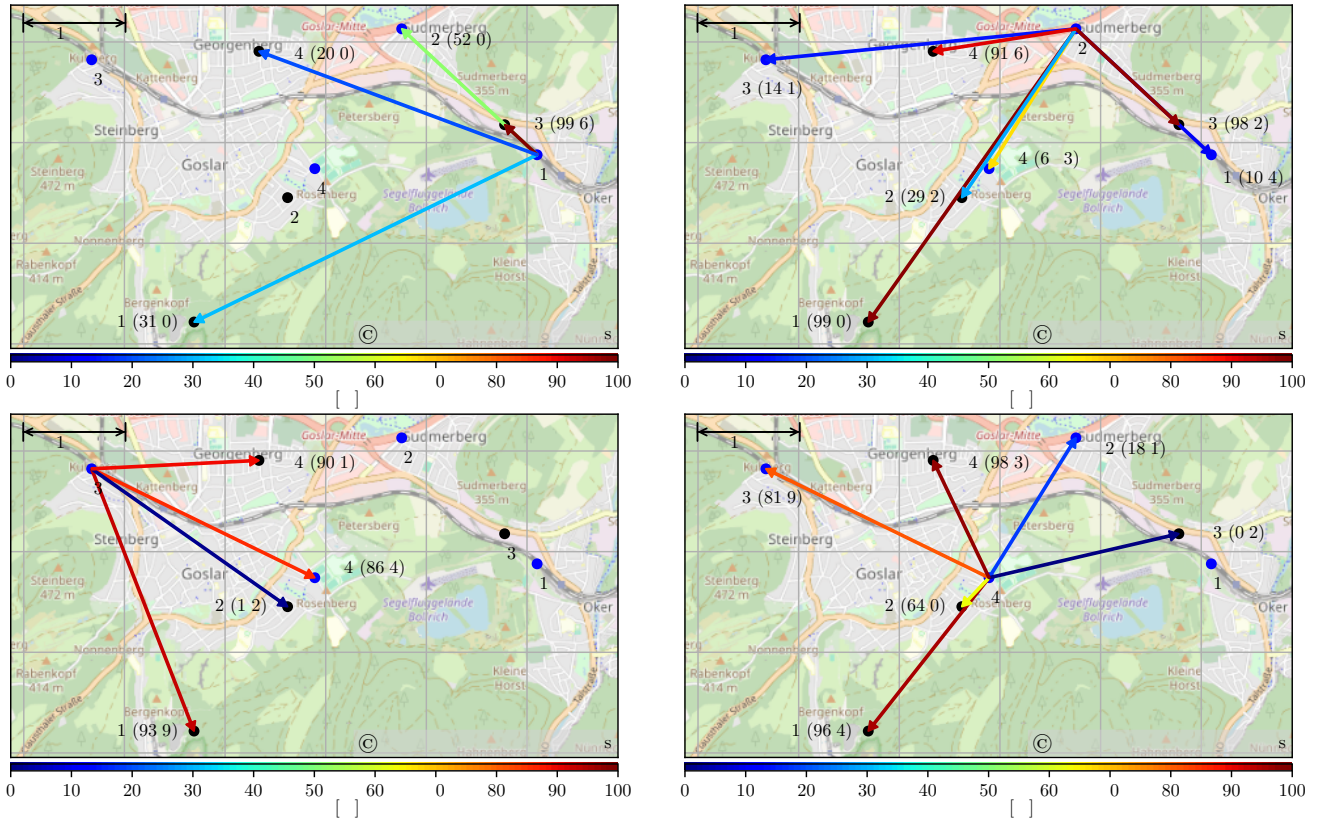


Fig. 6: Topographical visualization of the deployment locations of nodes and gateways. The figure includes the baseline PRRs as obtained when using standard single-hop LoRaWAN for each of the four nodes. Gateways are highlighted by black circles, whereas nodes are marked by blue ones. Missing links indicate that no packet was received. The grid spacing is approximately 1 km.

TABLE II: Baseline link quality parameters for all nodes as received by the gateways and neighboring nodes.

