Doctor of Philosophy

A Real-Time LoRa Protocol for Periodic and Aperiodic Data Transmission

The Graduate School of the University of Ulsan

Department of Electrical, Electronic and Computer Engineering

QUY LAM HOANG

A Real-Time LoRa Protocol for Periodic and Aperiodic Data Transmission

Supervisor: Professor Oh Hoon

A Dissertation

Submitted to
the Graduate School of the University of Ulsan
In partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

by

QUY LAM HOANG

Department of Electrical, Electronic and Computer Engineering
University of Ulsan, Korea
February 2022

A Real-Time LoRa Protocol for Periodic and Aperiodic Data Transmission

This certifies that

the dissertation of QUY LAM HOANG is approved.

Committee Chair Prof. Yoon, Seokhoon

Committee Member Prof. Oh Hoon (Advisor)

Committee Member Prof. Koo, Insoo

Committee Member Prof. Jo, Dongsik

Committee Member Dr. Yoo, Daesung

Department of Electrical, Electronic and Computer Engineering

University of Ulsan, Korea

February 2022

Acknowledgments

Without the unconditional support of many people, this dissertation would not have been done. I would like to express my gratitude to everyone who helped me with this dissertation.

First and foremost, I would like to express my sincere appreciation to my advisor, Oh Hoon professor, for giving me a chance to join the Ubiquitous Computing (Ubicom) lab. During six years working here, I have gained a lot of experience in both doing research and doing real projects. Professor Oh Hoon gave me invaluable instructions in completing my Ph. D degree. I appreciate his support, encouragement, and advice in both studies and life.

I would also like to thank the committee members for giving me useful comments and suggestions to make this dissertation comprehensive.

During my time studying at the University of Ulsan (UOU), I had a great opportunity to be friends with other students, both from Vietnam and from other countries, who also come to Korea to find something that makes them better life, maybe. I am grateful to them for their sincere friendship, enthusiastic help, and cheerfulness during my stay in Ulsan.

I am deeply indebted to my parents for their love and support. Even they did not stay beside me during the Ph. D journey, but they always pray for me and give me a lot of encouragement to get over troubles in life.

Lastly, I would like to express my love for my wife and my little daughter. Without them, I would not be who I am today. Thank you for coming into my life.

A Real-Time LoRa Protocol for Periodic and Aperiodic Data Transmission

by

Quy Lam Hoang

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Electrical, Electronic and Computer Engineering)

February 2022

Abstract

Due to the provision of a long-range and robust link, the LoRa technology draws attention for its use in industrial data collection networks. In such networks, an end device or task can either transmit data periodically or transmit data in an event-driven manner to a server. This Ph.D. dissertation proposes a real-time LoRa protocol that can deal with both periodic and aperiodic data effectively. Based on the definition of a frame-slot architecture, a *logical slot indexing algorithm* is devised to tag a logical index to each slot in a frame. The slots are logically partitioned such that they are first scheduled for periodic data by the slot scheduling algorithm and then the remaining unscheduled slots are used for event-driven aperiodic data. In this logical frame partitioning, the unscheduled slots appear in an interleaved fashion so that aperiodic tasks can transmit data with low delay and fairness while every periodic task still completes data transmission before the beginning of the next period. To deal with the problems of data collision and traffic congestion for aperiodic data, a *two-level collision avoidance* scheme is proposed that adopts the notion of contention window and delay slot.

To evaluate the performance of the proposed protocol, we utilize two evaluation methods: (1) conducting experiments on the testbed of the real LoRa devices to evaluate the data transmission of periodic tasks and (2) performing simulations with various

scenarios to evaluate the data transmission of aperiodic tasks. It was proven by experiment and simulation that the proposed protocol achieves better performance compared with other recent protocols, LoRWAN, ILoRa and RT-LoRa. Specifically, it can not only guarantee the timely delivery of periodic data, but also deal with aperiodic data with high reliability, fairness, and low delay compared with others. Therefore, it is expected that the proposed protocol is highly suitable for industrial monitoring and control systems that require the support for the transmission of both periodic and aperiodic data.

Contents

List of Figures

List of Tables

Chapter 1. Introduction

In this Chapter, the fundamentals of the LoRa networks are briefly described in the first subsection. Then, the next subsection discusses the issues and problems in designing a real-time LoRa protocol and previous approaches. Next, our approach to deal with both periodic real-time data and aperiodic data transmission and the method used to evaluate the proposed approach are presented. Finally, the last subsection summarizes the contributions and the organization of the dissertation.

1.1 LoRa Networks

1.1.1 LoRa Physical Layer

LoRa [1] is a long-range wireless narrowband technology that uses low-power devices to transmit data in long ranges with low data rates. LoRa technology offers efficient and flexible two-way communication between a server and a large number of end devices, thereby enabling a large number of IoT applications for both civil and industrial applications. The LoRa devices operate in unlicensed sub-GHz ISM band and supports different frequencies for different regions, for example 868 MHz in Europe, 433 MHz in Asia, and 915 MHz in North America. LoRa technology employs the chirp spread spectrum (CSS) technique [2] that limits the data rate, but allows the demodulation for extremely low strength signals, even lower than the noise floor, thereby enabling long transmission range.

The LoRa transmission is characterized by three main modulation factors: spreading factor (SF), bandwidth (BW), and coding rate (CR).

Spreading factor (SF): SF defines the signal spreading level by configuring the ratio between symbol rate and chip rate. Data rate and transmission range can be controlled by changing the value of SF in range between 7 and 12. The larger the SF used, the farther the signal will be able to propagate and still be received by the RF receiver without errors [3]. The spreading technique allows LoRa nodes to overcome the signal attenuation problem, even in industrial environment that has a lot of obstructions such as construction machines and steel walls, thereby enabling a start topology [4].

Bandwidth (BW): BW is the width of frequencies in the transmission band. LoRa can be configured to use bandwidths in fixed values from 7.8 KHz to 500 KHz. Choosing a narrower bandwidth will result in a slower transfer rate, but improved range while choosing a wider bandwidth will result in an increased data rate, but a shorter range. The BW of 125 kHz is commonly used in various applications.

Coding rate (CR): CR is the Forward Error Correction rate adopted by the LoRa demodulator to protect the transmitted signal from interference. The coding rate describes the *ratio* of actual data to error-correcting data added and can be set to either 4/5, 4/6, 4/7 or 4/8. As such, a higher coding rate will not increase range, but will make a signal more reliable if interference is present.

Table 1.1. Spreading factor (SF) versus bit rate and time-on-air

SF	Bitrate	Transmission range	Time on air
7	5470 bps	2 km	56 ms
8	3125 bps	4 km	100 ms
9	1760 bps	6 km	200 ms
10	980 bps	8 km	370 ms
11	440 bps	11 km	740 ms

12	290 bps	14 km	1400 ms

(With bandwidth 125kHz, coding rate 4/5, app payload 10 bytes)

The combination of these factors determines bit rate, receiver sensitivity and the packet *time-on-air* (*ToA*), that eventually affects transmission range and energy consumption of the device, respectively. Table 1.1 summarizes the bitrate, transmission range and time-on-air according to different SFs.

