LoRaHop: Multi-Hop Support for LoRaWAN Uplink and Downlink Messaging

Pei Tian, *Graduate Student Member, IEEE*, Carlo Alberto Boano, *Member, IEEE*, Xiaoyuan Ma, *Member, IEEE*, and Jianming Wei, *Member, IEEE*

***Abstract*—LoRaWAN is one of the most popular protocols to build low-power wide area networks. Unfortunately, it adopts a star topology, which limits network coverage and may also cause an unnecessary decrease in energy efficiency as well as scalability. In fact, end-devices that are deployed far away from a gateway need to increase their transmission power or spreading factor to sustain reliable communications, which increases their energy expenditure as well as the size of the collision domain. The only alternative is the deployment of additional gateways or dedicated relay nodes, which results in higher costs and deployment efforts. In this work, we introduce LoRaHop, an extension of LoRaWAN that enriches end-devices with the ability to form a mesh network and to seamlessly relay packets to/from a gateway, thereby providing LoRaWAN networks with multi-hop support for both uplink and downlink messaging. LoRaHop leverages concurrent transmissions to enable a reliable and efficient data collection or dissemination over the mesh network, as well as to simplify network formation. Furthermore, LoRaHop embeds a mechanism that simplifies rendezvous across devices and that minimizes the impact of mesh operations on existing LoRaWAN transmissions. We implement LoRaHop on off-the-shelf LoRa end-devices (ensuring their interoperability with commercial LoRaWAN gateways and network servers), and evaluate its performance on an outdoor testbed. Our results show that LoRaHop can effectively extend the coverage of a LoRaWAN network while improving reliability by up to 98.33% and reducing energy consumption by up to 48.02%. Our findings further demonstrate that using LoRaHop to create a multi-hop LoRaWAN network that communicates using low spreading factors brings significant benefits in terms of energy efficiency and scalability compared to the use of a single-hop LoRaWAN network using high spreading factors.**

***Index Terms*—ChirpBox, Efficiency, LoRaDisC, LoRaWAN compatibility, Mesh network, Network coverage, Reliability.**

I. Introduction

Low-Power Wide-Area Network (LPWAN) technologies are increasingly used to develop Internet of Things (IoT) systems that require the deployment of wireless sensors and actuators across large geographical areas. The use of LPWANs is especially promising in application areas such as precision farming [1], water management [2], environmental monitoring [3], and smart cities [4], which benefit from the interconnection of cheap low-power devices on a large scale.

This work is partially funded by the Special Fund for Basic Research on Scientific Instruments of the National Natural Science Foundation of China (No. 51827814), the Science and Technology Innovation Plan of Shanghai Science and Technology Commission (20DZ1201002), and the China Scholarship Council (CSC). This work is also partially supported by the TU Graz LEAD project "Dependable IoT in Adverse Environments". *(Corresponding author: Jianming Wei.)*

Pei Tian and Jianming Wei are associated with the Shanghai Advanced Research Institute, Chinese Academy of Sciences, China, as well as the University of Chinese Academy of Sciences, China (e-mail: tianpei2018@sari.ac.cn; wjm@sari.ac.cn).

Carlo Alberto Boano is with the Institute of Technical Informatics, Graz University of Technology, Austria (e-mail: cboano@tugraz.at).

Xiaoyuan Ma is with Svenska Kullagerfabriken (SKF) Group, China (e-mail: ma.xiaoyuan.mail@gmail.com).

Among LPWAN technologies, LoRa is one of the most popular and widespread [5], as it uses unlicensed spectrum and hence does not require subscriptions nor suffers limitations on data traffic (in contrast to technologies such as NB-IoT, LTE-M, and Sigfox). Moreover, users have full control of the network infrastructure and can enjoy the rich ecosystem supported by the open LoRa Alliance community, which provides end-to-end integration of hardware and software components.

One of these components is LoRaWAN [6], a reference architecture and medium access control protocol that specifies a star-of-stars topology in which gateways relay messages between end-devices and a central network server. Hence, in LoRaWAN, end-devices interact exclusively with gateways, which enable their connection to a LoRaWAN network and to the Internet. For LoRaWAN class A end-devices[1], the communication to a gateway is asynchronous and follows a pure ALOHA scheme, *i.e.,* an end-device can transmit a message at an arbitrary point in time and then switch to low-power mode to preserve its limited energy budget. Thanks to its simplicity and open-source availability, LoRaWAN has grown on a global scale, with tens of thousands gateways deployed just within The Things Network initiative [7].

LoRaWAN limitations w.r.t. coverage and scalability. The ease of connecting end-devices to LoRaWAN gateways through a simple star topology and medium access control (MAC) scheme, however, comes with a number of limitations. In fact, to be able to join a LoRaWAN network, each end-device needs to be within reach of at least one gateway. Unfortunately, although LoRa's communication range is often advertised in the order of (tens of) kilometers, it is much lower in real-world deployments, as signal attenuation and distortion as well as environmental factors and urban canyons strongly affect the link quality [8]–[10]. Although one can simply add more gateways near the end-devices that are disconnected or exhibit an unreliable connectivity, this drastically increases costs (starting from roughly a hundred dollars per gateway [11]) as well as deployment effort, and does not scale with the magnitude of devices that are expected to be installed in the near future. Moreover, a gateway deployment may not be possible in disaster scenarios as well as in remote areas, where a constant power supply and reliable network connectivity cannot be guaranteed. An alternative is to increase the communication range of LoRa devices by using a higher spreading factor (SF), *i.e.,* by increasing the receiver's sensitivity at the cost of a longer packet airtime, or by using a higher transmission (TX) power [12]. However, both approaches result in a large increase in *energy consumption*, and

[1]The LoRaWAN specification defines three classes of end-devices in order to satisfy different application requirements, as detailed in Sect. II-A3.

may still be insufficient to ensure a reliable connectivity with an existing gateway. Besides, by increasing the communication range of end-devices, one also enlarges the size of the *collision domain*, which may strongly affect network performance. Indeed, the use of pure ALOHA, unlicensed spectrum, and single-hop communication over multiple kilometers, results in severe scalability issues [13]–[15], as LoRaWAN devices do not perform channel sensing and hence cannot assess whether the medium is free from ongoing transmissions. This makes LoRaWAN devices susceptible to the activities of other devices operating within the same network (*i.e.,* internal interference) as well as to the activities of any co-located device operating on the very same frequencies (*i.e.,* external interference).

The quest for multi-hop support in LoRaWAN. To tackle all these limitations, one way forward is to enable the formation of *mesh networks*, such that other LoRa end-devices in close proximity can be used to relay information to one or more gateways that were previously unreachable. The creation of multi-hop LoRa networks has attracted a significant amount of research; however, all proposed approaches have severe limitations and drawbacks. Indeed, most of the existing solutions lack compatibility with LoRaWAN [16]–[19], do not support downlink messaging [20], or require the use of custom gateways [21], [22] and extra hardware [23]. Moreover, existing solutions lack scalability and energy efficiency, as they either support only a limited number of hops/devices [22]–[25], require end-devices to be permanently powered [18], [19], or incur a large overhead to establish and maintain routes [16]. Recent works [26]–[28] have proven the feasibility of using *concurrent transmissions* (CT) to build multi-hop LoRa networks, which has the potential to eliminate the overhead of establishing and maintaining routes in classical routing-based approaches [29]. However, to date, the use of CT to build multi-hop LoRaWAN networks has not been investigated yet.

Our contributions. In this work, we are the first to leverage CT to build multi-hop support for LoRaWAN networks for both uplink and downlink messaging. Specifically, we enrich LoRaWAN end-devices with the ability to form a mesh network and to seamlessly relay packets to/from a gateway, as shown in Fig. 1. This way, one can practically extend the range of a LoRaWAN network without the need to deploy additional gateways or change their configuration. In fact, end-devices that are unable to establish a direct connection with existing gateways can leverage other end-devices to forward information to/from a gateway accordingly. We achieve this by developing *LoRaHop*, an add-on protocol compatible with LoRaWAN that makes use of CT and network coding for reliable and efficient data collection/dissemination over a multi-hop mesh network. Specifically, LoRaHop performs several flooding rounds during which end-devices communicate on multiple channels without the need to establish and maintain routes or a network hierarchy: this allows to minimize collisions, maximize throughput, and reduce energy consumption.

LoRaHop also embeds network formation and rendezvous capabilities that allow end-devices to autonomously join a mesh network and to only communicate in absence of other LoRaWAN communications to/from a gateway. To accomplish this, end-devices forming the mesh network exchange information about their radio usage, derive a matrix of radio schedules, and communicate only when everyone's radio is otherwise idle, which helps minimizing the impact of mesh operations on the periodic LoRaWAN transmissions of relay nodes.

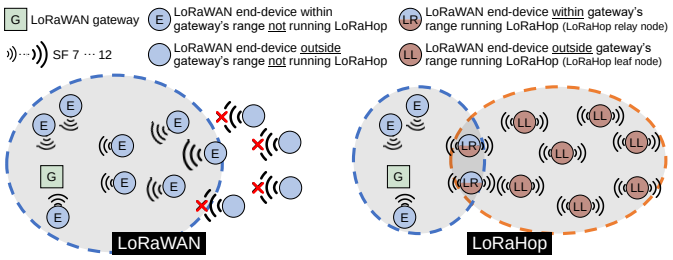


Fig. 1. LoRaHop allows to form multi-hop LoRaWAN networks that can be used to forward information to/from end-devices that are unable to establish a direct connection to a gateway, thereby extending the network coverage while minimizing the SF used for communication.

We further show that multi-hop support for LoRaWAN networks allows not only to extend coverage (*i.e.,* to enable the connection of end-devices that are normally disconnected from a gateway), but also gives the freedom to artificially decrease the TX power or SF in an attempt to reduce the size of the collision domain. As we show in Sect. V, this can significantly minimize energy consumption and internal interference, leading to an increase in both network performance and scalability.

We implement LoRaHop on ChirpBox nodes [28] consisting of an off-the-shelf `STM32L476RG` ultra low-power microcontroller as well as an `SX1276Mb1MAS` LoRa transceiver, and showcase how it allows to form a multi-hop LoRaWAN network connected to a commercial `RAK7243` gateway. We then experimentally evaluate the performance of LoRaHop in terms of energy consumption and reliability on an outdoor LoRa testbed with 21 nodes deployed in the city of Shanghai.

Our results show that the use of LoRaHop allows to effectively extend the coverage of a LoRaWAN network with minimal energy expenditure, allowing – among others – to increase the packet reception rate by up to 98.33% and to reduce the energy consumption by up to 48.02%. Our experimental results further confirm that using LoRaHop to create a multi-hop LoRaWAN network employing a low SF leads to several performance improvements compared to the use of a single-hop network employing a high SF.

This article proceeds as follows. After providing the reader with background information on LoRaWAN and CT, we make the following contributions:

- We present LoRaHop, an add-on protocol enabling support for multi-hop LoRaWAN networks, detailing its architecture and functionality (Sect. III).
- We shed light on LoRaHop's design and implementation, discussing its reliable data collection and dissemination, how it performs an efficient network formation and rendezvous, as well as a number of optimizations increasing its energy efficiency (Sect. IV).
- We evaluate LoRaHop on a real-world testbed and show how its use can increase network coverage, reliability, and energy efficiency (Sect. V).

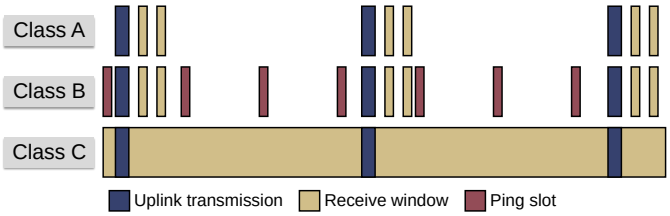


Fig. 2. Uplink and downlink communication for different LoRaWAN device classes. Class A end-devices can enter two receive windows after an uplink transmission. Class B end-devices can receive additional downlink messages from a gateway during scheduled ping slots. Class C end-devices are able to receive downlink messages from a gateway at anytime, except when performing uplink transmissions.

After discussing LoRaHop's limitations and how to further increase its functionality in Sect. VI, we review related work in Sect. VII and conclude the paper in Sect. VIII.