G/N	Node N1			Node N2			Node N3			Node N4																		
	PRR	RSSI in [dBm]	SNR in [dB]	PRR	RSSI in [dBm]	SNR in [dB]	PRR	RSSI in [dBm]	SNR in [dB]	PRR	RSSI in [dBm]	SNR in [dB]																
G1	31.0	-134	-108	-126.8	-20.0	-9.5	-14.7	99.0	-119	-91	-102.9	-12.8	8.5	4.2	93.9	-132	-93	-115.1	-19.3	5.3	-4.4	96.4	-128	-93	-110.0	-17.3	6.0	-1.2
G2	-	-	-	-	-	-	-	29.2	-141	-132	-136.7	-20.5	-13.2	-17.5	1.2	-140	-137	-138.8	-20.2	-18.0	-19.3	64.0	-141	-115	-134.4	-21.2	-9.2	-15.5
G3	99.6	-90	-69	-75.8	3.0	10.8	7.5	98.2	-107	-93	-99.9	-18.0	3.8	-3.1	-	-	-	-	-	-	-	0.2	-104	-104	-104.0	-21.5	-21.5	-21.5
G4	20.0	-141	-112	-134.8	-19.8	-11.2	-15.7	91.6	-134	-89	-120.5	-17.5	3.2	-3.8	90.1	-141	-84	-122.7	-17.2	3.2	-5.2	98.3	-130	-72	-114.5	-11.0	7.0	1.3
N1	-	-	-	-	-	-	-	10.4	-119	-112	-116.2	-21.5	-16.8	-19.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
N2	52.0	-119	-112	-117.9	-21.2	-12.8	-17.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	18.1	-118	-114	-117.2	-21.8	-13.5	-18.1
N3	-	-	-	-	-	-	-	14.1	-123	-120	-122.0	-21.2	-15.5	-18.5	-	-	-	-	-	-	-	81.9	-122	-113	-121.0	-20.8	-4.5	-12.3
N4	-	-	-	-	-	-	-	67.3	-123	-115	-122.0	-21.0	-7.2	-15.4	86.4	-123	-116	-121.1	-21.0	-4.8	-12.2	-	-	-	-	-	-	-

the own payload in the *FRMPayload* field and afterwards injects the appended data. This keeps the appended payloads encrypted with the corresponding AppSKeys but the appended device addresses and frame counters unencrypted (cf. Figure 4 (b)), allowing to identify and append data over multiple hops. However, based on the device address and frame counter, *Backpack-LoRa* can recognize already forwarded frames and prevent routing loops. On the application server side, the appended fields are extracted and the payload is decrypted using the corresponding blocks (cf. Table I) and AppSKeys.

#### IV. EVALUATION

In order to evaluate our approach, we implemented *Backpack-LoRa* on commodity hardware, based on *Arduino MKR WAN 1310* single board computers [28], equipped with *ARM SAMD21* microcontrollers [29] and *Semtech SX1276*

LoRa transceivers [27]. To be compliant with the LoRaWAN specification, we configured  $SF = 12$ ,  $BW = 125$  kHz, Error correction rate (CR) =  $4/5$  and enabled cyclic redundancy check (CRC). Four of these devices were deployed in an urban environment spanning over a distance of approximately 4.5 km x 1.5 km. We also used four gateways for our analysis. The locations of nodes and gateways are visualized in Figure 6. For a complete evaluation, we used two metrics: (1) The change of the PRR per gateway, and (2) the energy overhead introduced by *Backpack-LoRa*.