The LoRa packet consists of a preamble, the payload, and CRC. The preamble duration, $T_{preamble}$, is given as follows:

$$T_{preable} = \left(n_{preamble} + 4.25\right) \times T_{sym} \tag{1.1}$$

where $T_{sym} = {2^{SF}}/{BW}$, and 4.25 indicates a number of symbols that are included mandatorily in the preamble part. The payload duration, $T_{payload}$, can be calculated as follows:

$$T_{payload} = payloadSymbNb \times T_{sym} \tag{1.2}$$

where *payloadSymbNb* indicates the number of symbols that is used to represent the payload, given as follows:

$$payloadSymbNb = 8 + max\left(ceil\left(\frac{8BL-4SF+28+16-20H}{4(SF-2DE)}\right)(CR+4), 0\right) \tag{1.3}$$

where *PL* is payload size in bytes, *H*=0 and *H*=1 when a header enabled and disabled, respectively, and *DE*=0 and *DE*=1 when *Low Data Rate Optimize* is enabled and disabled, respectively. Finally, the packet time-on-air, T_{packet}, can be calculated as the sum of the preamble duration and the payload duration as follows:

$$T_{packet} = T_{preamble} + T_{payload} \tag{1.4}$$

1.1.2 LoRaWAN

LoRaWAN [5] is a low-power, wide-area networking protocol designed by LoRa Alliance to provide two-way data communication service that is flexible and easy for deployment. The LoRaWAN network consists of a number of end devices or nodes, one or more gateways (GWs), and a network server (NS). An end device can be associated with one or more GWs instead of one particular GW. The data transmitted by the end device can be directly received by multiple gateways simultaneously. Furthermore, each LoRa GW equipped with a powerful transceiver is capable of receiving and decoding multiple packets simultaneously in the same channel [6]. The LoRaWAN protocol stack is fully implemented on end devices and the network server, whereas a gateway just acts as a bridge that relays the packets from the end devices to the server and vice versa. Since the LoRaWAN network operates in unlicensed bands, both end devices and GWs must follow the channel access regulation that limits transmission power and duty cycle. In the European region, the duty cycles is limited to 0.1%, 1%, or 10% depending on the frequency bands [7].

LoRaWAN refers to the communication protocol and system architecture in which end devices are implemented with three protocol layers: Physical layer, data link layer, and application layer. Its data link layer has a proprietary MAC protocol running on the LoRa physical layer. For the reliability service in MAC layer, the protocol supports two types of messages: *confirmed message* and *unconfirmed message*. When an end device sends a confirmed message, the server has to respond with an acknowledgement message. According to the LoRaWAN specification [8], there are three classes of end device: Class A, Class B and Class C. These classes are defined to satisfy the requirements of different applications.

With Class A, upon receiving uplink message, a server reserves two downlink slots, RX1 and RX2 with different intervals, RECEIVE_DELAY1 and RECEIVE_DELAY2, respectively, as illustrated in Figure 1.1. Then, if a certain end device sends confirmed message, the server utilizes one of these two downlink slots to respond with an acknowledgement message. Furthermore, it utilizes these two downlink slots to send a command message to the end device that has sent a message regardless of message type.

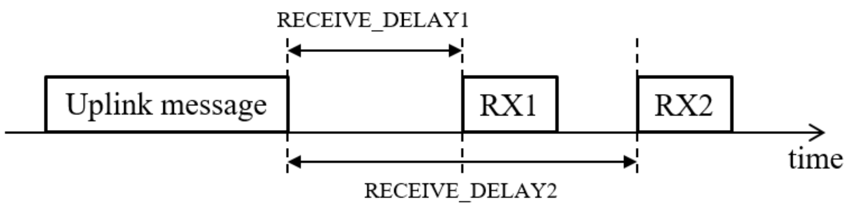


Figure 1.1. Slot timing of LoRaWAN Class A device

Since an end device employs the Aloha protocol to access the channel, it can send data anytime with any spreading factor and goes to sleep. Then, it wakes up to wait for a downlink message at either RX1 or RX2. In RX1, a sever sets the transmission parameters (frequency, CR, SF) to the parameter values of the end device; whereas, in RX2, it sets those parameters to default values defined by LoRaWAN.

Class A devices can only receive downlink communications following uplink transmissions, so it can suffer from either latency of downlink message transmission or the increase of power consumption in application that requires a great number of downlink transmissions. Thus, Class B is designed for the demand of additional downlink traffic applications. Devices in Class B exploit periodic beacons sent by the gateway to allow the schedule of additional downlink traffic. Meanwhile, the devices in Class C always listen to the channel to receive every upcoming message. Class C is suitable for applications which require low latency in downlink messages.

1.2 Real-Time LoRa Protocol

1.2.1 Issues and Problems

Recently industrial applications have been considering the use of the LoRa technology for data collection networks in work fields due to the provision of a long-range and stable link and the ease of deployment [4, 9]. One of those applications is an industrial safety monitoring and control system in which a server provides a safety service by analyzing of the data collected from end devices through a network. A server may collect data from a node periodically to keep track of the location and condition of workers or equipment, or to monitor the time-varying situation of the target work field. On the other hand, a server may collect data in an event-driven manner. For example, a server may ask a specific node to turn on or off a specific sensor module in the node for energy management. Then, the node will change the operating status of the sensor module and report the result to the server. As another example, a node may have to transmit data *instantly* if it detects any abnormal or emergency situations such as the emission of toxic gas and a worker's entrance to a restricted access zone. Thus, this requires the design of a real-time LoRa MAC protocol that can deal with both periodic and aperiodic data effectively.

However, it is not easy the design a LoRa MAC protocol that can support the transmission of both periodic and aperiodic data types in industrial monitoring and control systems since the following issues need to be considered.

- First, in industrial monitoring and control systems, a server collects data regularly from each end device, thereby incurring relatively high traffic loads. For the worse, since LoRa data has a lengthy *time-on-air* (*ToA*), the industrial LoRa network suffers from high data collision between different data transmissions.
- Second, in industrial monitoring and control systems, each node can have its own data transmission period (that becomes time constraint) to prevent erroneous operations from the temporal inaccuracy of sensor data. Moreover, different sensor nodes may have different data transmission periods since some nodes have to send data more frequently than others.
- Third, a node may transmit data to the server in event-driven manner if it detects an emergency in the work field and has to report to the server. In this case, the data has to be transmitted with low delay.
- Finally, nodes that are located far away from the gateway can experience the signal suppression by the nodes that are located nearby the gateway, resulting in the lack of fairness in data transmission. Due to imperfect orthogonality characteristic and capture effect, LoRa gateway is allowed to receive only the strongest signal among signals transmitted simultaneously, even with different SFs [10].