II. Background

Before detailing the design of LoRaHop, we first introduce LoRa technology as well as the LoRaWAN protocol, detailing its architecture and the three device classes supported (Sect. II-A). We then describe the principle of CT, which is exploited by LoRaHop to carry out a reliable and efficient data collection/dissemination (Sect. II-B).

A. LoRa and LoRaWAN

1) LoRa Technology: LoRa is a low-power radio technology based on chirp spread spectrum (CSS) modulation that is able to transmit data at distances of up to several kilometers [30]. The transmission range achieved when communicating using LoRa devices mainly depends on the employed SF and SF power. The SF, whose typical values are between 7 and 12, controls the chirp rate and hence how fast data is transmitted. A signal modulated with a higher SF has higher sensitivity and can be received with less errors, but requires much longer to be transmitted: this implies a longer transmission range at a price of a lower data rate and increased energy consumption. A higher TX power improves the transmission range, but results in a higher energy consumption.

2) LoRaWAN Architecture: LoRaWAN is a protocol that operates on top of LoRa and specifies a network structure consisting of end-devices and gateways. Specifically, a LoRaWAN network is based on a star topology where LoRaWAN end-devices are directly connected to a gateway and transmit data following a *pure ALOHA* access scheme, *i.e.,* transmissions occur at arbitrary times whenever data is available. Gateways can receive the uplink transmissions of several end-devices sent on different channels simultaneously, but can only send individual packets to an end-device during a given time-slot.

3) LoRaWAN Device Classes: To satisfy the requirements of a vast range of different applications, LoRaWAN supports three devices classes, as illustrated in Fig. 2:

- *Class A*: This is the default and most popular LoRaWAN configuration, as end-devices can operate with the highest energy efficiency. Uplink packets can be sent by end-devices at any time. End-devices then wait and open up to two short receive windows, which represent the only opportunity to receive downlink packets. After the transmission and receive windows, the end-devices enter sleep mode and preserve their limited energy budget.
- *Class B*: In this configuration, additional receive windows (called "ping slots") are scheduled for downlink packets, and can be used by end-devices to receive data from the gateway more frequently. These extra receive windows (as well as the beacons transmitted from the gateway to provide end-devices with a timing reference) increase the overall energy consumption of the end-devices.
- *Class C*: In this configuration, end-devices keep their radio active in receive mode at all times (*i.e.,* they do not sleep), which provides the lowest possible latency for downlink traffic at the cost of a high energy expenditure.

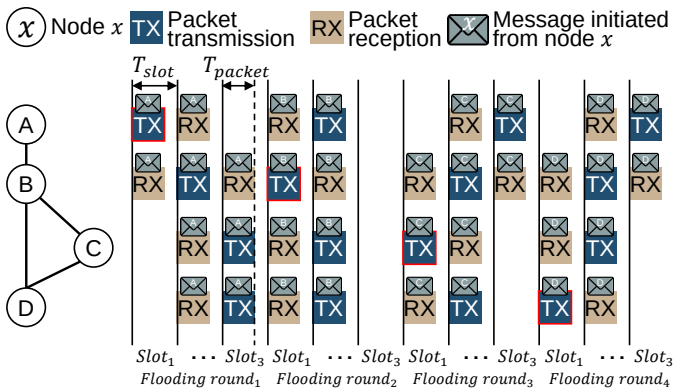


Fig. 3. The concurrent transmissions paradigm is based on the simultaneous transmission of multiple nodes: packets can still be successfully received thanks to capture effect and constructive interference, leading to a powerful primitive for data collection and dissemination within flooding rounds.

Typical LoRaWAN application scenarios are monitoring systems where class A end-devices deployed over large geographical areas *periodically* upload data to the gateway, *e.g.,* temperature [31] and soil moisture [32] data in smart farming applications or water flow and pressure information in smart water distribution networks [33], [34]. Since they are the most widespread, we explicitly consider LoRaWAN class A devices when designing LoRaHop: unless differently specified, we will hence refer to this device class in the remainder of this paper.

4) LoRaWAN Radio Regulations: LoRa devices make use of the freely-available industrial, scientific, and medical (ISM) radio frequency bands. Due to national regulations, the use of these frequencies may vary across countries. For instance, in Europe, each LoRa device must comply with a transmission duty cycle limit of 0.1 or 1% (*i.e.,* devices can transmit data for up to 3.6 or 36 s every hour). The use of techniques such as listen before talk (LBT) and adaptive frequency agility (AFA) can alleviate these restrictions, allowing for a duty cycle limit of 100 s per hour, equivalent to a 2.7% duty cycle. LBT requires each device to scan the channel before transmission: if any interference is detected, the device must back off and wait before attempting the next transmission. AFA requires the device to dynamically switch between multiple channels.

B. Concurrent Transmissions (CT)

Traditionally, simultaneous packet transmissions are considered to be harmful, as they typically cause collisions and result in a drop of reliability. Recently, a new wave of research works

based on the CT paradigm has shown that intentionally letting two or more nodes transmitting a packet within a small delay can lead to a successful packet reception, thanks to the capture effect and constructive interference [29]. Fig. 3 exemplifies the idea behind CT: packets of duration T_{packet} are transmitted in synchronous time-slots of duration $T_{slot} > T_{packet}$. In case two or more neighbors concurrently transmit a packet, one of the transmitted packets can be received and decoded successfully if the incoming signals satisfy certain power and timing conditions (*e.g.,* if a signal is significantly stronger and/or arrives slightly earlier than the others [35]).

CT can hence be exploited to efficiently collect and disseminate data over different flooding rounds. Fig. 3 shows an example of four flooding rounds in which each node disseminates information to all other nodes in the mesh network. Specifically, in the first flooding round, the packet sent by node A in the first time-slot is re-transmitted by node B in the successive time-slot (*i.e.,* the message broadcasted by node B in the second time-slot will be received by all nodes, as they are all neighbours of node B). Nodes C and D re-transmit the packet received from node B in the third time-slot (node A does not, since it has already transmitted this packet at least N_G times before, with $N_G = 1$ in this case), and the dissemination of node A's packet terminates, after reaching all nodes in the network.

Nodes B, C, and D initiate further packet transmissions (circled in red) in the successive flooding rounds, which are progressively disseminated to all other nodes in the network. Note that the number N_{slot} of time-slots in a flooding round ($N_{slot} = 3$ in this example) can be configured as a function of the number of nodes in the network and its diameter.

To increase both throughput and reliability when dealing with large networks and data packets, one can couple CT with *network coding*, as shown in [36]–[38]. The key idea is to transmit combinations of multiple packets rather than transmitting packets as independent units: this allows to deal with packet loss without the need of explicit re-transmissions.

The CT principle has been extensively studied on devices employing the IEEE 802.15.4 narrowband physical layer (PHY) [29], [35], [39]. Its feasibility was later also shown for the ultra-wideband IEEE 802.15.4 PHY [40], as well as for Bluetooth Low Energy [41], [42] and LoRa devices [26], [27]. In particular, the use of CT on top of LoRa radios has been originally showcased by Bor et al. [26], and later analyzed in detail by Liao et al. [27].

III. LoRaHop: Overview

We introduce LoRaHop by describing its architecture (Sect. III-A) and the protocol used to exchange data within the mesh network (Sect. III-B). We further discuss how the nodes in the mesh network agree upon a common schedule for communications (Sect. III-C), as well as how downlink/uplink LoRaWAN traffic is relayed from/to a gateway (Sect. III-D).

A. Architecture

LoRaHop enables the creation of a mesh network extending the range of a LoRaWAN network to nodes that cannot directly communicate with a gateway, as shown in Fig. 1.

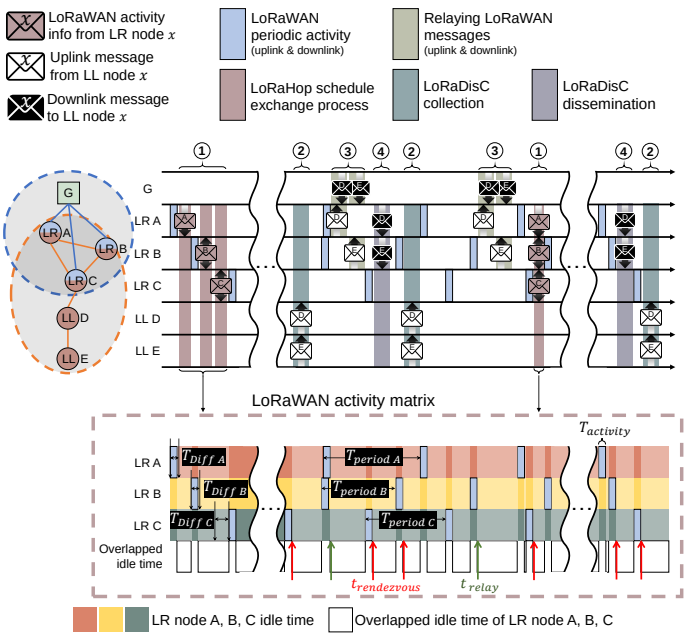


Fig. 4. Overview of LoRaHop's operations in an exemplary network consisting of a gateway, three relay nodes (LR), and two leaf nodes (LL). Using LoRaDisC, CT-based flooding rounds are used to collect/disseminate uplink/downlink LoRaWAN messages from/to LL nodes throughout the mesh network, as well as to derive and exchange the schedule used for mesh communications. LR nodes take care of forwarding messages created by or destined to LL nodes to/from a gateway, giving the latter the illusion of having a direct link with the LL nodes. Note that phases illustrated as ①–④ may consist of single or multiple flooding rounds.

We distinguish between four types of devices: LoRaWAN gateways (G), LoRaHop relay nodes (LR) within a gateway's range, LoRaHop leaf nodes (LL) outside a gateway's range, as well as LoRaWAN end-devices (E) within a gateway's range but not involved in forming the mesh network.

LR nodes can directly interact with a gateway in a reliable way, and act as relay to let LL nodes join the LoRaWAN network as well as to forward their traffic from/to a gateway.

LL nodes are typically too far from a gateway to establish a direct connection, or can only establish a direct connection that is unreliable (*i.e.,* with a high packet loss ratio), and hence need to leverage LR nodes to properly connect to the LoRaWAN network. Moreover, LL nodes may also want to reduce the employed SF to connect to a gateway in an attempt to minimize their energy consumption. As we show in Sect. V, forming a multi-hop mesh network operating at a low SF may introduce several benefits compared to the use of a single-hop network operating at a high SF – among others, a higher energy efficiency and a smaller collision domain.

B. CT-based Data Exchange

Data collection & dissemination with LoRaDisC. To support the exchange of uplink and downlink LoRaWAN traffic over a mesh network, LoRaHop builds on top of *LoRaDisC*, an all-to-all protocol based on CT that we originally designed to orchestrate the operations of an infrastructure-less LoRa testbed [28]. LoRaDisC allows a reliable and efficient data collection and dissemination over multiple hops by operating

over different *flooding rounds* with synchronous slots (such as the one illustrated in Fig. 3). During these flooding rounds, a node can initiate the transmission of a message that is quickly propagated throughout the mesh by having all adjacent nodes re-transmitting the same message concurrently for a given amount of times N_G, as exemplarily shown in Sect. II-B.

Sequence of LoRaHop operations. LoRaHop activities can be divided into four main phases, as illustrated in Fig. 4. First, LR/LL nodes forming the mesh network derive and exchange the schedule used to communicate (*schedule exchange process* ①): this happens during network bootstrap as well as periodically over dedicated LoRaDisC flooding rounds, as detailed in Sect. III-C. LL nodes have then the chance to send LoRaWAN uplink messages destined to the gateway over dedicated LoRaDisC flooding rounds used for data *collection* ②.

One of the LR nodes receiving these uplink messages will then relay them to the gateway ③. When doing so, the LR node uses the address of the LL node, giving the gateway the illusion of having a direct link with the LL node, *i.e.,* gateways are not aware of the existence of a mesh network and of the fact that LR nodes act as relays. The LR node follows the standard LoRaWAN protocol when relaying packets to/from the gateway. Specifically, it opens by default two downlink reception windows at scheduled times (*e.g.,* 1 s and 2 s) after transmitting an uplink packet, as dictated by the LoRaWAN specification. Any downlink message sent by a gateway to the LR node in response to the uplink transmission of an LL node will be circulated in the mesh network via dedicated LoRaDisC flooding rounds used for *dissemination* ④. Hence, LoRaDisC is used to propagate both LoRaWAN uplink and downlink messages generated by or destined to LL nodes over the mesh network.