##### A. Packet Reception Rate

First, we determined the packet reception rates (PRRs) to assess the network performance with and without *Backpack-LoRa*. Therefore, we analyzed the received metadata of the network server which includes all gateways which have re-

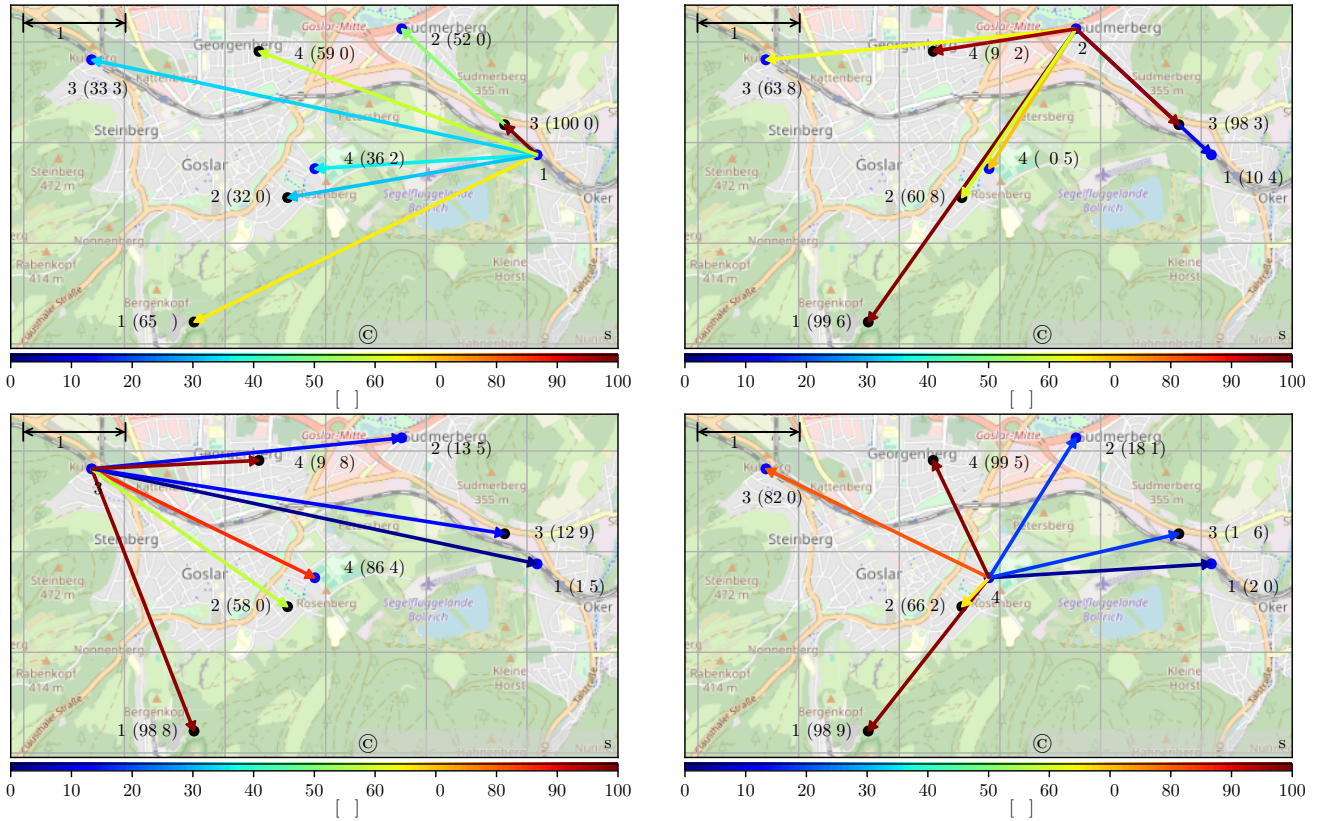


Fig. 7: Connectivity map as achieved by *Backpack-LoRa* with up to 4-hops. Gateways are highlighted by black circles, whereas nodes are marked by blue ones. Missing links indicate that no packet was received. The grid spacing is approximately 1 km.

TABLE III: PRRs and their improvements of the links between all nodes and gateways with different numbers of maximal hops.