The first two issues can be tackled by the use of the time division multiple access (TDMA); however, the latter two issues have to be treated in a best-effort manner since the arrival of aperiodic data is not controllable and different levels of signal attenuation due to pathloss and obstructions in the work field is not avoidable. The use of TDMA raises another issue regarding the execution of scheduling, which can be either centralized or distributed. Most WSN protocols [11-13] employ a centralized approach in which a server generates a slot schedule based on information about the whole network, and distributes it to participating nodes. However, changes in the topology cause the update of the schedule and overhead in redistributing the slot schedule. In the slotted sense multiple access (SSMA) protocol [14] that uses the distributed scheduling approach, every node generates its own slot schedule to overcome the difficulty; this still has limitations in time synchronization and tree construction in a large-scale network. Fortunately, this may not be a problem in LoRa networks since the LoRa technology provides a long-range and robust link [5, 6] that allows the use of a single-hop star topology. However, the distribution of slot schedule in LoRa network should be carefully considered since it is still a burden due to its low bandwidth.

1.2.2 Previous Approaches

The previous approaches that deal with periodic and aperiodic data transmission in a star topology divide a frame into two sections and allocate different sections to the different types of tasks. The IEEE 802.1 standard (WiFi) [15] mainly deals with best-effort services at different Quality of Service (QoS) levels while it has limitations in dealing with real-time traffic [16]. To support the transmission of data with different QoS levels, WiFi uses a superframe that has distinct time periods as a *contention period* (CP) and a *contention-free period* (CFP), and uses CSMA/CA to avoid collision during CP while it uses a slot scheduling during CFP for guaranteed transmission. The IEEE 802.15.1 standard (Bluetooth) also provides different QoS levels as best-effort and guaranteed best-effort transmissions [17]; however, it incurs some additional overhead by having a master manage the transmission of a slave for reliable transmission. Nevertheless, WiFi and Bluetooth may not experience the degradation of performance notably under high bandwidth since the size of a control message is significantly smaller than that of data.

In wireless sensor networks (WSNs) [18] or LPWA networks [19] such as LoRa, SigFox, and NB-IoT that are constrained by resources and data rate, control messages should be used in a limited manner with careful consideration. Similarly to WiFi, the IEEE 802.15.4 standard [20] also uses a superframe that consists of a *contention access period* (CAP) and a *contention free period* (CFP). However, it exposed some problems such as unbounded delay and limited reliability in data transmission according to multiple studies [21, 22]. The IEEE 802.15.4e protocol [11] dealt with those problems. It defines different MAC behavior modes to address various requirements for industrial applications. One of these modes is the Deterministic and Synchronous Multi-channel Extension (DSME) mode that is designed to support deterministic delay and high reliability in data transmission with adaptability to time-varying traffic. Even though the DSME mode can support the transmission of both periodic and aperiodic traffic in a flexible manner [11], it may not be suitable to industrial WSNs in which the topology can often be changed, considering that the change of topology cannot only damage the integrity of a real-time service, but also increase control overhead considerably by triggering network reconstruction and slot rescheduling.

Meanwhile, a couple of research have dealt with both periodic and aperiodic data transmission in LoRa protocols [23, 24]. In [23], the authors proposed the Industrial LoRa protocol (ILoRa) that borrowed the frame partitioning method of the IEEE 802.15.4 standard. They used the Aloha protocol during CAP for aperiodic data transmission while using an offline slot schedule during CFP. In [24], they improved the ILoRa protocol to have a real-time LoRa protocol (RT-LoRa) by using the Slotted Aloha during CAP for aperiodic data transmission and using the QoS-oriented slot scheduling mechanism during CFP for periodic data transmission. However, this approach can waste slots since a node always occupies the number of slots corresponding to the integer multiple of the superframe size that is equal to its period. Furthermore, since the frame are physically divided into two different periods, an aperiodic task that arrives during CFP has to wait for channel competition until the beginning of the next CAP, resulting in increased transmission delay.

1.3 Our Approach

For convenience, τ_x as a *periodic task* on node *x* is represented as follows:

$$\tau_x = (x, p_x)$$

Where *x* and p_x indicate a node address and a transmission period, respectively, and a_x as an *aperiodic task* that arrives at time *t* on node *x* is represented as follows:

$$a_x = (x, t)$$

We assume that every aperiodic data can be transmitted within one data slot. If it cannot be fit into one data slot, it can be segmented and transmitted separately.

To handle two different types of data, periodic and aperiodic data, a number of LoRa protocols [23, 24] uses the same frame division approach as the IEEE 802.15.4 standard. Let this frame division be a *zone-based frame division (ZFD)*. In ZFD, since a frame size is determined as the smallest one of all task periods, every periodic task should have its transmission period as the integer multiple of the frame because they do not employ any real-time scheduler. Therefore, a task can waste ($n - 1$) slots if it uses the transmission period of $n \times$ *frame*, $n \geq 1$. On the other hand, an aperiodic task has to wait until the beginning of the following CAP, thereby increasing the delay of data transmission.

To deal with both periodic and aperiodic data effectively, the proposed approach uses a slotted frame in which the frame is divided into 2^N slots, where N is a non-negative integer. For the easy slot scheduling of periodic tasks, the *logical slot indexing* (*LSI*) algorithm is devised to assigns a logical slot index to every slot in the frame such that if a task with the transmission period of 2^{N-k} slots is assigned 2^k slots corresponding to the 2^k sequential logical slot indices starting with any logical slot index, and transmits one data in each assigned slot, it can always transmit one data per its transmission period. Given a set of periodic tasks, each of them is scheduled such that it obtains slots corresponding to the sequential logical slot indices starting with the last scheduled logical slot index. After scheduling all the periodic tasks, the unscheduled slots are accessed through contention by aperiodic tasks. In this way, the scheduled slots and the unscheduled slots appear in the interleaved fashion, thereafter referred to as a *slot-based frame division (SFD)*. Therefore, the SFD approach can reduce the average transmission delay of an aperiodic task compared to the ZFD approach.