Reliable and efficient flooding. LoRaDisC uses multiple flooding rounds during which nodes communicate on multiple channels: this allows to minimize collisions, maximize throughput, and speed up the information exchange across nodes. Moreover, LoRaDisC takes advantage of network coding: all devices transmit previously-received packets with random linear network coding (RLNC) as in Mixer [36], a protocol that transmits blended packets concurrently. When a device receives enough combinations of those blended packets, it can successfully decode all packets that were transmitted[2]. LoRaDisC further embeds an *early termination mechanism*, as described in Sect. IV-B, in order to eliminate unnecessary time-slots within the flooding rounds used for data collection.

C. Mesh Network Formation and Schedule Exchange

LoRaHop activities follow a given schedule synchronizing the operations of LL and LR nodes. Such schedule is derived by letting LR nodes flood information about their uplink/downlink LoRaWAN activities (phase ① in Fig. 4).

This allows to derive the LoRaWAN *activity matrix*, which contains information about the planned LoRaWAN activities of each LR node and is used to identify *when* to schedule LoRaDisC flooding rounds for collecting uplink traffic (phase ②) or disseminating downlink traffic (phase ④) from/to the LL nodes. In other words, the LoRaWAN activity matrix can be used to derive the periods of time in which the radio of *all* LR nodes is idle and could hence be reused for mesh communications. This ensures that mesh traffic does not overlap with the periodic interactions between LR nodes and gateways. Moreover, such an activity matrix is also used by LR nodes to derive when to start relaying uplink/downlink messages to/from a gateway (phase ③).

Note that the ability to derive and exchange a common schedule allows not only to prevent interference with the planned LoRaWAN uplink/downlink transmissions of LR nodes, but also to maximize energy efficiency. In fact, as LR and LL nodes know the exact point in time in which each LoRaDisC flooding round begins and terminates, they can turn off their radio in the remaining time accordingly.

Deriving a common schedule. The bottom part of Fig. 4 shows how to construct the LoRaWAN activity matrix (from which the schedule of the various phases ① to ④ is derived).

We take advantage of the fact that interactions between most LoRaWAN class A devices and gateways are periodic or can be modelled as periodic traffic [43], [44]. In fact, LoRaWAN class A devices are typically used to build monitoring systems that report data to a gateway in a cyclic fashion [45]–[47].

We hence let each LR node derive its LoRaWAN activity period by analyzing the timings of its radio usage [48], and disseminate this information through the mesh network in dedicated LoRaDisC flooding rounds (phase ① in Fig. 4). Specifically, each LR node derives and disseminates three values: the duration of an uplink/downlink interaction with a LoRaWAN gateway ($T_{activity}$), the period between consecutive interactions with a gateway (T_{period}), as well as the relative difference in time between the beginning of the last/next gateway interaction and the dissemination of this information (T_{diff}). Each LR/LL node merges this information and derives the time-slots during which *all* LR nodes are not carrying out any LoRaWAN activity. These time-slots are marked in Fig. 4 as "*overlapped idle time*", and can be reused by LR nodes to perform any activity, while being sure not to affect their normal interactions with LoRaWAN gateways. Combining this information with the known duration of a LoRaDisC flooding round, one can then identify sufficiently large time-slots that can be reused for mesh communications (*i.e.,* to schedule a LoRaDisC collection ② or dissemination ④) or for relaying data from/to a gateway ③. The beginning of these time-slots is marked in Fig. 4 as $t_{rendezvous}$ and t_{relay}, respectively, and their exact computation is explained in detail in Sect. IV-A.

Joining a network. Once the mesh network is established, LR nodes periodically flood information about the LoRaWAN activity matrix during dedicated LoRaDisC rounds (phase ① in Fig. 4), so that new LR/LL nodes can join the mesh network. Nodes that are not yet part of the network keep their radio on and listen for incoming messages embedding the activity matrix, from which they derive the LoRaDisC schedule needed to join the mesh network. During network bootstrap, LR nodes

[2]LoRaDisC adds a bit-field named `coding_vector` in the header of each packet. This bit-field contains up to M bits indicating which of the M packets generated from N devices are combined in the current packet. Each device implements Gaussian elimination with all received packets.

start broadcasting their own LoRaWAN activity information during their idle time. LR/LL nodes keep their radio on and listen for such messages, which they re-broadcast. This way, the schedule exchange process is triggered, which allows all nodes in the mesh network to exchange the LoRaWAN activity matrix and derive the LoRaDisC schedule.

D. Relaying LoRaWAN Uplink/Downlink Messages

It is worth emphasizing that the gateway is *not* involved in the mesh communications. The LR nodes first collect and queue all the messages received from LL nodes through the LoRaDisC collection rounds. On a second step, the LR nodes upload the received uplink messages to the gateway. If the LL node requires a downlink packet from the gateway, it piggybacks this request in the message sent to the LR nodes. The LR nodes will then wait for such downlink message from the gateway, queue it locally, and relay it to the intended LL node during the next available LoRaDisC dissemination round (phase ③ in Fig. 4). Additional details on when LR nodes schedule relaying activities is provided in Sect. IV-A.

Compatibility to LoRaWAN. Because of LoRaHop's design, all nodes in the mesh are fully interoperable with existing LoRaWAN gateways and network servers, *i.e.,* the use of LoRaHop does not require any hardware or software modification to LoRaWAN gateways and servers. In fact, the latter are fully agnostic to the path taken by a packet (before reaching a gateway, or after being sent by a gateway). LoRaHop simply extends the MAC layer of LoRaWAN end-devices (enriching them with the ability of establishing and operating on a mesh network), *without* affecting its original operations: the communications between LR and LL nodes using LoRaDisC are scheduled outside normal LoRaWAN activities, as explained in detail in Sect. IV-A. Moreover, the LoRaWAN application layer and application data format is unchanged.

IV. LoRaHop: Design and Implementation

We provide next additional details about LoRaHop's design. We start by discussing how the schedule orchestrating the mesh operations is derived and how we ensure that all nodes can effectively compute rendezvous points corresponding to the beginning of the LoRaDisC flooding rounds (Sect. IV-A). We then provide additional details on LoRaDisC's inner workings (Sect. IV-B), including an early termination extension that increases energy efficiency during data collection. We finally provide additional details about our prototypical LoRaHop's implementation on off-the-shelf LoRa devices (Sect. IV-C).

A. Synchronizing Network Operations

Nodes that are part of the mesh network are not equipped with GPS modules and hence do not have a common notion of time. To synchronize their operations, we leverage the characteristics of CT-based data exchange.

Constructing the LoRaWAN activity matrix. As discussed in Sect. III-C, we let the nodes in the mesh network derive an *activity matrix* of all LR nodes, such that mesh communications do not interfere with the LoRaWAN traffic between LR nodes and gateways (phase ① in Fig. 4). This *activity matrix* can be seen as a time series indicating the time spent by each LR node in interactions with a LoRaWAN gateway. As LoRaWAN end-devices have typically constrained memory resources and as the payload of packets should be minimized, it is not possible to transmit/store each LoRaWAN activity of every LR node over a lengthy time series. Hence, we exploit the periodicity of LoRaWAN activities, and only transmit/store their *duration* and *occurrence* (denoted as $T_{activity}$ and T_{period} in Fig. 4). To this end, we let LR nodes disseminate these values during dedicated flooding rounds (they do so sequentially, based on their hard-coded logical ID). In addition to $T_{activity}$ and T_{period}, LR nodes also send T_{diff}, *i.e.,* the relative difference in time between the beginning of the last/next gateway interaction ($t_{activity\_begin}$) and the dissemination of this information (t_{dissem}). Most of these values can be derived by calling specific radio functions (*e.g.,* those returning the time in which the last packet transmission or reception was initiated), and by time-stamping the calls to interrupt functions signaling the complete transmission or reception of a packet. For example, when using the Semtech SX1276 chip, one can use the `SX1276SetTx` function and the time at which the `rxdone` interrupt function is called to derive $t_{activity\_begin}$ and $T_{activity}$[3]. T_{diff}, instead, corresponds to $t_{dissem} - t_{activity\_begin}$: the value is computed on the fly and is appended to the packet right before the latter is transmitted. t_{dissem} is computed as the time in which the packet is prepared plus a fixed offset capturing the difference in time between the packet preparation and its actual transmission[4]. LR/LL nodes receiving the packet can hence derive t_{dissem} as:

$$t_{dissem} = t_{rx\_done} - Slot_{number} \cdot T_{slot}, \tag{1}$$

where t_{rx_done} is the time at which the `rxdone` interrupt function is called, and $Slot_{number} \cdot T_{slot}$ is the duration of the CT-based flood until the packet was received (*i.e.,* the number of slots in the flood that happened before packet reception). Whilst T_{slot} is hard-coded and hence known in advance (as discussed in Sect. II-B), the $Slot_{number}$ is inferred from the content of the received packet, which embeds the current slot number information. Therefore, the start of the last/next LoRaWAN activity for each LR node can be derived as $t_{activity\_begin} = t_{dissem} - T_{diff}$. The following activities can be inferred as $t_{activity\_begin\_n} = t_{activity\_begin} + n \cdot T_{period}$, where n represents n^{th} LoRaWAN activity since the LR node transmitted its activity information. By repeating this process, all LR/LL nodes in the mesh network obtain the LoRaWAN activity matrix depicted in the bottom of Fig. 4.

Calculating LoRaDisC rendezvous points. Once the activity matrix has been constructed, the next step consists in identifying the time-slots during which all LR nodes are not carrying out any LoRaWAN activity, *i.e.,* the time-slots marked as "*overlapped idle time*" in Fig. 4. Sufficiently long idle times, *i.e.,* longer than the duration of a LoRaDisC phase

[3]We consider the worst case, *i.e.,* a downlink transmission occurring in the second reception window (see Fig. 2). We hence measure $T_{activity}$ as the difference between the time returned by the `rxdone` interrupt function called in the second reception window and the `SX1276SetTx` function.

[4]We measured this offset to be constant (≈21.636 ms) with a logic analyzer.

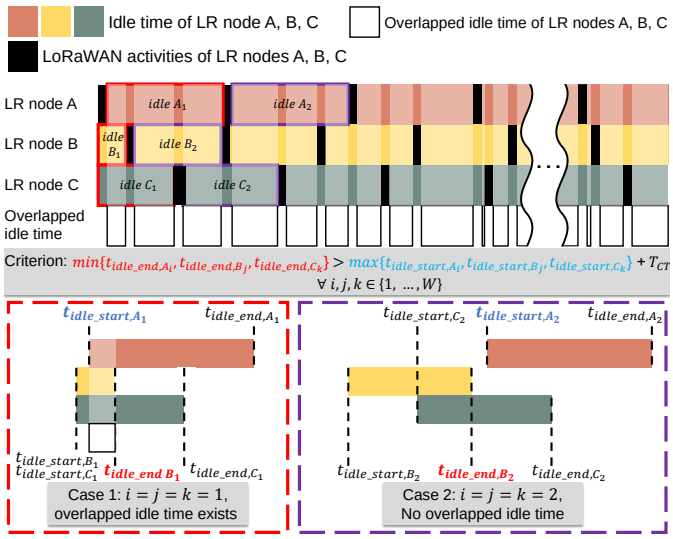


Fig. 5. Computation of next rendezvous points given the LoRaWAN activity matrix. The algorithm looks for overlapped idle time-slots that are sufficiently large to embed a LoRaDisC flood and that hence satisfy the criterion shown.

$T_{CT} = N_{slot} \cdot T_{slot} \cdot N_{flooding}$ (with N_{slot} and $N_{flooding}$ being the number of slots in a flooding round and the number of flooding rounds, respectively), can be used for LoRaDisC communications within the mesh. Fig. 5 illustrates the conditions that are necessary for this.