G/N	Node N1				Node N2				Node N3				Node N4			
	1 hop	2 hop	3 hop	4 hop	1 hop	2 hop	3 hop	4 hop	1 hop	2 hop	3 hop	4 hop	1 hop	2 hop	3 hop	4 hop
G1	31.0	65.5 (+34.5)	65.6 (+0.1)	65.7 (+0.1)	99.0	99.4 (+0.4)	99.6 (+0.2)	99.6 (+0.0)	93.9	98.8 (+4.9)	98.8 (+0.0)	98.8 (+0.0)	96.4	98.9 (+2.5)	98.9 (+0.0)	98.9 (+0.0)
G2	-	15.6 (+15.6)	31.3 (+15.7)	32.0 (+0.7)	29.2	59.9 (+30.7)	60.6 (+0.7)	60.8 (+0.2)	1.2	56.1 (+54.9)	58.0 (+1.9)	58.0 (+0.0)	64.0	66.2 (+2.2)	66.2 (+0.0)	66.2 (+0.0)
G3	99.6	100.0 (+0.4)	100.0 (+0.0)	100.0 (+0.0)	98.2	98.3 (+0.1)	98.3 (+0.0)	98.3 (+0.0)	-	0.1 (+0.1)	12.7 (+12.6)	12.9 (+0.2)	0.2	17.5 (+17.3)	17.6 (+0.1)	17.6 (+0.0)
G4	20.0	56.9 (+36.9)	59.0 (+2.1)	59.0 (+0.0)	91.6	97.2 (+5.6)	97.2 (+0.0)	97.2 (+0.0)	90.1	97.8 (+7.7)	97.8 (+0.0)	97.8 (+0.0)	98.3	99.5 (+1.2)	99.5 (+0.0)	99.5 (+0.0)
N1	-	-	-	-	10.4	10.4 (+0.0)	10.4 (+0.0)	10.4 (+0.0)	-	-	1.5 (+1.5)	1.5 (+0.0)	-	2.0 (+2.0)	2.0 (+0.0)	2.0 (+0.0)
N2	52.0	52.0 (+0.0)	52.0 (+0.0)	52.0 (+0.0)	-	-	-	-	-	13.5 (+13.5)	13.5 (+0.0)	13.5 (+0.0)	18.1	18.1 (+0.0)	18.1 (+0.0)	18.1 (+0.0)
N3	-	7.9 (+7.9)	33.3 (+25.4)	33.3 (+0.0)	14.1	63.8 (+49.7)	63.8 (+0.0)	63.8 (+0.0)	-	-	-	-	81.9	82.0 (+0.1)	82.0 (+0.0)	82.0 (+0.0)
N4	-	33.5 (+33.5)	36.2 (+2.7)	36.2 (+0.0)	67.3	70.5 (+3.2)	70.5 (+0.0)	70.5 (+0.0)	86.4	86.4 (+0.0)	86.4 (+0.0)	86.4 (+0.0)	-	-	-	-

ceived a certain packet. This allows us to easily calculate the PRR per gateway and create a connectivity map. Additionally, to gather connectivity information between nodes, the nodes were programmed to regularly, i.e., every 100 uplinks, send an additional frame which contains link statistics for nodes from which at least one packet has been overheard, including PRRs as well as extremal and average values for Received Signal Strength Indication (RSSI) and signal-to-noise ratio (SNR) as reported by the LoRa transceiver. Figure 6 visualizes the baseline PRRs for all links from nodes to gateways and nodes to nodes. The figure only considers single-hop links, i.e., as in the standard LoRaWAN approach and serves as the baseline for the subsequent evaluation. The numerical values for PRR, RSSI and SNR are given in Table II. As it can be seen, the PRRs vary strongly over the deployment. For example, data from node N1 is neither received from nodes N3 and N4 nor from gateway G2. However, it has a functional link to node N2 with a PRR of 52% and e.g., to

gateway G3, with a PRR of 99.6%. On the contrary, node N2 can communicate with all nodes and gateways with PRRs ranging from 10.4% to 99%. We can further observe a strong directionality between multiple links. For example, the link  $N1 \rightarrow N2$  shows a PRR of 52%, whereas the reverse link, i.e.,  $N2 \rightarrow N1$  can only deliver 10.4% of the packets. By analyzing the RSSI and SNR values, we can observe that node N1 receives packets from node N2 with a significantly lower mean SNR (-19.2 dB vs. -17.7 dB), i.e., 1.5 dB lower, indicating disadvantageous receiving or ambient conditions, which may be caused by e.g., the presence of increased noise floor at the deployment location or manufacturing tolerances. On the opposite, the links between nodes N3 and N4 show very similar performance in terms of PRRs and both RSSI and SNR, indicating comparable ambient conditions at their locations or less manufacturing tolerances. In contrast, the connectivity map as achieved by *Backpack-LoRa*, is visualized in Figure 7 and the numerical results, considering differ-