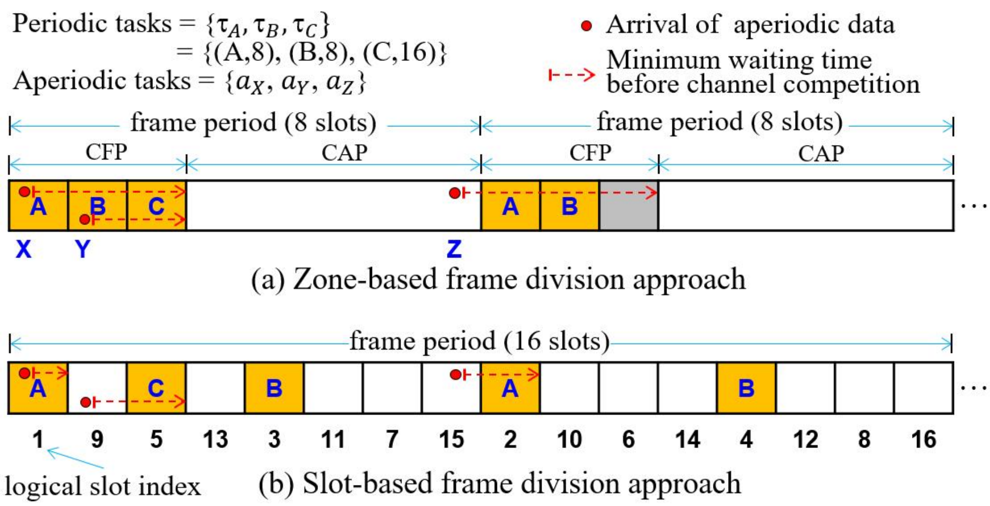


Figure 1.2. Comparison of two frame division approaches

Figure 1.2 illustrates how periodic and aperiodic tasks are scheduled with ZFD and SFD. Consider a set of periodic tasks $\{\tau_A, \tau_B, \tau_C\} = \{(A,8),(B,8),(C,16)\}$ and three aperiodic tasks a_X, a_Y, and a_Z with their respective arrival times as shown in the figure. Referring to Figure 1.2-(a) that illustrates the scheduling in ZFD, the frame size is determined based on a periodic task with minimum period (i.e., 8 slots). Two periodic tasks τ_A and τ_B are assigned one slot per frame while task τ_C gets one slot per two frame periods, resulting in one wasted slot. On the contrary, take a look at Figure 1.2-(b) that illustrates the scheduling in SFD. The frame size was determined to be sufficiently large by considering the total slot demands of periodic tasks and their periods. In this example, the frame size is 16 slots long. Periodic tasks τ_A, τ_B, and τ_C are assigned 5 slots, starting with a logical slot index of 1. It is observed that non-deterministically arriving aperiodic tasks can get an unscheduled slot quickly.

The use of SFD gives the following advantages. First, the exactly required number of slots can be allocated to periodic tasks by using the slot scheduling algorithm without wasting slots. Second, the distributed unscheduled slots allow an aperiodic task to find an unscheduled slot quickly. In ZFD, for channel contention, an aperiodic task has to wait by the start of the next CAP unless it arrives during CAP or if it arrives at the later part of CAP, it can fail to transmit data and wait for the whole CFP. Third, since aperiodic tasks contend a channel only at the slot boundary, it takes advantage of the principle of Slotted Aloha naturally. Finally, aperiodic tasks achieve the fairness in data transmission due to the randomness of their arrivals and the distribution of unscheduled slots.

Considering the advantages of the SFD approach in supporting both periodic and aperiodic data transmission, this Ph.D. dissertation presents a novel real-time LoRa MAC protocol to support two types of data efficiently. The protocol uses a *logical frame partitioning* such that the slots in the frame are logically divided into a set of scheduled slots for periodic data and a set of unscheduled slots for aperiodic data by the slot scheduling algorithm. In this partition, the scheduled slots and the unscheduled slots appear in the interleaved fashion instead of two physically split sections. This allows a node with aperiodic data to find an unscheduled slot fast. To secure the reliability of aperiodic data transmission, the proposed protocol includes a *two-level collision avoidance* scheme in which a node selects an unscheduled slot randomly among a specified number of forthcoming unscheduled slots named a *contention window* and then contends a channel in the selected slot again using a random number of *delay slots* to avoid collision before data transmission. The former and the latter are called a *level one collision avoidance* (*L1-CA*) scheme and a *level two collision avoidance* (*L2-CA*) scheme, respectively. If a node fails to acquire a channel in the selected slot, it restarts *L1-CA* with a doubly increased contention window for congestion control.

1.4 Evaluation Method

One metric for evaluation includes *packet delivery ratio* (PDR) defined as the ratio of data packets received at the server to those transmitted by all nodes. Another is *average delay* defined as the elapsed time from the time when a node generates a packet to the time when the server receives the packet. The delay includes a binary back-off delay, random delay within a slot, channel activity time, and packet ToA; it does not include the delay in the backhaul network. The other one is the *fairness* of data delivery defined as the degree regarding how equally every node can have the chance of success in packet transmission without being sacrificed by the capture effect. The fairness is examined by comparing the PDRs of nodes as median, maximum, minimum, and the 1st and the 3rd quartile values.

The proposed protocol, RTLoRa-LFP, is evaluated by both experiment and simulation. For periodic data transmission, experiment was performed on a testbed with a single-channel gateway and fifteen nodes to examine the reliability of transmission using slot schedule. It is believed that a single channel is sufficient to reveal the key operational characteristics of our protocol. The LoRaWAN protocol [5] that uses the Aloha approach and one of its variation [25] that uses Slotted Aloha approach are used for performance comparison. For aperiodic data transmission, the proposed protocol that uses the SFD approach was compared with ILoRa [23] and RT-LoRa [24] that use the ZFD approach in terms of packet delivery ratio (PDR), and the delay and fairness of data transmission under three scenarios that are differentiated by node distributions, traffic loads, and the intensity of the hidden node problem.

1.5 Contribution and Dissertation Organization

The dissertation presents the design and performance evaluation of a Real-time LoRa MAC protocol that can transmit both periodic and aperiodic data effectively. The proposed protocol not only guarantees that periodic tasks finish their data transmission within their respective time constraints, but also allows aperiodic tasks to transmit their data with high reliability and short delay. In summary, the contributions of this research are as follows:

- We identify the problems in designing a real-time LoRa protocol that support different data types in the industrial monitoring and control system and present the motivation of this study.
- We propose a frame-slot architecture that defines the frames on different channels. A frame consists of a downlink section and an uplink section, and the uplink section is further divided into a number of slots to facilities the data transmission from end nodes to gateways.
- We devise a *logical slot indexing (LSI) algorithm* that assigns a logical slot index to each slot of the frame. The LSI algorithm allows a server to generate a slot schedule such that if it assigns sequential logical slot indices to a periodic task, and the task transmits one data in each assigned slot, it can always transmit one data per its transmission period.
- We propose *logical frame partitioning* frame such that the slots in the frame are logically divided into a set of scheduled slots for periodic data and a set of unscheduled slots for aperiodic data by the slot scheduling algorithm. In this partition, the scheduled slots and the unscheduled slots appear in the interleaved

fashion instead of two physically split sections, thereby allowing aperiodic task to select an unscheduled slot to transmit data with low delay.

- For aperiodic data transmission, we propose *two-level collision avoidance* scheme in which a node selects an unscheduled slot randomly within a *contention window* and then contends a channel in the selected slot again to avoid collision before data transmission.

The dissertation consists of six chapters structured as follows:

Chapter 1 presents the fundamental knowledge about the LoRa networks, issues and problems in designing a real-time LoRa protocol and previous approaches. Then, the contributions and organization of the dissertation is given.

Chapter 2 provides the related works including the discussion on the traditional wireless communication technologies such as Wifi, Bluetooth and wireless sensor networks, and also the discussion on the recent LoRa protocols.