Specifically, given the next W idle time-slots of a node Q, whose begin/end is derived as:

$$\begin{aligned} t_{idle\_start,Q_i} &= t_{activity\_begin} + T_{activity} + i \cdot T_{period} \\ t_{idle\_end,Q_i} &= t_{activity\_begin} + (i+1) \cdot T_{period} \\ & i = 1, \ldots, W, \end{aligned} \tag{2}$$

and given three exemplary LR nodes (A, B, C), the criterion that needs to be satisfied is $T_{overlap} > T_{CT}$, where:

$$\begin{aligned} T_{overlap} = \min\{t_{idle\_end,A_i}, t_{idle\_end,B_j}, t_{idle\_end,C_k}\} - \\ \max\{t_{idle\_start,A_i}, t_{idle\_start,B_j}, t_{idle\_start,C_k}\} \\ \forall i,j,k \in \{1, \ldots, W\}. \end{aligned} \tag{3}$$

In other words, the condition to be satisfied is:

$$\begin{aligned} \min\{t_{idle\_end,A_i}, t_{idle\_end,B_j}, t_{idle\_end,C_k}\} > \\ (\max\{t_{idle\_start,A_i}, t_{idle\_start,B_j}, t_{idle\_start,C_k}\} + T_{CT}) \\ \forall i,j,k \in \{1, \ldots, W\}. \end{aligned} \tag{4}$$

In principle, one would need to compute W^{N_Q} combinations to derive the next suitable idle time-slot, with N_Q being the number of LR nodes in the network. As this would be too expensive in terms of computational overhead, we arrange all idle time-slots in a sequence from 1 to W, and pick them for each LR node in an ascending order. As soon as we find the first overlap that satisfies the above requirement, we use it as next rendezvous point $t_{rendezvous}$ (*i.e.,* as the beginning of the next LoRaDisC flooding round) and stop the computation. The process is summarized in Algorithm 1. Note that the radio is turned off between LoRaDisC phases to minimize the energy expenditure and that we add a guard time before each LoRaDisC phase to compensate for clock drift among nodes.

Algorithm 1: Calculating LoRaDisC rendezvous points

Input:
W: the number of considered idle time-slots for each LR node;
$T_{collection}$: duration of the LoRaDisC collection phase*;
T_{dissem}: duration of the LoRaDisC dissemination phase*;
$T_{overlap}$: overlap duration;
t_{idle_start,Q_i}: start time of the i^{th} idle time-slot of node Q;
*A function of the employed N_{slot}, T_{slot}, and $N_{flooding}$.

Output:
$t_{collection}$: starting instant of LoRaDisC collection phase;
$t_{dissemination}$: starting instant of LoRaDisC dissemination phase;

Initialization:
$t_{collection} = 0$;
$t_{dissemination} = 0$.

Rendezvous point search:
Derive W idle time-slots based on LoRaWAN activity matrix;
Arrange all idle time-slots in a time sequence from 1 to W;
Pick idle time-slots from each LR node in an ascending order;
for *each combination* **do**
  if *there are any downlink requests from LL nodes and* $t_{dissemination} == 0$ **then**
    if $T_{dissem} \leq T_{overlap}$ **then**
      $t_{dissemination} = \max\{t_{idle\_start,Q_i}\}, \forall i$;
      return $t_{dissemination}$
  else
    if $T_{collection} \leq T_{overlap}$ **then**
      $t_{collection} = \max\{t_{idle\_start,Q_i}\}, \forall i$;
      return $t_{collection}$
return no_rendezvous_found;

Scheduling relaying activities. Once the LL nodes sent the uplink messages destined to the gateway within a LoRaDisC collection phase[5], the LR nodes will take care of relaying them to the gateway and to receive from the gateway any downlink traffic destined to the LL nodes, as discussed in Sect. III-D. These relaying activities take place at specific instants denoted as t_{relay} and phase ③ in Fig. 4. Similar to $t_{rendezvous}$, we also derive t_{relay} based on the information collected in the LoRaWAN activity matrix and on the expected duration of the relaying activities to/from the gateway. Specifically, we derive the duration of a relaying phase as:

$$T_{relay} = N_{LL} \cdot T_{activity} \tag{5}$$

where N_{LL} is the number of LL nodes, and $T_{activity}$ is the time that elapses between the beginning of an uplink transmission from an LR node to the gateway, and the completion of a downlink transmission (in the second reception window) from the gateway to an LR node. This time can be derived from the maximum number of application-level data that can be transmitted to a gateway by an LL node per iteration, from the maximum length of a gateway response, as well as from the used radio parameters (*e.g.,* the SF): these values are either hard-coded or exchanged within the data collection phase. The relaying phase is hence divided into N_{LL} slots in which an LR node relays the messages relative to a specific LL node only: this ensures that multiple LR nodes do not interfere with each other during their relaying activities. To balance the load and minimize energy expenditure across LR nodes, the relaying activities for the various LL nodes in the network are equally

[5]LL nodes put their LoRaWAN address in the uplink packets sent to the LR nodes. The latter do not replace this address when communicating with the gateway, which makes the gateway unaware of the presence of LR nodes.

distributed among LR nodes based on the node IDs of the LL nodes. For example, in a network with four LL nodes and two LR nodes, the two LL nodes with the lowest node ID use the first LR node for their relaying activities, whereas the remaining two LL nodes with the highest node ID use the second LR node. Note that LL nodes are not involved in the relaying phase, *i.e.,* they sleep throughout this phase.

Maintaining synchronization. The LoRaWAN activity matrix needs to be updated periodically, otherwise – after a long period of time from its original computation – there is a risk of timer data overflow that would cause an incorrect calculation of the rendezvous and relay time. Therefore, before calculating these times, we update the LoRaWAN activity time of all LR nodes to the most recent one. Moreover, the LoRaWAN activity matrix needs to be periodically recomputed to avoid large synchronization errors caused by clock drifts among the various nodes in the network. For this reason, LR nodes disseminate their LoRaWAN activity information every N_F flooding rounds (with N_F being a parameter configured at compile time and set to 10 in our current implementation).

B. LoRaDisC

LoRaDisC is a multi-hop protocol that can support both one-to-all data dissemination and all-to-one data collection on top of LoRa radios [28]. By exploiting CT, LoRaDisC avoids the need of establishing and maintaining routing tables, and inherits the high reliability and energy efficiency of CT-based flooding, which was demonstrated throughout several editions of the EWSN dependability competition [49]–[51].

Data collection & dissemination primitives. LoRaDisC provides two primitives: data collection and dissemination. Each primitive consists of a series of flooding rounds which include a number of CT slots in each round as presented in Sect. II-B. Configurations such as the payload length, the number of CT slots, and LoRa communication parameters (*e.g.,* SF and bandwidth) are notified by the flood initiator, which is the first LR node joining the mesh network in our current implementation. By exploiting network coding, in LoRaDisC multiple packets are exchanged in one flooding round. Nodes blend the received packets using RLNC and transmit random linear combinations of previously-received packets in the following CT slots. During a LoRaDisC collection phase, any LL node is allowed to generate a packet and all packets generated by LL nodes are ultimately collected by LR nodes within $N_{flooding}$ flooding rounds (set to one in our current implementation). During a LoRaDisC dissemination phase, each LR node floods the downlink messages received from the gateway through the mesh network within $N_{flooding}$ flooding rounds.

Early termination. To improve energy efficiency, we introduce an early termination mechanism in LoRaDisC to conclude the data collection phase if all LR nodes have already received an uplink message from their respective LL nodes. Within a flooding round, the number of time-slots N_{slot} is proportional to the number of LR and LL nodes[6], and may be over-provisioned to cope with the dynamicity of links in outdoor environments and maximize the reliability of transmissions. To avoid unnecessary re-transmissions over several time-slots and flooding rounds, LR nodes can signal in a dedicated `termination` field that they have already received the uplink traffic from all their LL nodes. Such `termination` field, which contains one bit per LR node is then piggybacked in the packets flooded through the network. Should all LR nodes have received the expected uplink transmissions (*i.e.,* all `termination` bits are set to 1), the collection phase can be terminated. Note that the flood does not terminate instantly, but nodes still transmit information for a few time-slots to help neighbors that may have not yet realized that the data collection can be terminated. Afterwards, each node turns off its radio. Hence, the early termination optimization helps reducing energy consumption without sacrificing reliability.

Use of LBT and AFA. LoRaDisC incorporates both LBT and AFA to increase duty cycle per channel and improve data transfer speed. To comply with spectrum access regulations, LoRaDisC requires each node to listen for 5 ms before transmission to check if the channel is clear. The desynchronization error introduced by this additional delay does not affect the reliability of the flooding, as the data rate of LoRa is much lower compared to that of classical CT-based protocols in which messages should be immediately re-transmitted [29]. LoRaDisC utilizes multiple channels for communication, with each CT slot being assigned a primary and a secondary channel. Prior to transmitting a packet, nodes initiate a clear channel assessment (CCA) check on the primary channel. If the primary channel is clear, the transmission starts after 5 ms. If the primary channel is occupied, the node delays transmission for a given duration (*e.g.,* 10 ms) before performing another CCA check on the secondary channel, initiating the transmission if this channel is free. Receiving nodes listen by default to the primary channel and receive data if present. If no data is detected on the primary channel, the nodes switch to the secondary channel. Channels are periodically altered in order to optimize throughput while complying to duty-cycle regulations, and are determined based on the flooding round number and CT slot number.

C. Framework Implementation

Fig. 6 shows the high-level modules that compose our implementation of the LoRaHop framework. The latter builds upon the LoRaDisC primitives for data collection and dissemination, which are practically implemented by leveraging CT-based multi-hop communication together with RLNC. As discussed in Sect. IV-B, LoRaDisC also embeds an early termination to maximize energy efficiency during the data collection phase. Furthermore, our implementation of LoRaHop features a *scheduler* module that takes care of constructing and disseminating the LoRaWAN activity matrix, from which the LoRaDisC rendezvous points ($t_{rendezvous}$) and the time in which LR nodes should start their relaying activities (t_{relay}) are derived, as discussed in Sect. IV-A.

Hardware and software. We implement LoRaHop on off-the-shelf LoRaWAN class A end-devices consisting of an

[6]For example, in our current implementation, we use N_{slot}=60 for a mesh network with two LR nodes and 18 LL nodes, or we use N_{slot}=20 for a mesh network with one LR node and four LL nodes (see Sect. V)

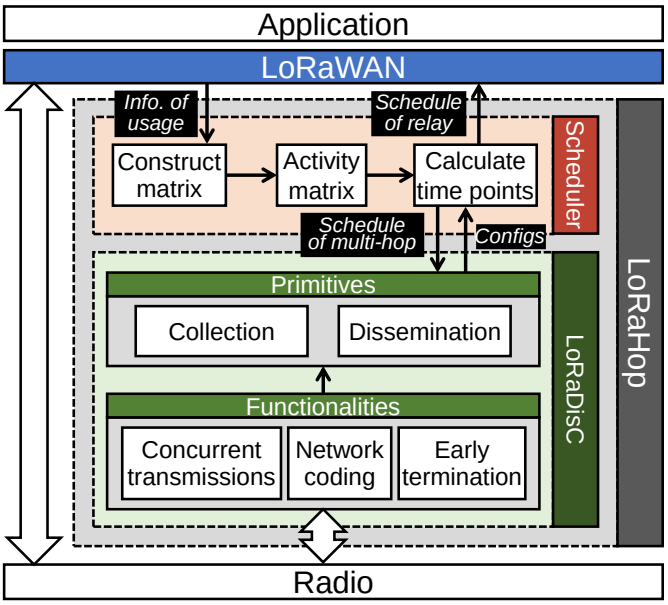


Fig. 6. High-level modules in our implementation of the LoRaHop framework.

STM32L476RG microcontroller and an SX1276MB1MAS radio. These are the same nodes used in the ChirpBox testbed [28], which are shown in Fig. 7 (a). As LoRaWAN gateways, we use the off-the-shelf RAK7243 [52] device, which supports eight channels and operates at 470 MHz[7]. We use the open-source ChirpStack [53] to build a LoRaWAN network server and set up the LoRaWAN network. All LoRaWAN packets that are received and sent by the gateway are uploaded to the server. Please note that *we do not make any changes to the LoRaWAN gateway and ChirpStack server*: this is possible because LoRaHop is fully compatible with LoRaWAN and works seamlessly together.

Activation modes for LoRaWAN end-devices. In order to minimize the implementation effort and simplify debugging, in our current prototype each LoRaWAN end-device needs to be registered to the server in either over-the-air activation (OTAA) mode or activation by personalization (ABP) mode, where device address (DevAddr), security session, and network parameters are fixed in ABP mode. LR nodes impersonate LL nodes by knowing their DevAddr, security session, and network parameters in ABP mode. However, as discussed in Sect. VI, the LR nodes could forward packets without the need of decrypting them: as a consequence, they do not necessarily need to acquire all the security credentials of LL nodes.