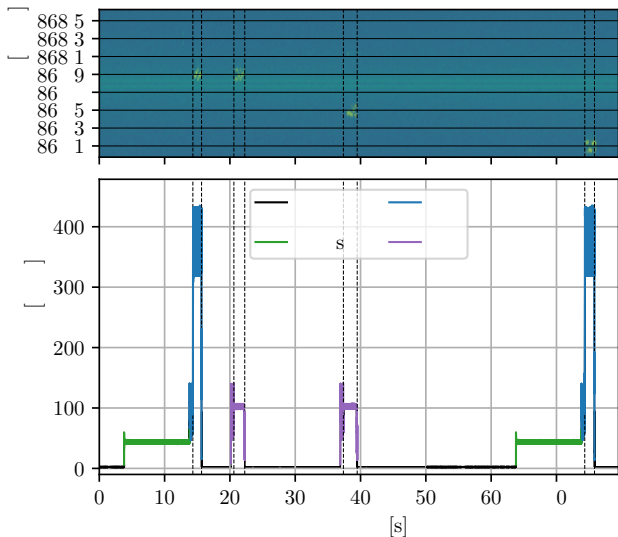


Fig. 8: Illustration of the wireless channel activity (upper plot) and power consumption of the node (lower plot).

ent numbers of hops are given in Table III. We can see significant improvements of the connectivity. For example, with *Backpack-LoRa*, node N1 is able to reach gateway G2 and node N3 is able to reach gateway G3, which was not possible with standard LoRaWAN. Furthermore, various links experience a strong increase in PRR, e.g.,  $N3 \rightarrow G2$ , which rises from 1.2% when considering single-hop communication to 56.1% in case of 2-hops, an increase of 54.9%. The highest benefits are obtained, as expected, by extending LoRaWAN by only a single hop. 3- or 4-hops only barely improve PRRs. Only few exception exist, e.g., for the link  $N1 \rightarrow G2$ , which also significantly benefits from 3-hops over 2-hops, leading to an additional increase in PRR of 15.7%. Overall, *Backpack-LoRa* offers significant improvements of PRRs as compared to standard LoRaWAN of up to 54.9% and allows to reach gateways which were previously unreachable.

### B. Energy Overhead

In order to quantify the energy overhead, we have derived an energy model and analyzed a node running *Backpack-LoRa* using a *Nordic Semiconductor Power Profiler Kit II* [30]. Additionally, we also captured the activity on the wireless channel using an *Ettus research USRP b205mini-i* software defined radio (SDR) [31]. The results thereof are shown exemplary in Figure 8. We classified four different event types for the device, including: *measure*, during which the node collects data for transmission, *TX* and *RX* during which the node transmits and receives data using the LoRa transceiver and *sleep*, during which most components are turned off to save energy. The plots show an excerpt, where the node sends data to the application server (at 13 s) and receives a downlink frame containing neighbor information (at 20 s). As defined in the *EU868* regional parameters, the downlink in the *RX1* window is sent at the same channel as the previous uplink [10]. Subsequently, the node begins to listen on the determined channel (cf. Section III, at 38 s) to overhear a

TABLE IV: Energy overhead in [mJ] of *Backpack-LoRa* per device.