Chapter 3 discusses the problems in real-time scheduling for LoRa networks and presents our proposed logical slot indexing algorithm and related theorems.

Chapter 4 introduces a detail description of the Real-time LoRa protocol with logical frame partitioning for periodic and aperiodic data transmission and the analysis of slot utilization of the proposed protocol.

Chapter 5 provides the performance evaluation of the proposed protocols with the experiment and simulation results.

Finally, Chapter 6 gives conclusions on the dissertation and the future research direction.

Chapter 2. Related Works

2.1 WiFi and Bluetooth

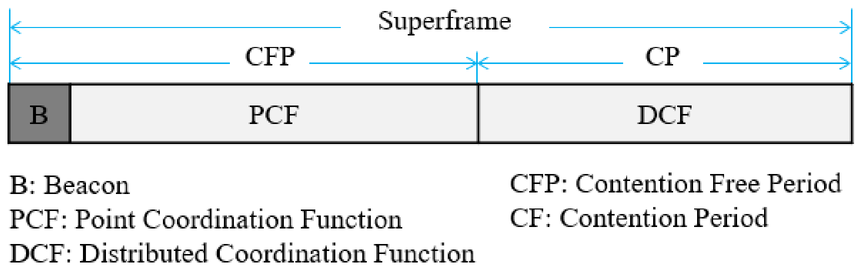


Figure 2.1. IEEE 802.11 superframe structure

Many studies regarding the reliable transmission for two types of data, periodic and aperiodic data, have been conducted so far under wireless communication technologies such as WiFi and Bluetooth. Wifi (or IEEE 802.11 standard) provides two operating modes as *distributed coordination function* (DCF) and *point coordination function* (PCF). In the former mode, it employs the carrier sense multiple access with collision avoidance (CSMA/CA) mechanism for reliable data transmission such that a node is only allowed to transmit data if it detects an idle channel; otherwise, it waits till channel becomes idle and then transmits data. If collision happens, the colliding nodes employ the binary exponential backoff technique to wait a random amount of time before trying again later. To further reduce the possibility of collisions, the DCF mode may use the *clear-to-send/request-to-send* (CTS/RTS) technique to guarantee that both the sender and receive starting data communication only if they agree with each other.

The latter mode, point coordination function, provides an alternative means of accessing the wireless medium to accommodate applications that require real-time service. In this mode, the WiFi standard uses a *superframe* that consists of a *contention-free period* (CFP) and a *contention period* (CP) as shown in Figure 2.1. The Access Point (AP) transmits a beacon frame at the beginning of the contention-free period to announce its network. The beacon frame contains all the information about the network to help a node to discover and identify a nearby AP to associate with it. For data transmission, a node employs CSMA/CA to avoid collision during CP and uses a slot scheduling during CFP for guaranteed transmission. This structure allows the protocol to support different data types in different time portion.

Bluetooth (IEEE 802.15.1 standard) is another common wireless communication technology that provides different QoS levels as best effort and guaranteed best effort [17]. It is more suitable for applications that require high data rate transmission under short-term and short-range connection. The Bluetooth technology also supports different QoS levels as a best-effort or guaranteed best-effort transmission [17]; however, it incurs some additional overhead by having a master node control the transmission of a slave node for reliable transmission. Nevertheless, both WiFi and Bluetooth may not experience the degradation of performance noticeably since they have high bandwidth and use a tiny control message compared to data.

2.2 Wireless Sensor Networks

Similarly to WiFi, the IEEE 802.15.4 standard [20] uses a superframe that consists of a *contention access period* (CAP) and a *contention free period* (CFP) as shown in Figure 2.2.

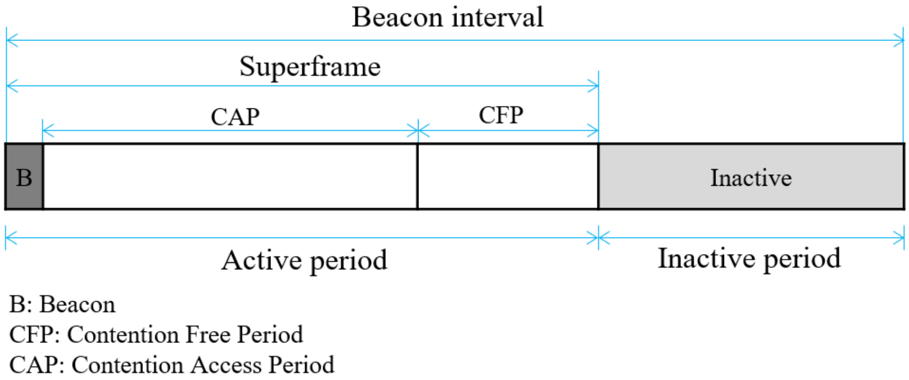


Figure 2.2. IEEE 802.15.4 superframe structure

The beacon frame, which is broadcast in the first slot of each superframe, defines the boundaries of the superframe. The beacons are used to identify the network, synchronize the time, and provide the superframe's setup. The superframe can have an inactive time during which the nodes go into low-power mode to save power. In CAP, a node must compete with other devices utilizing a slotted CSMA/CA method for data transmissions. Data transmissions in CFP, on the other hand, are scheduled according to guaranteed time slots (GTSs) allocated to the node by the coordinator. In networks without beacon synchronization, unslotted CSMA-CA is used to transfer data frames, and assured data transmissions via GTSs are not possible. Even though the IEEE 802.15.4 standard can support the transmission for different data types owing to the superframe structure, it incurs some problems such as unbounded delay and limited reliability in data transmission according to some studies [21, 22].

To address the shortcomings of the existing 802.15.4 MAC protocol, the *IEEE 802.15.4e MAC Enhancement Standard* [26] has been proposed. The 802.14.3e protocols is a low-power multi-hop MAC protocol that is designed to meet the requirements of industrial applications. To achieve this purpose, it incorporates some techniques from other industrial wireless communication standards such as WirelessHART and ISA 100.11.a [27]. In particular, 802.15.4e adds MAC behavior modes, which are additional MAC protocols designed to support specific application domains, to the existing 802.15.4 standard. Among these modes, the Deterministic and Synchronous Multi-channel Extension (DSME) mode was introduced to achieve deterministic delay and high reliability in data transmission and also to improve the adaptability to time-varying traffic. Even though the DSME mode provides the mechanisms to accommodate both periodic and aperiodic data, it does not provide the complete implementation that can deal with a dynamic network topology in industrial environments [11]. In fact, the change of topology can not only damage the quality of a real-time service, but also increase control overhead considerably by triggering network reconstruction and slot rescheduling.

2.3 LoRa Networks

Even though WiFi, Bluetooth and WSNs provide the mechanisms for the reliable transmission of periodic and aperiodic data, WiFi and Bluetooth are seldom used in data collection networks due to limited scalability, spectrum inefficiency, or high energy consumption, and WSNs have difficulty in meeting industrial requirements due to multi-hop routing and the dynamic nature of WSN topology [28]. The LoRa technology can overcome those limitations due to a simple star network topology enabled by the long transmission range and low energy consumption, the capture effect that enables the protective operation from other LoRa networks [29], and the provision of high link stability and multiple data rates [30, 31].