V. Evaluation

We evaluate LoRaHop experimentally on an outdoor testbed with 21 nodes deployed in the city of Shanghai. Our evaluation answers the following questions:

- Does LoRaHop effectively allow to extend the coverage of a LoRaWAN network thanks to its multi-hop support for both uplink and downlink traffic?

[7]Note that LoRaHop can seamlessly run on any ISM band, including those with duty cycle constraints (*e.g.,* the EU868 band). To ensure compliance with the regulations, LoRaHop monitors and tracks the use of the radio, which allows to determine if any channel is being used beyond its limits.

(a) ChirpBox end-device (b) LoRaWAN gateway

Fig. 7. ChirpBox end-device consisting of an STM32L476RG microcontroller and an SX1276MB1MAS radio (a). LoRaWAN commercial gateway RAK7243 based on an SX1301 radio, which was connected to a 5.8 dBm antenna and deployed on the roof of a building in the university campus (b).

- How does the reliability and energy efficiency of LoRaHop vary as a function of different network configurations, densities, and diameters?
- How much performance improvements can LoRaHop bring compared to classical (single-hop) LoRaWAN?
- Does a LoRaHop multi-hop network in which nodes employ a low SF perform better than a single-hop LoRaWAN network in which nodes employ a high SF?
- How much energy can be saved thanks to LoRaDisC's early termination optimization?

A. Experimental Setup

Before presenting our experimental results, we briefly describe our experimental setup, which makes use of an instance of the ChirpBox testbed deployed in the city of Shanghai.

Outdoor testbed. Fig. 8 shows a map of the testbed deployment, with the exact location of the 21 end-devices (labelled from 0 to 20) and of the gateway device (labelled as G). The ChirpBox testbed allows us to remotely control all devices and to run experiments at arbitrary times. It also supports any test firmware that meets specific constraints (*e.g.,* maximum size of the binary file, and adherence to specific libraries that should be included in the compilation). ChirpBox end-devices can store the log data of each test run: this includes logs of packets that were sent and correctly decoded by each end-device (useful to compute statistics about the packet reception ratio) as well as the time spent in different radio modes (useful to compute the energy consumption of each device).

Packet format and TX power. In our experiments, we send LoRaWAN packets with a length of 20 to 22 bytes[8]. We employ LoRaWAN's default settings, *i.e.,* a bandwidth of 125 kHz as well as a coding rate of 4/5, and configure LoRaHop to use the very same settings. Based on this configuration, LoRaHop and LoRaWAN packets when using SF=7 have the same on-air duration of 56.58 ms, whereas LoRaWAN packets sent at SF=12 last 1482.75 ms[9]. We constrain the TX power of all end-devices to 0 dBm in order to limit their communication range and show the effectiveness of LoRaHop in extending the coverage of a LoRaWAN network.

[8]The exact packet length depends on the number of LL nodes in the network, as the LoRaDisC packet header includes the `coding_vector`, whose bit length is proportional to the number of LL nodes in the mesh. Note that uplink and downlink packets have the same length.

[9]In this paper, we only use SF=7 within LoRaHop's mesh network.

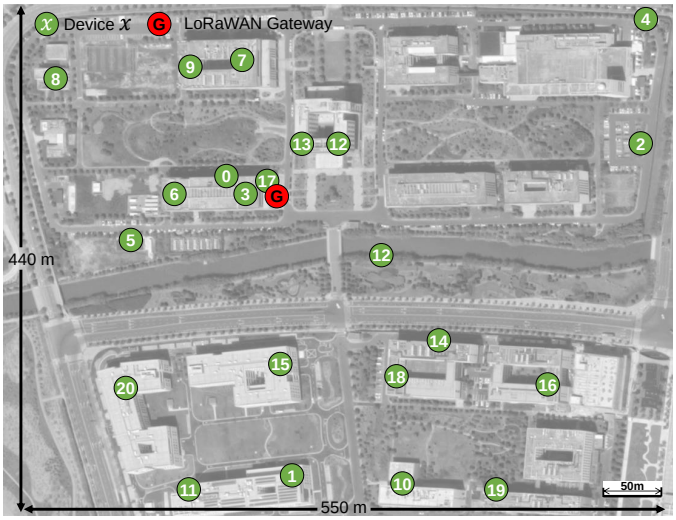


Fig. 8. Map of our outdoor deployment consisting of 21 ChirpBox end-devices (green circles) and a LoRaWAN gateway (red circle) deployed in the city of Shanghai. The end-devices as well as the LoRaWAN gateway are deployed on the top of buildings that are part of the University campus.

Performance metrics. In our evaluation, we compare the performance of LoRaHop with LoRaWAN using two main metrics: *reliability* (in terms of packet reception ratio – PRR) and energy consumption (in Joule). The PRR for uplink transmissions is computed as the number of packets correctly received and decoded by the gateway divided by the number of packets that were sent by an LL node. Similarly, for downlink transmissions, the PRR maps the number of packet correctly received by an LL node to the number of packets that were originally sent by the gateway. This data is extracted from the logs in the ChirpStack server. Note that we make use of *unconfirmed* messages when testing the uplink functionality alone, and *confirmed* messages when testing the downlink functionality: in fact, the gateway needs to receive a packet requesting a confirmation before transmitting a downlink packet. The energy consumption is defined as the consumed RF power by all end-devices in the network to send a pre-defined number of packets (100 in our experiments). The energy consumption is calculated in software following the approach proposed in Contiki's Energest [54]. We have experimentally derived that the average energy consumption of the employed LoRa radio is 53.42 mA (when transmitting at 0 dBm) and 46.81 mA (during reception), with a voltage of 3.88 V, which results in a power consumption of 207.37 mW and 181.72 mW, respectively. Sending 100 uplink messages with a 22-byte payload using SF=12 and SF=7 results in an energy consumption of 30.75 J and 1.16 J, respectively. When considering also the energy consumption caused by keeping the radio on to receive any downlink traffic during the receive window, the energy consumption with SF=12 and SF=7 amounts to 49.81 J and 6.03 J, respectively. Note that we focus on the consumption of the LoRa radio only (as this outweighs significantly that of the microcontroller), and we do not consider the energy consumption during idle time, so that our results are independent of the selected data transmission interval.

Network configurations. We evaluate the performance of LoRaWAN and LoRaHop with different network configurations, *i.e.,* by selecting different amounts and combinations of LL and LR nodes. Table I and Table II summarize all the 21 combinations studied in our experimental campaign: these vary from sparse configurations with two LL nodes and one LR node to dense configurations with 16 LL nodes and two LR nodes. Note that nodes not included in a combination are turned off and do not relay packets nor actively participate in the data transmission. Nodes 17 and 18 were chosen as LR nodes, as they can communicate with the majority of nodes in the network, and as they allow to maintain a reliable connection with the LoRaWAN gateway despite using SF=7 at all times. Many of the selected LL nodes, instead, are too far from the gateway to establish a (reliable) connection at SF=7. Therefore, we configure them to use SF=12 when using classical LoRaWAN (*i.e.,* a single-hop network) and to use SF=7 when forming the multi-hop mesh network using LoRaHop. Unless differently stated, nodes transmit data periodically: specifically, we let each LR and LL node send 100 packets (one every 25 seconds) and repeat each experiment three times.

TABLE I
NETWORK CONFIGURATIONS USED IN OUR EVALUATION.

LL node ID (number of LL nodes)	LR node ID
14, 16 (2)	18
4, 14, 16 (3)	18
2, 4, 14, 16 (4)	18
2, 4, 12, 14, 16 (5)	18
2, 4, 12, 14, 16, 19 (6)	18
2, 4, 12, 14, 15, 16, 19 (7)	18
2, 4, 12, 14, 15, 16, 19, 20 (8)	18
1, 2, 4, 12, 14, 15, 16, 19, 20 (9)	18
1, 2, 4, 11, 12, 14, 15, 16, 19, 20 (10)	18
1, 2, 4, 10, 11, 12, 14, 15, 16, 19, 20 (11)	18
1, 2, 3, 4, 10, 11, 12, 14, 15, 16, 19, 20 (12)	18
1, 2, 3, 4, 10, 11, 12, 13, 14, 15, 16, 19, 20 (13)	17, 18
1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15, 16, 19, 20 (14)	17, 18
1, 2, 3, 4, 5, 6, 10, 11, 12, 13, 14, 15, 16, 19, 20 (15)	17, 18
1, 2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14, 15, 16, 19, 20 (16)	17, 18

B. Effectiveness, Reliability, and Efficiency of LoRaHop

LoRaHop allows to reliably and efficiently connect all LL nodes to the gateway by forming a multi-hop mesh network interfacing with the selected LR nodes.

Reliability. When establishing a classical LoRaWAN single-hop network, the average PRR varies between 30% and 60% depending on the selected network configuration in both uplink and downlink messaging, as shown in Fig. 10 (light green line). This hints that several end-devices are unable to establish a reliable connection to the gateway with a single-hop network, despite the use of a higher SF. Instead, when using LoRaHop, the average PRR is always above 95% regardless of the selected network configuration (dark green line): this holds true for both uplink and downlink traffic. Hence, LoRaHop effectively increases network coverage and allows to connect LL nodes that would otherwise be unable to reliably communicate with the gateway. This is also visible from Fig. 9, which displays the PRR between each device and the gateway. When using a LoRaWAN single-hop network, even the use of SF=12

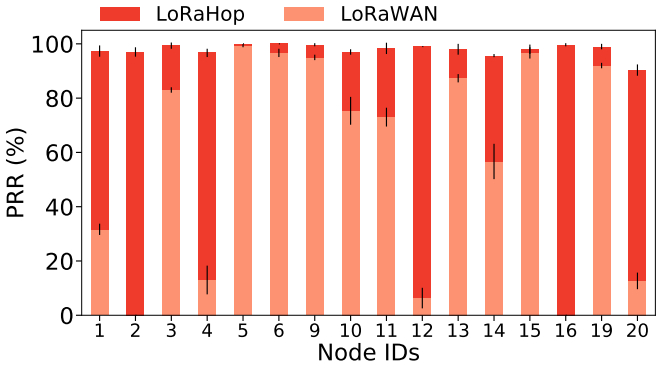


Fig. 9. PRR between each end-device (LL node) and the gateway. The creation of a mesh network using LoRaHop allows to reliably connect to the gateway also end-devices that would otherwise be disconnected or performing unreliably when using a classical LoRaWAN single-hop network.

TABLE II
NETWORK CONFIGURATIONS WHEN EVALUATING THE IMPACT OF NETWORK DIAMETER ON LORAHOP'S PERFORMANCE.

LL node ID (total number of hops)	LR node ID
10, 14, 15, 19 (4)	18
10, 14, 16, 19 (5)	18
1, 14, 16, 19 (6)	18
4, 14, 16, 19 (7)	18
1, 14, 16, 20 (8)	18
2, 4, 14, 16 (10)	18

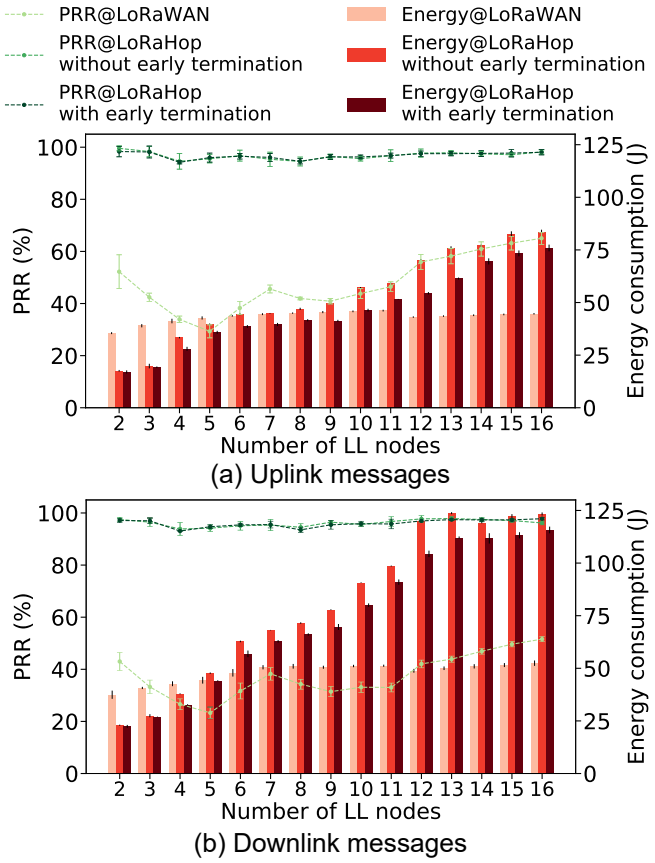


Fig. 10. Reliability and energy consumption of LL/LR nodes using LoRaWAN (single-hop network) and LoRaHop (multi-hop support) for uplink and downlink communication as a function of the number of LL nodes. When using LoRaWAN, LL nodes use SF=12, whereas SF=7 is used with LoRaHop. LoRaHop allows to extend network coverage and reliability, and LoRaDisC's early termination optimization saves up to 22.36% energy consumption.

does not allow all devices to communicate with the gateway (*e.g.,* nodes 2 and 16 have no connection) or results in a poor reliability (*e.g.,* nodes 4 and 20 have a PRR<20%). Instead, when creating a mesh network using LoRaHop, all nodes can connect to the gateway and sustain a PRR close to 100%.