Energy [mJ]	LoRaWAN		Backpack-LoRa		
	single uplink	re-transmission	2 nodes	3 nodes	4 nodes
Overhearing	0.0	0.0	240.9	549.0	924.1
LoRaWAN TX	534.8	1069.6	596.8 (-44.2%)	720.6 (-32.6%)	844.5 (-21.0%)
LoRaWAN RX	178.2	356.4	178.2 (-50.0%)	178.2 (-50.0%)	178.2 (-50.0%)
Total	713.0	1426.0	1015.9 (-28.8%)	1447.8 (+1.5%)	1946.8 (+36.5%)

With a mean power draw of  $p_{tx} = 378.0$  mW for TX and  $p_{rx} = 102.4$  mW for RX as well as mean static mode switching overheads of  $e_{tx,s} = 36.3$  mJ for TX and  $e_{rx,s} = 37.9$  mJ for RX.

neighbor's transmission. Thereby we include a guard time of 500 ms around the expected transmission time to compensate for clock drifts. The node extracts the relevant content of the overheard packet and stores it for its next scheduled uplink. During the next transmission time slot (at 73 s) the node sends both its own just measured data value as well as the appended data. The transmission time can be calculated based on the symbol duration  $t_{sym} = \frac{2^{SF}}{BW}$  and number of symbols  $n_{sym}$  for a given payload size  $n_{pl}$  (in bytes) [27]. For a network of  $m$  nodes, the total LoRaWAN frame size consist of 13 bytes for the LoRaWAN header, 3 bytes payload and  $(m - 1)$  times appended neighbor's data: 4 bytes for device address, 2 bytes frame counter and 3 payload bytes. For  $SF = 12$ ,  $BW = 125$  kHz,  $CR = 4/5$ , enabled CRC and explicit header mode, this is shown in Equations (3) to (5) [27].

$$n_{pl}(m) = 16 + (m - 1) * 9, m \in [1, 2, 3, \dots] \quad (3)$$

$$n_{sym}(n_{pl}) = 12.25 + 8 + \left\lceil \frac{8 * n_{pl} - 4}{40} \right\rceil * 5 \quad (4)$$

$$t_{air}(m) = n_{sym}(n_{pl}(m)) * t_{sym} \quad (5)$$

To quantify the energy overhead required for different numbers of nodes, we measured the individual durations of the different events and multiplied them with their power consumption ( $p_{tx}$  for TX,  $p_{rx}$  for RX). We analyzed 10 consecutive cycles consisting of overhearing, transmitting and receiving (during LoRaWAN's *RX1* and *RX2*). As the energy consumption during sensor measurements and processing on the micro-controller is highly application specific, we focused on the transceiver activities only. Thereby, the energy consumption of each event consist of a static component ( $e_{tx,s}$  for TX,  $e_{rx,s}$  for RX) including mode switching and e.g., radio turn on/off time, and a dynamic component, based on the length of transmission or receive windows. Thus, assuming a duration of 500 ms for each of the *RX1* and *RX2* windows, a *Backpack-LoRa* network with  $m$  nodes, where each node can overhear the others, and a guard time of 500 ms, the energy overhead per device can be derived as shown in Equation (6).

$$e_{total}(m) = e_{tx,s} + t_{air}(m) * p_{tx} + 2 * (e_{rx,s} + 0.5 s * p_{rx}) + \sum_{i=1}^{m-1} e_{rx,s} + (0.5 s + t_{air}(m)) * p_{rx} \quad (6)$$

Table IV shows the mean values of the individual event types as well as the energy consumption for different numbers of nodes as compared to LoRaWAN's standard single-hop approach. The baseline, i.e., a single-hop LoRaWAN transmission and listening during *RX1* and *RX2*, energy requirement

comes at a total of 713.0 mJ, consisting of 534.8 mJ for transmission and 178.2 mJ for listening. If redundant transmissions are used in a network to increase the PRR, a single re-transmission naturally comes at the same cost, i.e., a total of 1426.0 mJ. *Backpack-LoRa* requires 1015.9 mJ per node for a network of 2 nodes. As compared to re-transmissions, this is a reduction of 410.1 mJ or 28.8%, which comes from the fact, that *Backpack-LoRa* only forwards selective parts of the LoRaWAN header instead of the entire frame. The energy costs per device for a *Backpack-LoRa* network of 3 and 4 nodes are 1447.8 mJ and 1946.8 mJ, respectively. Overall, *Backpack-LoRa* requires up to 28.8% less energy per device as compared to re-transmissions and thus offers higher energy-efficiency to the network.