LoRaWAN has recently been adopted to used in a variety of applications, including smart cities, environmental monitoring and location tracking, etc. [3]. Due to the random nature of data transmission, a LoRaWAN node suffers from the high probability of data collision as traffic increases. The collision problem with the scalability issue is studied in papers [32-35]. Furthermore, it obvious that the collision problem becomes worse with the use of the higher spreading factors (SF) that have the longer transmission range and the lengthier packet ToA.

The data transmission protocol for industrial monitoring and control systems should be able to deliver periodic data within their time constraints and aperiodic data in a best-effort manner. However, the LoRa technology is vulnerable to data collision, especially in industrial monitoring systems that involve heavy data traffic. So far, a lot of efforts were made to secure the reliability of data transmission in LoRa networks. The early studies regarding data transmission mostly focused on evaluating the performance of LoRaWAN [32-35], indicating that LoRaWAN is not suitable to applications that have relatively high data traffic due to data collisions. Some studies tried to improve reliability in data transmission by using the slotted transmission constrains the start of data transmission only to the boundaries of time slots [36, 37] or CSMA with the listen-before-talk (LBT) mechanism [38, 39]. Meanwhile, some protocols employ a *slot scheduling* to remove data collision completely. In the slot scheduling approach, every node is allocated a distinct time slot such that if it transmits data within the allocated slot, no data collision occurs [40, 41]. In [42], a relay node connected directly to GW creates a subnet with a small number of nodes in the underground zone and then collects data from them by using a slot schedule. However, it still relies on LoRaWAN to forward the collected data to GW. The on-demand LoRa protocol [43] allows a server to collect data from nodes in a cluster that is managed by a clusterhead. By using a short-range wake-up radio combined with the LoRa module, the clusterhead awakes its members and schedules transmission slots for them. Although this approach improves reliability, it can limit its application scenario since its operation relies on the short-range wake-up radio. The authors in [44] improve the concurrent transmission technique used in the Glossy approach [45] by employing the notion of transmission time offset, referred to as the *offset-CT* approach. This approach allows two or more nodes to transmit data concurrently, but with different time offsets; however, it suffers from high energy consumption since it involves flooding in every data transmission.

2.4 Industrial LoRa protocols

Recently, some studies dealt with the problem of reliable and/or real-time data transmission for industrial applications [46-48]. In [46], the authors mentioned the adaptation of the time-slotted channel hopping (TSCH) [11] used in IEEE 802.15.4e to LoRa networks for reliability without giving a specific slot scheduling method. In [47], they proposed a distributed slot scheduling using a hash function with which every node determines its own slots such that different nodes do not occupy the identical slot. With this approach, the waste of slots is inevitable since the frame size should be large enough to guarantee the distinct allocation of slots. In [48], the real-time LoRa protocol generates a slot schedule based on the data transmission periods of nodes such that every node can transmit data before the beginning of the next period if it sends data according to the slot schedule; it does not address the transmission of aperiodic data.

A couple of protocols have been proposed to deal with both periodic and aperiodic data in LoRa networks. Both of them borrowed the superframe structure used in the IEEE 802.15.4 standard with some modifications. In the first protocol, Industrial LoRa or

ILoRa [23], the authors introduced a medium access control scheme for LoRa networks that facilitates the deployment of industrial wireless networks for Industrial Internet of Things. In the second protocol, RT-LoRa [24], the authors presented a medium access strategy for LoRa that can support both periodic real-time data transmission and aperiodic data transmission.

2.4.1 Industrial LoRa (ILoRa)

In the Industrial LoRa protocol (ILoRa) [23], the end nodes transmit data directly to a sink node that acts as the network coordinator, thereby forming a star network topology. To support both periodic and aperiodic data transmission, ILoRa works based on the zone-based frame division approach (used in the IEEE 802.15.4 standard) that splits a superframe physically. The superframe consists of five sections, the beacon section, the Contention Access Period (CAP), the Contention-Free Period (CFP), the Downlink Period and the CFP Ack section, as shown in Figure 2.3.

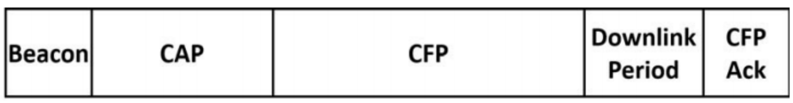


Figure 2.3. The structure of superframe proposed in ILoRa

During CAP, the node can transmit aperiodic data to the gateway by using the Pure ALOHA mechanism. When an aperiodic data arrives, a node generates the transmission parameter pair including frequency channel and SF and transmits the data packet on the selected channel and SF after a random delay. During CFP, the node transmits data using an offline slot schedule. The CFP is sliced into a number of timeslots and the size of each timeslot is long enough to transmit a data packet with the maximum-sized application payload. It was shown by simulation that the nodes can transmit periodic data without collision problem since they follow the slot schedule. However, a high portion number of aperiodic data are lost due to the collision problem since the node use the Aloha protocol during CAP.

In summary, ILoRa provides support for both periodic and aperiodic data transmission by employing the superframe structure. However, this approach used an offline slot schedule in simulation without relying on any specific mechanism for slot assignment. In addition, they do not address the collision problem caused by the Aloha protocol for transmission aperiodic data.

2.4.2 Real-Time LoRa (RT-LoRa)

The RT-LoRa protocol [24] improves ILoRa to support a real-time service. It uses Slotted Aloha during CAP for aperiodic data transmission while it suggests a slot scheduling approach during CFP for periodic data transmission. For a real-time service, it defines a superframe that corresponds to the shortest data transmission period among nodes and requires every node to transmit one data every superframe to meet time constraints. They also present the QoS-oriented slot scheduling mechanism to meet different QoS requirements for periodic data. It classifies nodes into three QoS classes as *Normal*, *Reliable*, and *Most Reliable* and assigns the resources such as spreading factors (SFs) and transmission slots differently to the nodes according to QoS levels. Specifically, a “Normal” node, a “Reliable” node, and a “Most Reliable” node is assigned one slot associated with the lowest possible SF, one slot associated with the highest possible SF, and multiple slots, each of them being associated with a distinct SF, respectively.

RT-LoRa tried to support both a real-time service and QoS together; however, it can increase the size of a superframe and restrict the schedulability considerably because high SF occupies a long slot and two nodes using different SFs on the same channel can cause collision if their signal strength offset is less than a specified threshold [10]. For aperiodic data transmission, they showed by simulation that the Slotted Aloha protocol used in CAP can only alleviate the probability of collision to some extent. Thus, Slotted Aloha may not be suitable for the applications that require high reliability in data transmission. In addition, the transmission delay of aperiodic data has not been evaluated in their studies.