Energy consumption. Furthermore, Fig. 10 also compares the average energy consumption of LL/LR nodes when forming a single-hop network using classical LoRaWAN and when supporting the formation of a multi-hop mesh network using LoRaHop. The energy consumption of LoRaHop is proportional to the number of LL nodes, whereas it remains relatively constant when using single-hop LoRaWAN. Note that the energy consumption for downlink messaging in Fig. 10(b) is higher than that for uplink messaging: this is because downlink packets follow an uplink packet, and hence the computed energy accounts for both contributions. Moreover, the figure allows to quantify the energy savings introduced by LoRaDisC's early termination optimization, which can be up to 22.36% without significant impact on communication reliability. In the remainder of this paper, all LoRaHop evaluation results will make use of the early termination optimization.

Impact of the number of hops. As mentioned above, the energy consumption of LoRaHop is proportional to the number of LL nodes in the network. This is because the number of time-slots N_{slot} in a LoRaDisC flooding round is proportional to the number of nodes in the network. Moreover, as mentioned in Sect. II-B, N_{slot} is also proportional to the network diameter. Hence, the average number of hops between LL nodes and LR nodes also affects LoRaHop's energy consumption. To quantify the impact of the network diameter on both reliability and energy consumption, we fix the number of LL nodes to 4, and vary the total number of hops between the selected LL nodes and an LR node. The total number of hops is calculated as the sum of the number of hops between each individual LL and the LR node. Table II summarizes the network configurations that we use in our study. Fig. 11 shows that, indeed, LoRaHop's energy consumption increases as a function of the network diameter (due to the higher number of CT slots in each LoRaDisC flooding round), whereas LoRaWAN's energy consumption remains constant across all configurations[10]. Note that LoRaHop's reliability remains very good despite the increase in number of hops, as the PRR remains above 95% (in contrast to that of single-hop LoRaWAN): this confirms the trends observed in Fig. 10.

Trading reliability for energy efficiency. Our previous results have shown that LoRaHop allows to connect LL nodes that would otherwise be disconnected from or unreliably connected to the gateway, thereby sustaining an almost perfect reliability (100% PRR). The price for this comes in terms of energy consumption: whilst for small mesh networks the use of LoRaHop is even more efficient than setting up a single-hop LoRaWAN network, this is no longer true for networks with several LL nodes and/or large diameters. Therefore, we

[10]Note that the configuration showing 4 LL nodes in Fig. 10 is equivalent to the configuration showing 10 hops in Fig. 11.

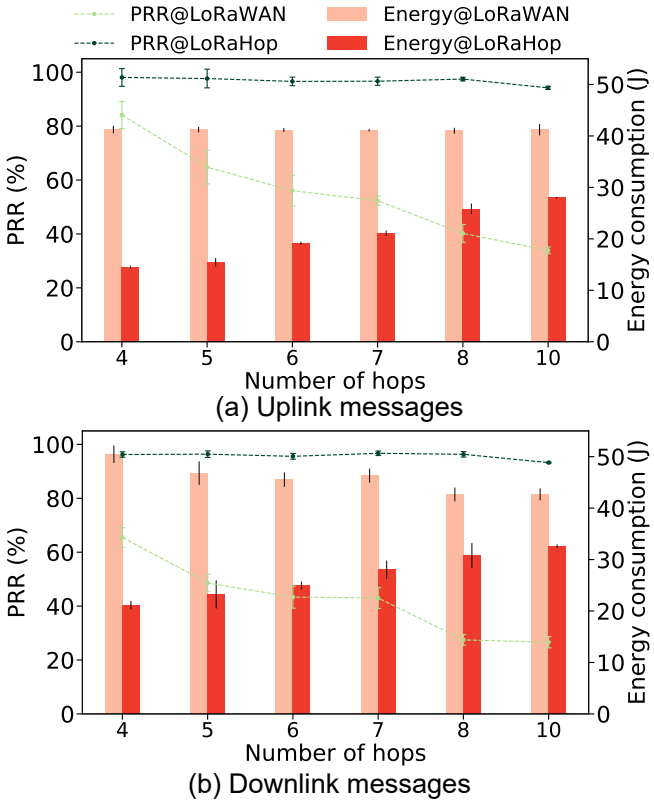


Fig. 11. Reliability and energy consumption of LL/LR nodes using LoRaWAN (single-hop network) and LoRaHop (multi-hop support) for uplink and downlink communication as a function of the total number of hops between LL and LR nodes. When using LoRaWAN, LL nodes use SF=12, whereas SF=7 is used with LoRaHop. Our results show that LoRaHop sustains a high reliability regardless of the number of hops, and that its energy consumption increases as a function of the network diameter.

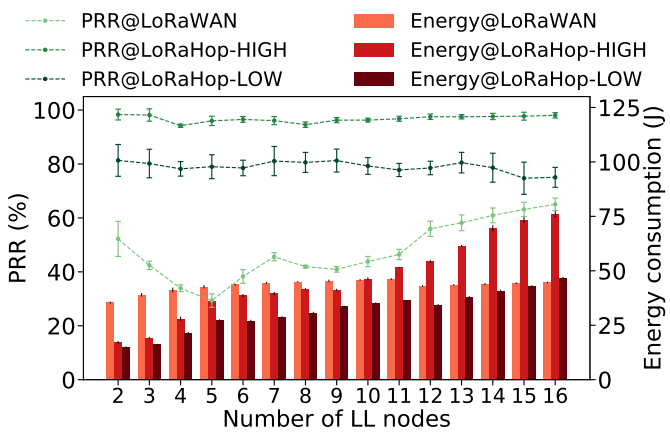


Fig. 12. Reliability and energy consumption of single-hop LoRaWAN, LoRaHop-HIGH, and LoRaHop-LOW for uplink communication as a function of the number of LL nodes. When using LoRaWAN, LL nodes use SF=12, whereas SF=7 is used with LoRaHop. With LoRaHop, one can trade reliability for energy efficiency, and outperform LoRaWAN with respect to both metrics.

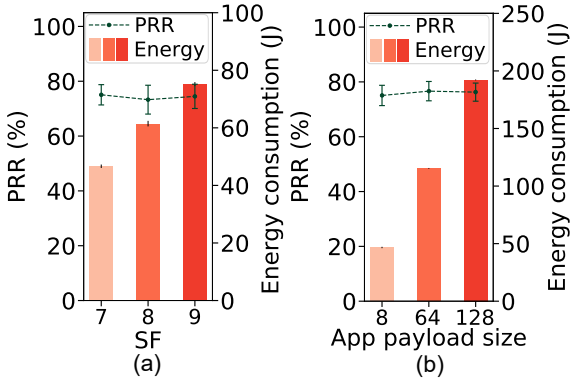


Fig. 13. Performance of LoRaHop for different SF (a) and payload lengths (b). When varying the payload length, the SF is fixed to 7.

study next if and how we can trade reliability for energy efficiency in LoRaHop. Specifically, we reduce the number of LoRaDisC slots (N_{slots}) used in each flooding round (which negatively affects the reliability of data transmissions) and study LoRaHop's performance as a function of two N_{slots} values. We name these two configurations LoRaHop-HIGH and LoRaHop-LOW. When using {2, 4, 8, 12, 16} LL nodes, N_{slots} = {12, 25, 35, 50, 60} for LoRaHop-HIGH, whereas N_{slots} = {10, 20, 25, 30, 40} for LoRaHop-LOW, respectively. All other configurations are the same as the ones we employed in Sect. V-B. Fig. 12 compares the performance of LoRaHop-HIGH and LoRaHop-LOW to that of single-hop LoRaWAN[11]. LoRaHop-HIGH and LoRaHop-LOW exhibit an average PRR across all network configurations listed in Table I of 96.8% and 78.9% as well as an energy consumption of 46.2 J and 31.6 J, respectively. In comparison, LoRaWAN exhibits an average PRR of 47.9% and an energy consumption of 43.2 J. This means that LoRaHop-LOW effectively sustains both a higher reliability (+64.7%) and energy efficiency (+26.8%) than LoRaWAN. This shows that LoRaHop can trade reliability for energy efficiency when the application requirements do not enforce a PRR close to 100%.

Performance for different SFs and payload lengths. In our previous experiments, we have fixed the SF and the payload size. We also evaluate LoRaHop's performance for additional SF configurations and payload lengths, showing their impact on PRR and energy consumption. Specifically, we use the LoRaHop-LOW configuration with 16 LL nodes and 2 LR nodes depicted in Table I, and vary the SF from 7 to 9, as well as send packets with a payload length of 8, 64, and 128 bytes. Fig. 13 shows the results: using higher SFs and payload lengths results, as expected, in an increase in energy consumption (proportional to the increased airtime of packets), whereas the reliability remains unaffected. This highlights the importance of compressing information and optimizing the payload size of packets in real-world deployments [55].

[11]Note that the configuration shown in Fig. 12 (LoRaHop-HIGH) is the same as the one used in Fig. 10 (with early termination).

C. Impact of LoRaHop on the Size of the Collision Domain

We show next how LoRaHop also allows to increase the performance of surrounding LoRaWAN networks, as the size of the *collision domain* reduces when nodes communicate over a multi-hop mesh network with a lower SF. In fact, when enforcing a single-hop network (LoRaWAN's default behaviour), nodes may be required to use a high SF to reach the gateway. This, however, enlarges the size of the collision domain and increases the probability of packet collisions: as we show experimentally, this strongly affects network performance.

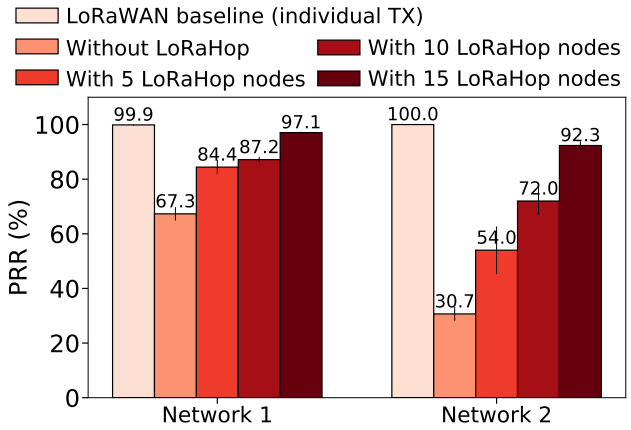


Fig. 14. Using LoRaHop allows to reduce the size of the collision domain and increase the performance of surrounding LoRaWAN networks.

We use the testbed nodes shown in Fig. 8 to set up two coexisting LoRaWAN networks: a first network in which 15 end-devices communicate to gateway G (*Network 1*)[12], and a second network in which node 2 acts as an end-device communicating to node 4, which serves as a single-channel gateway (*Network 2*). We then study the PRR in both networks as a function of the number of end-devices in *Network 1* forming a mesh network using LoRaHop. We run 5-minute experiments using different configurations in which end-devices transmit LoRaWAN frames with an 8-Byte payload. *Network 2* operates at TX power 0 dBm and uses SF=7. *Network 1* employs SF=12 and a TX power of 6 dBm[13].