## V. CONCLUSION AND OUTLOOK

In this paper, we have introduced *Backpack-LoRa*, a multi-hop extension for LoRaWANs. Instead of simply forwarding overheard frames in their entirety, our approach extracts the most relevant information from the received LoRaWAN frames, appends them to the node's own payload and schedules it for the next upcoming uplink transmission. Based on the LoRaWAN specification, we have determined the required information that needs to be retained in forwarded frames, such that the application server can decrypt received frames, while reducing the header information to a minimum. We have implemented *Backpack-LoRa* on commodity hardware and evaluated it in a real outdoor deployment. Our results show that our approach enables uplink communication with gateways which were out of reach with standard LoRaWAN, and increases the PRRs by up to 54.9%. Furthermore, we derived an energy consumption model to quantify the energy overhead of *Backpack-LoRa* and showed that our approach requires up to 28.8% less energy as compared to a simple re-transmission approach.

## ACKNOWLEDGMENT

This work was supported by the German Federal Ministry of Education and Research (BMBF) within the scope of the project EXDIMUM.

## REFERENCES

- [1] O. Iova, A. L. Murphy, G. P. Picco, L. Ghio, D. Molteni, F. Ossi, F. Cagnacci *et al.*, "LoRa from the City to the Mountains: Exploration of Hardware and Environmental Factors," in *Int. Conference on Embedded Wireless Systems and Networks*, 2017, pp. 317–322.
- [2] N. Souza Bezerra, C. Åhlund, S. Saguna, and V. A. de Sousa Jr, "Temperature Impact in LoRaWAN — A Case Study in Northern Sweden," *Sensors*, vol. 19, no. 20, p. 4414, 2019.
- [3] L. Parri, S. Parrino, G. Peruzzi, and A. Pozzebon, "Offshore LoRaWAN Networking: Transmission Performances Analysis Under Different Environmental Conditions," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–10, 2020.
- [4] C. A. Boano, M. Cattani, and K. Römer, "Impact of Temperature Variations on the Reliability of LoRa - An Experimental Evaluation," in *7th ACM International Conference on Sensor Networks*, 2018, pp. 39–50.
- [5] LoRa Alliance, Inc., *LoRaWAN 1.0.3 Specification*, 2018.
- [6] D. Lundell, A. Hedberg, C. Nyberg, and E. Fitzgerald, "A Routing Protocol for LoRa Mesh Networks," in *19th IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks*, 2018, pp. 14–19.
- [7] S. Kim, H. Lee, and S. Jeon, "An Adaptive Spreading Factor Selection Scheme for a Single Channel LoRa Modem," *Sensors*, vol. 20, no. 4, p. 1008, 2020.
- [8] M. N. Ochoa, A. Guizar, M. Maman, and A. Duda, "Evaluating LoRa Energy Efficiency for Adaptive Networks: From Star to Mesh Topologies," in *13th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, 2017, pp. 1–8.
- [9] W. Joosen, D. Hughes *et al.*, "Smart-Hop: Low-Latency Multi-hop Networking for LoRa," in *18th IEEE International Conference on Distributed Computing in Sensor Systems*, 2022, pp. 17–20.
- [10] LoRa Alliance, Inc., *LoRaWAN 1.0.3 Regional Parameters*, 2018.
- [11] C.-H. Liao, G. Zhu, D. Kuwabara, M. Suzuki, and H. Morikawa, "Multi-Hop LoRa Networks Enabled by Concurrent Transmission," *IEEE Access*, vol. 5, pp. 21 430–21 446, 2017.