Chapter 3. Logical slot indexing algorithm

3.1 Problems in real-time scheduling for LoRa networks

In industrial monitoring and control systems, each task is required to send one data packet to a server per its own data transmission period. Since a task can have a transmission period shorter than the length of a frame, it can transmit data multiple times within one frame period. Suppose that a frame is divided into 2^N data transmission slots. If a task has its transmission period that corresponds to $2^N/2^k$ $(0 \le k \le N)$ slots, it should be allocated 2^k slots within the frame such that each allocated slot appears only once per its transmission period. However, since different tasks can have different periods, it may not be easy to make the whole task set schedulable. If a brute-force scheduling method is employed, a slot schedule can be biased such that some tasks take all the early slots, thereby preventing other tasks from being scheduled.

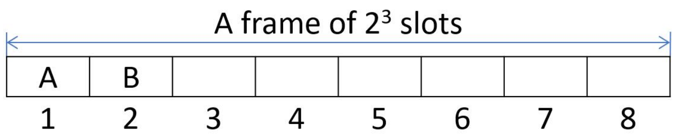


Figure 3.1. An example of a biased scheduling

For example, consider a frame of 2^3 slots (N = 3) as depicted in Figure 3.1 and three tasks A, B, and C that have the periods of 2^3, 2^3 and 2^1 slots, respectively. If tasks A and B take slots 1 and 2, respectively, task C that requires 4 slots cannot be feasibly scheduled since all the slots in the first quarter frame are not available.

The slot scheduling based approach includes some additional functions such as time synchronization and the generation, distribution, and maintenance of a slot schedule. With a star topology, time synchronization is relatively easy [49]; however, the implementation of other functions can be very costly in low bandwidth LoRa networks. One way to alleviate scheduling overhead is to divide nodes into groups so that a slot scheduling can be done for each group independently. Furthermore, if all nodes can know a *frame structure* and the periods of all tasks in the network, every node can generate the same slot schedule in distributed manner. The schedule distribution problem can be reduced to the distribution problem of task scheduling information.

3.2 Logical slot indexing algorithm

A *logical slot indexing (LSI)* algorithm assigns a *logical index* to each slot in a frame such that given a frame of 2^N slots, if a task $\tau_x = (x, p_x)$ with transmission period $p_x = {2^N}/{2^k}$ selects 2^k slots sequentially starting from an arbitrary logical index, it can send one data packet per period p_x to a server. For example, suppose that any two sequential logical indices are assigned to two slots such that they do not belong to the same side of two equally divided parts of the frame. Then, if a task with period $p_x = {2^N}/{2^1}$ slots selects those two slots, it can send one packet per p_x. This principle has to hold for any task with various periods.

Definition 1. Given any logical index i, j sequential logical slot indices from i to $i + j - 1$ are said to be *sound* if $2^{k-1} < j \leq 2^k$ and each of them is assigned to a slot that belongs to one of the 2^k equally divided sections of a frame.

Definition 2. The j sequential logical slot indices from 1 to j are said to be *feasible* if (1) the $j-1$ sequential logical slot indices from 1 to $j-1$ are *feasible* and (2) for all $i < k$ such that $2^{k-1} \leq j < 2^k$, the logical slot indices from $j - 2^i + 1$ to j are sound.

Definition 2 defines the feasibility of the sequential logical slot indices in a recursive form where condition (2) guarantees the soundness of 2^i logical slot indices backward from j, recursively such that $(j), (j-1, j), (j-3, j-2, j-1, j)$, ..., and $(j - 2^i + 1, \ldots, j-1, j)$ are sound. However, this does not guarantee that $(j-2, j-1)$are sound, thereby requiring condition (1).

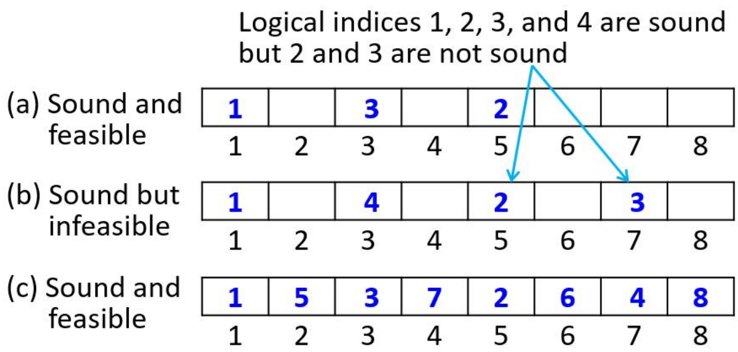


Figure 3.2. An example of soundness and feasibility of the sequential logical slot indices.

Take a look at Figure 3.2(a). According to Definition 1, the three assigned logical slot indices (1, 2, 3) are sound since j (= 3) falls in (2^1, 2^2] and each logical index belongs to one of 2^2 equally divided sections. According to Definition 2, they are feasible since when $j = 2$, (1, 2) is feasible and when $j = 3$, (2, 3) is recursively sound such that (3) and (2, 3) are sound. However, the logical slot indices (1, 2, 3, 4) in Figure 3.2 (b) are not feasible since condition (1) of Definition 2 is satisfied such that (1, 2, 3) is not feasible since (2, 3) is not sound. Figure 3.2(c) shows an example of a feasible sequence.

The design of the LSI algorithm follows Definition 2. The algorithm can guarantee condition (1) by repeating "*satisfying the recursive soundness*" given in condition (2) with the whole frame whenever it assigns a new logical index. To realize condition (2) in assigning a new index j, the algorithm selects the section with the smaller maximum index after dividing the whole section into two smaller sections, thereby making $(j-1, j)$ sound. If it performs this process one more time with the selected section, it can make $(j-3, j-2, j-1, j)$ sound. It can continue this process to guarantee the recursive soundness until it finds an index-free section or finds that the whole frame is fully indexed. The algorithm is detailed in Algorithm 1.

Algorithm 1: Logical slot indexing (LSI)

Input: A given *index-free* frame that has 2^N slots
Output: The frame with the feasible logical slot indices
\# lsi: A logical slot index
\# $selectedS$: A variable to select a section
\# $S^i(\alpha)$: The i[th] divided section whose max. logical index is α

$lsi \leftarrow 1$
while $S^0(lsi-1)$ is not fully indexed **do**
    $selectedS \leftarrow S^0(lsi-1)$ # the whole frame
    $i \leftarrow 1$
    while $selectedS$ is not index-free **do**
        divide $selectedS$ into $S^i(u)$ and $S^i(v)$
        if $u < v$ **then**
            $selectedS \leftarrow S^i(u)$
        else
            $selectedS \leftarrow S^i(v)$
        end if
        $i \leftarrow i+1$
    end while
    assign lsi to one arbitrary slot in $selectedS$
    $lsi \leftarrow lsi+1$

```
16:  end while
```