Fig. 14 illustrates our results. We use as a baseline the performance of a single-hop LoRaWAN configuration in which all end-devices in *Network 1* transmit data *individually* to the gateway, *i.e.,* we let only one end-device at a time transmit data to avoid collisions. As all nodes in *Network 1* employ SF=12 and a sufficiently high TX power, the PRR in *Network 1* is close to perfect (100%), and the same applies to that of *Network 2*. Note that, to avoid any external interference biasing the experiments, we have moved to frequencies that are not used by surrounding LoRaWAN networks.

Fig. 14 also shows the PRR when all end-devices in *Network 1* periodically send uplink data to the gateway every 2.5 s. They do so by randomly picking one out of eight available channels for each transmission. In *Network 2*, instead, the two devices always use the same channel, which overlaps with one of the eight channels used by *Network 1*. We then vary the number of end-devices employing classical LoRaWAN (with SF=12) and the number of end-devices forming a mesh network (operating at SF=7) using LoRaHop in *Network 1*[14].

When all 15 end-devices use LoRaWAN with SF=12, the large collision domain affects the transmissions of *Network 2*, whose PRR drops to $\approx$30%. As we increase the number of end-devices using LoRaHop with SF=7 in *Network 1*, the collision domain is reduced, and the PRR of *Network 2* increases to 54%, 72%, and 92%, respectively, when 5, 10, and 15 end-devices form a mesh network using LoRaHop[15]. Similarly, in *Network 1*, the number of collisions are also reduced (and hence the PRR increased) when more nodes use LoRaHop, although to a lower extent than in *Network 2*.

This proves that using LoRaHop to create a multi-hop LoRaWAN network employing a low SF may lead to significant performance improvements compared to the use of a single-hop LoRaWAN network operating at a high SF.

VI. Discussion and Future Work

Support for multiple gateways. In our evaluation, we have focused on networks with only one LoRaWAN gateway. However, in principle, multiple gateways can be supported in a LoRaWAN network to increase network coverage and enhance communication reliability [56]. The existence of multiple gateways does not have an impact on LoRaHop's design and inner working. In fact, LoRaHop's relaying of uplink packets from LR nodes to multiple gateways would also work seamlessly. This is because LL nodes transmit uplink messages *without* specifying the address of the gateway. Therefore, all gateways that are in range to the LR nodes relaying the message will receive this information and upload it to the network server (regardless of whether other gateways have also received and uploaded the same information). In case of downlink messages, it is up to the network server to determine which gateway transmits the information. The LR nodes receiving these downlink messages can simply disseminate this information through the mesh network in the same way as explained in Sect. III. Possible future work includes the creation of a gateway selection method for downlink messaging that prioritizes LR nodes which have a higher signal strength to the gateway. This way, only the gateway that has more reliable connection to the LR node than others will transmit the downlink packet.

Reducing the number of collisions. As LoRaHop allows additional devices to join a LoRaWAN network, it increases the traffic to/from the gateway, which may lead to a higher number of collisions. Several works [57]–[59] have proposed mechanisms to decrease the number of collisions by exploiting time-division multiple access approaches (*e.g.,* the use of TSCH over LoRa) or by exploiting multiple gateways to decode messages. As LoRaHop is fully compatible and seamlessly integrated with LoRaWAN, almost any approach proposed in the literature for decreasing collisions in LoRaWAN can be seamlessly adopted in LoRaHop.

Insufficiently large overlapped idle time-slots. In our current implementation, the relaying phases take place within the same overlapped idle time-slot. As the number of LL nodes grows or if the LR nodes frequently contact the gateway, it may be difficult to find an overlapped idle time-slot that is sufficiently large to cover an entire relaying phase. In the future, we will

[12]Nodes 0, 1, 3, 5, 6, 9, 10, 11, 12, 14, 15, 16, 18, 19, 20 act as LoRaWAN end-devices and node G (RAK7243 device) acts as a multi-channel gateway.

[13]We have increased the TX power to ensure that all end-devices have a reliable connection to the gateway: this was not the case at 0 dBm (see Fig. 9).

[14]We investigate four configurations: (i) no end-device uses LoRaHop; (ii) 5 end-devices use LoRaHop, *i.e.,* nodes 12,14,16,18,19; (iii) 10 end-devices use LoRaHop, *i.e.,* nodes 12,14,16,18,19,1,3,11,15,20; (iv) all 15 end-devices use LoRaHop. Device 18 acts as LR node when using LoRaHop.

[15]Note that, in this experiment, we remove the two downlink receive windows after each uplink packet: this ensures that there is enough idle time available for the LoRaDisC collection and for the relaying phase.

allow splitting the relaying phases into multiple idle time-slots to improve scalability.

Adaptive LoRaDisC configuration. In our current LoRaHop implementation, values such as the number of slots in LoRaDisC (N_{slots}) are hard-coded based on the expected number of nodes in the network, network diameter, and amount of traffic generated by LL nodes. In the future, we plan to embed in LoRaDisC a mechanism that automatically adapts N_{slots} at runtime. We also plan to provide a blacklisting mechanism for channels that are congested by the transmissions of surrounding LoRaWAN networks.

Support for resource-constrained devices. Although we have already shown that LoRaHop can successfully run on resource-constrained LoRa end-devices, we aim to further improve the energy efficiency of the proposed solution. Specifically, in future work we plan to include support for multiple sub-networks in LoRaHop, so to reduce the number of necessary CT slots and hence decrease the overall radio on-time.

Security implications. Even though our prototypical LoRaHop implementation assumes that the LR nodes obtain all the security certificates from the LL nodes (see Sect. IV-C), one can relax this assumption. In fact, LR nodes could forward packets without the need of decrypting them. Specifically, the application data of LL nodes is first enclosed into an application-layer frame following the LoRaWAN format and then further encapsulated as the payload of a LoRaDisC frame when being transmitted between LR and LL nodes. The LR nodes only need to decode the LoRaDisC header in order to forward the application-layer frame, and do not need to decode its content. For this reason, the LR node does not necessarily need to obtain the security certificate of the LL nodes, as the LR node can forward the LoRaWAN packets to/from a gateway without decoding and encoding their data. Consequently, LoRaHop does not prevent the use of OTAA activation mode for LoRaWAN end-devices. In future work, we plan to adjust our LoRaHop implementation accordingly, and we also plan to enhance the security of LoRaHop by implementing an authentication process in which the identity of new end-devices is verified before allowing them to join the mesh network and/or to relay messages. This would prevent that a rogue node impersonates another device and/or interferes with the LoRaDisC communications.

VII. Related work

LoRaHop allows the creation of a LoRaWAN multi-hop mesh network in order to improve the network coverage and reliability, while minimizing energy consumption. We analyse next related works that aim to extend the coverage of a LoRaWAN network and that propose solutions for building multi-hop networks of devices embedding a LoRa radio. Furthermore, as LoRaHop leverages CT, we review other works making use of CT within LoRa networks.

Extending the coverage of LoRa networks. A large body of studies on LoRa focus on the issue of network coverage. These works study, for example, how to increase the receiver sensitivity on the gateway side [60]–[62], design custom LoRa chirp signals on the end-device side [63], or add router/extender nodes that are responsible of the forwarding of packets [18], [19], [21]. However, those works either need advanced hardware replacing commercial gateways [60]–[63] or end-devices [21], or rely on permanently-powered router nodes [18], [19], which is unfeasible if the end-devices have severe energy constraints. In addition, most of these studies only focus on uplink messaging, but do not support downlink messaging. In this article, we use off-the-shelf LoRaWAN end-devices and gateways, and do *not* change the original LoRaWAN stack/application, i.e., our approach can be seamlessly incorporated into LoRaWAN. Furthermore, LoRaHop supports both uplink and downlink messaging.

Multi-hop support for LoRa networks. A number of works have investigated how to build multi-hop LoRa networks [16], [17], [21], [22]. Most of these multi-hop networks, however, are not compatible with LoRaWAN [16], [17], or support only a limited number of hops/devices [22]–[25]. Ahmar et al. [64] have proposed a routing solution for LoRaWAN multi-hop networks that can reduce the number of end-devices operating at high SFs. This solution, however, requires that all end-devices have a connection to the gateway (*e.g.,* with the highest SF). This requirement is introduced due to the necessity of synchronize all nodes in the network to precisely calculate the wake-up time for the various transmissions. Instead, LoRaHop can form a multi-hop mesh network independently on the presence of a gateway in the surroundings of all nodes. On the contrary, LoRaHop purposely allows to connect LoRaWAN end-devices that are *not* in range with a gateway.

CT support for LoRa. The principle of CT has been first investigated and demonstrated in the context of short-range IEEE 802.15.4 narrowband systems [29], [35], and later also in ultra-wideband [40], [65] as well as Bluetooth Low Energy systems [41], [42]. As CT enable a fast and efficient network flooding, several works have proposed a number of CT-based protocols and primitives such as the low-power wireless bus [66], Chaos [67], Mixer [36], Crystal [68], and Harmony [38]. The reliability and efficiency of such techniques has been demonstrated by several editions of the EWSN dependability competition [51], which fostered and triggered the creation of CT-based frameworks that can simplify the creation of an application [69] and are capable of switch between multiple PHYs at runtime [70]. CT have also been investigated in the context of long-range communication on LoRa-based devices [26]–[28]. Specifically, Bor et al. [26] have been the first to experimentally demonstrate the feasibility of CT with LoRa. Liao et al. [27] have theoretically analyzed the feasibility of CT as well as developed a prototypical implementation. In previous work, we have exploited CT to build the communication backbone of an infrastructure-less LoRa testbed [28]. In this work, to the best of our knowledge, we propose the first LoRaWAN-compatible framework that makes use of CT to create a multi-hop mesh network, thereby extending the coverage and scalability of a LoRaWAN system.

VIII. Conclusions

In this paper, we propose LoRaHop, an extension of LoRaWAN that enriches end-devices with the ability to form

a multi-hop mesh network. This allows to extend network coverage to end-devices that would otherwise be too far from a gateway to establish (a reliable) communication, and allows the exchange of both uplink and downlink traffic in a seamless way. We provide details on the design of LoRaHop in Sect. III by describing the mesh network operations (which build upon LoRaDisC, a protocol based on the CT paradigm), as well as the scheduling of the various communication and relaying activities. After detailing the implementation of LoRaHop and our proof-of-concept on off-the-shelf devices in Sect. IV, we evaluate LoRaHop's performance experimentally in Sect. V and compare it with that of classical (single-hop) LoRaWAN. Our results show that LoRaHop can effectively extend the coverage of a LoRaWAN network while improving reliability by up to 98.33% and halving energy consumption.

REFERENCES

[1] N. Silva *et al.*, "Low-Cost IoT LoRa Solutions for Precision Agriculture Monitoring Practices," in *Proc. of the 19th EPIA Conf.*, 2019.

[2] Y. Cheng *et al.*, "Secure Smart Metering based on LoRa Technology," in *Proc. of the 4th ISBA Conf.*, 2018.

[3] P. J. Basford *et al.*, "LoRaWAN for Smart City IoT Deployments: A Long Term Evaluation," *Sensors*, vol. 20, no. 3, 2020.

[4] G. Pasolini *et al.*, "Smart City Pilot Projects Using LoRa and IEEE 802.15.4 Technologies," *Sensors*, vol. 18, no. 4, 2018.

[5] B. Foubert *et al.*, "Long-Range Wireless Radio Technologies: A Survey," *Future Internet*, vol. 12, no. 1, 2020.

[6] LoRa Alliance, "LoRaWAN Specification, v1.1," 2017. [Online] http://bit.ly/3NYM3QF – Last access: 2022-11-12.

[7] The Things Network. https://www.thethingsnetwork.org/.

[8] S. Demetri *et al.*, "Automated Estimation of Link Quality for LoRa: A Remote Sensing Approach," in *Proc. of the 18th IPSN Conf.*, 2019.

[9] C. Boano *et al.*, "Impact of Temperature Variations on the Reliability of LoRa: An Experimental Evaluation," in *Proc. of the 7th SENSORNETS Conf.*, 2018.

[10] M. Cattani *et al.*, "An Experimental Evaluation of the Reliability of LoRa Long-Range Low-Power Wireless Communication," *JSAN*, vol. 6, no. 2, 2017.

[11] The Things Network, "The Things Indoor Gateway," 2019. [Online] http://bit.ly/3HLEo6G – Last access: 2023-02-01.

[12] M. Bor *et al.*, "LoRa Transmission Parameter Selection," in *Proc. of the 13th DCOSS Conf.*, 2017.