- [12] H.-C. Lee and K.-H. Ke, "Monitoring of Large-Area IoT Sensors Using a LoRa Wireless Mesh Network System: Design and Evaluation," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 9, pp. 2177–2187, 2018.
- [13] A. Abrardo and A. Pozzebon, "A Multi-Hop LoRa Linear Sensor Network for the Monitoring of Underground Environments: The Case of the Medieval Aqueducts in Siena, Italy," *Sensors*, vol. 19, no. 2, p. 402, 2019.
- [14] B. Sartori, S. Thielemans, M. Bezunartea, A. Braeken, and K. Steenhaut, "Enabling RPL multihop communications based on LoRa," in *13th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, 2017, pp. 1–8.
- [15] S. Pang, J. Lu, R. Pan, H. Wang, X. Wang, Z. Ye, and J. Feng, "Optimizing Routing Protocol Design for Long-Range Distributed Multi-Hop Networks," *Electronics*, vol. 13, no. 19, p. 3957, 2024.
- [16] X. Jiang, H. Zhang, E. A. B. Yi, N. Raghunathan, C. Mousoulis, S. Chaterji, D. Peroulis, A. Shakouri, and S. Bagchi, "Hybrid Low-Power Wide-Area Mesh Network for IoT Applications," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 901–915, 2020.
- [17] H. Huh and J. Y. Kim, "LoRa-based Mesh Network for IoT Applications," in *5th IEEE World Forum on Internet of Things*, 2019, pp. 524–527.
- [18] K.-H. Ke, Q.-W. Liang, G.-J. Zeng, J.-H. Lin, and H.-C. Lee, "Demo Abstract: A LoRa Wireless Mesh Networking Module for Campus-Scale Monitoring," in *16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2017, pp. 259–260.
- [19] B. Zhang, V. Srinivas, Y. Zhang, and R. P. Dick, "LoRa Synchronized Energy-Efficient LPWAN," in *IEEE International Conference on Communications*, 2023, pp. 5383–5389.
- [20] J. Dias and A. Grilo, "LoRaWAN multi-hop uplink extension," *Procedia computer science*, vol. 130, pp. 424–431, 2018.
- [21] E. Sisinni, P. Ferrari, D. F. Carvalho, S. Rinaldi, P. Marco, A. Flammini, and A. Depari, "LoRaWAN Range Extender for Industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5607–5616, 2019.
- [22] P. Tian, C. A. Boano, X. Ma, and J. Wei, "LoRaHop: Multihop Support for LoRaWAN Uplink and Downlink Messaging," *IEEE Internet of Things Journal*, vol. 10, no. 17, pp. 15 376–15 392, 2023.
- [23] P. Tian, X. Ma, C. A. Boano, Y. Liu, F. Yang, X. Tian, D. Li, and J. Wei, "ChirpBox: An Infrastructure-Less LoRa Testbed," in *Int. Conference on Embedded Wireless Systems and Networks*, 2021, pp. 115–126.
- [24] C. Ebi, F. Schaltegger, A. Rüst, and F. Blumensaat, "Synchronous LoRa Mesh Network to Monitor Processes in Underground Infrastructure," *IEEE access*, vol. 7, pp. 57 663–57 677, 2019.
- [25] M. S. Aslam, A. Khan, A. Atif, S. A. Hassan, A. Mahmood, H. K. Qureshi, and M. Gidlund, "Exploring Multi-Hop LoRa for Green Smart Cities," *IEEE Network*, vol. 34, no. 2, pp. 225–231, 2019.
- [26] S. Feng, J. Chen, and Z. Zhao, "Cost Effective Routing in Large-scale Multi-hop LoRa Networks," in *IEEE Conference on Computer Communications Workshops*, 2022, pp. 1–6.
- [27] Semtech Corporation, *SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver*, 4th ed., 2015.
- [28] Arduino, *MKR WAN 1310*.
- [29] Microchip Technology Inc., *SAM D21/DA1 Family - Low-Power, 32-bit Cortex-M0+ MCU with Advanced Analog and PWM*, f ed., 2020.
- [30] Nordic Semiconductor, *Power Profiler Kit II Current measurement tool for embedded development*, 1st ed., 2020.
- [31] Ettus Research, *USRP B200mini, Product Overview, Features*, 2022.