[13] M. Bor *et al.*, "Do LoRa Low-Power Wide-Area Networks Scale?," in *Proc. of the 19th MSWiM Conf.*, 2016.

[14] K. Mikhaylov *et al.*, "Analysis of Capacity and Scalability of the LoRa Low Power Wide Area Network Technology," in *Proc. of the 22th European Wireless Conf.*, 2016.

[15] N. Kouvelas *et al.*, "P-CARMA: Politely Scaling LoRaWAN," in *Proc. of the 17th EWSN Conf.*, 2020.

[16] S. Hong *et al.*, "A Hierarchy-based Energy Efficient Routing Protocol for LoRa-Mesh Network," *IEEE Internet of Things Journal*, 2022.

[17] X. Jiang *et al.*, "Hybrid Low-Power Wide-Area Mesh Network for IoT Applications," *IEEE Internet of Things Journal*, vol. 8, no. 2, 2020.

[18] H.-C. Lee *et al.*, "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, 2018.

[19] K.-H. Ke *et al.*, "Demo Abstract: A LoRa Wireless Mesh Networking Module for Campus-Scale Monitoring," in *Proc. of the 16th IPSN Conf.*, 2017.

[20] J. Dias *et al.*, "LoRaWAN Multi-hop Uplink Extension," *Procedia Computer Science*, vol. 130, 2018.

[21] E. Sisinni *et al.*, "LoRaWAN Range Extender for Industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, 2019.

[22] H. Huh *et al.*, "LoRa-based Mesh Network for IoT Applications," in *Proc. of the 5th WF-IoT Conf.*, 2019.

[23] C. Ebi *et al.*, "Synchronous LoRa Mesh Network to Monitor Processes in Underground Infrastructure," *IEEE Access*, vol. 7, 2019.

[24] B. Van de Velde, "Multi-hop LoRaWAN: Including a Forwarding Node," tech. rep., 2017. [Online] http://bit.ly/3totdsz – Last access: 2022-11-12.

[25] M. Aslam *et al.*, "Exploring Multi-Hop LoRa for Green Smart Cities," *IEEE Network*, vol. 34, no. 2, 2020.

[26] M. Bor *et al.*, "LoRa for the Internet of Things," in *Proc. of the 1st MadCom Workshop*, 2016.

[27] C.-H. Liao *et al.*, "Multi-Hop LoRa Networks Enabled by Concurrent Transmission," *IEEE Access*, vol. 5, 2017.

[28] P. Tian *et al.*, "ChirpBox: An Infrastructure-Less LoRa Testbed," in *Proc. of the 18th EWSN Conf.*, 2021.

[29] M. Zimmerling *et al.*, "Synchronous Transmissions in Low-Power Wireless: A Survey of Communication Protocols and Network Services," *ACM Computing Surveys (CSUR)*, vol. 53, no. 6, 2020.

[30] J. Liando *et al.*, "Known and Unknown Facts of LoRa: Experiences from a Large-scale Measurement Study," *ACM Transactions on Sensor Networks*, vol. 15, no. 2, 2019.

[31] V. Keerthi *et al.*, "Cloud IoT Based Greenhouse Monitoring System," *Int. J. Eng. Res. App.*, vol. 5, no. 10, 2015.

[32] P. Angin *et al.*, "AgriLoRa: A Digital Twin Framework for Smart Agriculture," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 11, no. 4, 2020.

[33] H. Arkian *et al.*, "Potable Water Management with Integrated Fog Computing and LoRaWAN Technologies," *IEEE IoT Newsletter*, 2020.

[34] H. Nurwarsito *et al.*, "Development of Multi-Point LoRa Network with LoRaWAN Protocol," in *Proc. of the 8th ICITACEE Conf.*, 2021.

[35] F. Ferrari *et al.*, "Efficient Network Flooding and Time Synchronization with Glossy," in *Proc. of the 10th IPSN Conf.*, 2011.

[36] C. Herrmann *et al.*, "Mixer: Efficient Many-to-All Broadcast in Dynamic Wireless Mesh Networks," in *Proc. of the 16th SenSys Conf.*, 2018.

[37] M. Mohammad *et al.*, "Codecast: Supporting Data Driven In-Network Processing for Low-Power Wireless Sensor Networks," in *Proc. of the 17th IPSN Conf.*, 2018.

[38] X. Ma *et al.*, "Harmony: Saving Concurrent Transmissions from Harsh RF Interference," in *Proc. of the 39th INFOCOM Conf.*, IEEE, 2020.

[39] T. Chang *et al.*, "Constructive Interference in 802.15.4: A Tutorial," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, 2019.

[40] M. Trobinger *et al.*, "One Flood to Route Them All: Ultra-Fast Convergecast of Concurrent Flows over UWB," in *Proc. of the 18th SenSys Conf.*, ACM, 2020.

[41] B. Nahas *et al.*, "BlueFlood: Concurrent Transmissions for Multi-Hop Bluetooth 5 – Modeling and Evaluation," *ACM Transactions on Internet of Things*, vol. 2, no. 4, 2021.

[42] M. Baddeley *et al.*, "The Impact of the Physical Layer on the Performance of Concurrent Transmissions," in *Proc. of the 28th ICNP Conf.*, 2020.

[43] J. Haxhibeqiri *et al.*, "Low Overhead Scheduling of LoRa Transmissions for Improved Scalability," *IEEE Internet of Things Journal*, vol. 6, no. 2, 2018.

[44] J. Finnegan *et al.*, "Lightweight Timeslot Scheduling Through Periodicity Detection for Increased Scalability of LoRaWAN," in *Proc. of the 21st WoWMoM Conf.*, 2020.

[45] Q. Qadir *et al.*, "Low Power Wide Area Networks: A Survey of Enabling Technologies, Applications and Interoperability Needs," *IEEE Access*, vol. 6, 2018.

[46] Y. Liu *et al.*, "Efficient Load Balancing for Heterogeneous Radio-Replication-Combined LoRaWAN," *IEEE Transactions on Industrial Informatics*, 2022.

[47] M. Cattani *et al.*, "Adige: An Efficient Smart Water Network based on Long-Range Wireless Technology," in *Proc. of the 3rd CySWATER Workshop*, 2017.

[48] R. Hofmann *et al.*, "SERVOUS: Cross-Technology Neighbour Discovery and Rendezvous for Low-Power Wireless Devices," in *Proc. of the 18th EWSN Conf.*, 2021.

[49] M. Schuß *et al.*, "Moving Beyond Competitions: Extending D-Cube to Seamlessly Benchmark Low-Power Wireless Systems," in *Proc. of the 1st CPSBench Workshop*, 2018.

[50] M. Schuß *et al.*, "A Competition to Push the Dependability of Low-Power Wireless Protocols to the Edge," in *Proc. of the 14th EWSN Conf.*, 2017.

[51] C. Boano *et al.*, "EWSN Dependability Competition: Experiences and Lessons Learned," *IEEE IoT Newsletter*, 2017.

[52] RAKwireless, "RAK7243 Gateway," 2021. [Online] http://bit.ly/3Esd1Nn – Last access: 2022-11-12.

[53] O. Brocaar, "ChirpStack, Open-Source LoRaWAN® Network Server." [Online] https://www.chirpstack.io/ – Last access: 2022-11-12.

[54] A. Dunkels *et al.*, "Software-based On-line Energy Estimation for Sensor Nodes," in *Proc. of 4th EmNetS Workshop*, 2007.

[55] D. Săcăleanu *et al.*, "Data Compression in Wireless Sensor Nodes with LoRa," in *Proc. of the 10th ECAI Conf.*, 2018.

[56] A. Dongare *et al.*, "Charm: Exploiting Geographical Diversity through Coherent Combining in Low-Power Wide-Area Networks," in *Proc. of the 17th IPSN Conf.*, 2018.

[57] R. Piyare *et al.*, "On-Demand LoRa: Asynchronous TDMA for Energy Efficient and Low Latency Communication in IoT," *Sensors*, vol. 18, no. 11, 2018.

[58] D. Zorbas *et al.*, "Time-slotted LoRa Networks: Design Considerations, Implementations, and Perspectives," *IEEE Internet of Things Magazine*, vol. 4, no. 1, 2020.

[59] M. Haubro *et al.*, "TSCH-over-LoRa: Long Range and Reliable IPv6 Multi-hop Networks for the Internet of Things," *Internet Technology Letters*, vol. 3, no. 4, 2020.

[60] A. Dongare *et al.*, "Charm: Exploiting Geographical Diversity through Coherent Combining in LP-WANs," in *Proc. of the 17th IPSN Conf.*, 2018.

[61] C. Li *et al.*, "NELoRa: Towards Ultra-low SNR LoRa Communication with Neural-enhanced Demodulation," in *Proc. of the 19th ACM SenSys conf.*, 2021.

[62] Z. Xu *et al.*, "Pyramid: Real-time LoRa Collision Decoding with Peak Tracking," in *Proc. of the IEEE INFOCOM Conf.*, IEEE, 2021.

[63] C. Li *et al.*, "CurvingLoRa to Boost LoRa Network Throughput via Concurrent Transmission," in *Proc. of the 19th NSDI Conf.*, 2022.

[64] A. Ahmar *et al.*, "Smart-Hop: Low-Latency Multi-hop Networking for LoRa," in *Proc. of the 18th DCOSS Conf.*, 2022.

[65] D. Lobba *et al.*, "Concurrent Transmissions for Multi-hop Communication on UWB Radios," in *Proc. of the 17th EWSN Conf.*, 2020.

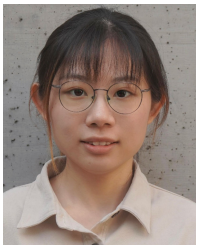
[66] F. Ferrari *et al.*, "Low-power Wireless Bus," in *Proc. of the 10th ACM SenSys Conf.*, 2012.

[67] O. Landsiedel *et al.*, "Chaos: Versatile and Efficient All-to-All Data Sharing and In-Network Processing at Scale," in *Proc. of the 11th ACM SenSys Conf.*, 2013.

[68] T. Istomin *et al.*, "Data Prediction + Synchronous Transmissions = Ultra-low Power Wireless Sensor Networks," in *Proc. of the 14th ACM SenSys Conf.*, 2016.

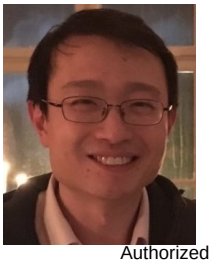
[69] R. Jacob *et al.*, "Synchronous Transmissions Made Easy: Design your Network Stack with Baloo," in *Proc. of the 16th EWSN Conf.*, 2019.

[70] M. Baddeley *et al.*, "OSF: An Open-Source Framework for Synchronous Flooding over Multiple Physical Layers," in *Proc. of the 19th EWSN Conf.*, 2022.



PEI TIAN received the B.S. degree in Electrical and Computer Engineering from the Donghua University, Shanghai (CN) in 2018. She is currently pursuing the Ph.D. degree with the Shanghai Advanced Research Institute, CAS (CN) and is a visiting Ph.D. student at Graz University of Technology (AT). Her research interests include wireless network systems, wireless network protocol design, and network testbeds.

CARLO ALBERTO BOANO received an MSc degree in computer engineering from Politecnico di Torino (IT) and KTH (SE) in 2009, as well as a PhD degree *sub-auspiciis praesidentis* in information and computer engineering from Graz University of Technology (AT) in 2016. He is currently an Associate Professor at the Institute of Technical Informatics at Graz University of Technology. His research focuses on the design of dependable networked embedded systems with emphasis on the efficiency and reliability of low-power wireless communications.



XIAOYUAN MA received the B.S. and M.S. degrees from Shanghai Jiao Tong University, Shanghai (CN) in 2010 and 2014, respectively. He received the Ph.D. degree at the Shanghai Advanced Research Institute, CAS (CN). He joined the Safety and Emergency Laboratory, Shanghai Advanced Research Institute, Chinese Academy of Sciences (CN), in 2011. His research interests encompass wireless sensor networks, and the implementations of embedded systems.

JIANMING WEI is currently a Professor of Shanghai Advanced Research Institute, CAS (CN). His research interests include the design of wireless sensor networks for signal processing and communication system, the system design for collaborative processing and information fusion as well as applications in industrial intelligence and urban public safety. He is also interested in real-time monitoring and early-warning technology, emergency communication and integrated automation control systems.