Let us take an example to see how the LSI algorithm works with a frame of $N = 3$. Without loss of generality, assume that the algorithm always selects the first slot within an index-free section to assign a new logical index. The algorithm starts with $lsi = 1$ at line 1. Then, the outer *while loop* in line 2 is repeated until the frame is fully indexed. First, the selected section becomes the whole frame $S^0(0)$ (line 3). Since the selected section is index-free, the logical index 1 is assigned to the first slot in the frame after the first round of the outer while loop. Next, the algorithm repeats the outer while loop to assign the logical slot index 2. The selected section, *selectedS*, becomes the whole frame $S^0(1)$ in line 3. Since it is not index-free, it is equally divided into two smaller sections, $S^1(1)$ with the first four slots and $S^1(0)$ with the later four slots (line 6). Then, the algorithm selects $S^1(0)$ since $0 < 1$ (line 8). Since the selected section is index-free, the algorithm exits the inner while loop (line 5). Then, $lsi(= 2)$ is assigned to the first slot of the selected section $S^1(0)$ (line 14), resulting in $S^1(2)$. Since the whole frame$S^0(2)$ is not fully indexed, the algorithm continues the outer while loop to assign the next $lsi(= 3)$. The final indexing result is given in Figure 3.3.

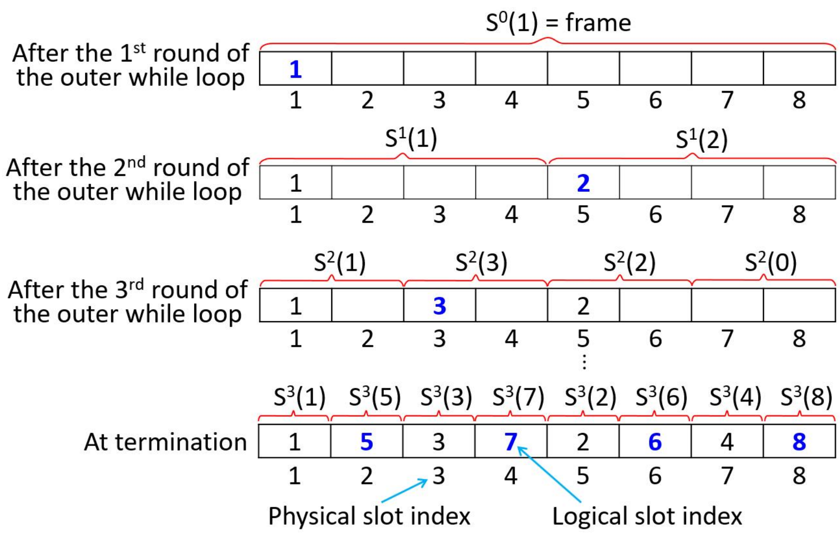


Figure 3.3. Execution of the LSI algorithm to assign the logical slot indices to a frame of 8 slots

Given a frame of n slots, the outer while loop is repeated n times, and the inner while loop is executed $log\ n$ times since the search space is reduced by half. In addition, in each iteration i of the inner while loop, $^{n}/_{2^i}$ comparisons for each of two divided sections are made to find the maximum slot index. Therefore, the time complexity function $T(n)$ of this algorithm can be expressed as follows: $T(n) = n\sum_{i=1}^{\log n}\frac{2n}{2^i}$. Therefore, $T(n) \in O(n^2)$. Note that the algorithm is executed once only if the frame size is redefined.

3.3 Lemmas and Theorems

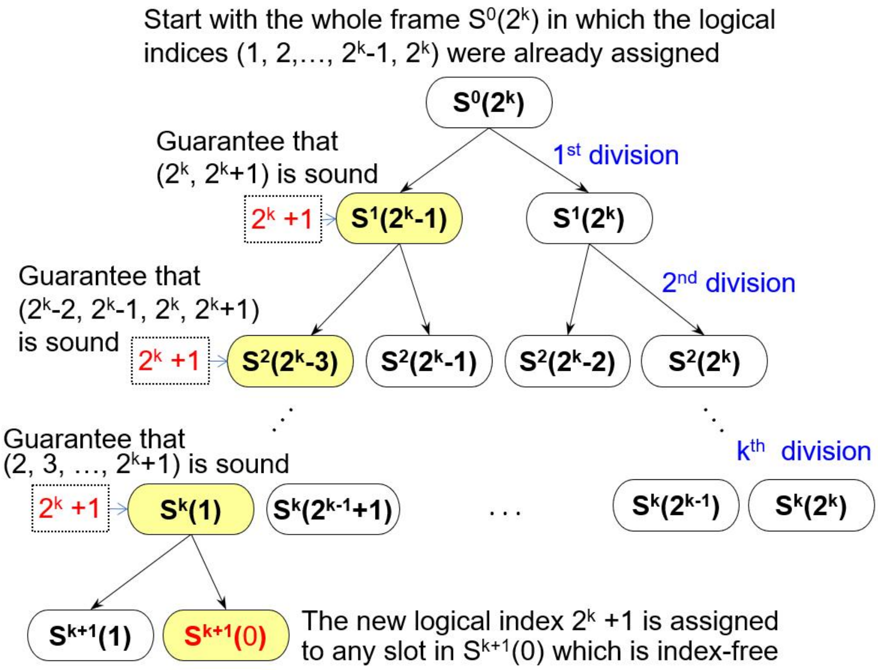


Figure 3.4. A process of assigning a new logical index $2^k + 1$ (the value in ( ) indicates the maximum logical index when the corresponding section is examined)

Lemma 1: Suppose that 2^k sequential logical indices are assigned and are feasible. Then, if the next logical index $2^k + 1$ is assigned by Algorithm 1, the logical slot indices $(2,3,\dots,2^k,2^k+1)$ are recursively sound.

Proof: Let us see how a logical slot index $2^k + 1$ is assigned by Algorithm 1, referring to Figure 3.4. It starts with the whole frame $S^0(2^k)$ in which 2^k sequential logical indices $(1,2,\dots,2^k - 1, 2^k)$ were already assigned. Since $S^0(2^k)$ is not index-free, it starts the inner while loop. It divides $S^0(2^k)$ into $S^1(2^k - 1)$ and $S^1(2^k)$ in line 6. This corresponds to 1st division in Figure 3.4. Then, it selects section $S^1(2^k - 1)$ that has the smaller maximum logical index to assign $2^k + 1$. This guarantees that $(2^k, 2^k + 1)$ is sound. However, if $S^1(2^k - 1)$ is not index-free, it has to be further divided into $S^2(2^k - 3)$ and $S^2(2^k - 1)$, resulting in the selection of $S^2(2^k-3)$ to assign 2^k+1, thereby making $(2^k-2, 2^k-1, 2^k, 2^k+1)$ sound (2nd division). The algorithm repeats the inner loop until it finds an index-free section. Continuing this process, the algorithm will assign 2^k+1 to any empty slot in $S^k(2^k-(2^k-1))$ (or $S^k(1)$) (k^{th} division), thereby making $(2, 3, \ldots, 2^k, 2^k+1)$ sound. Thus, we prove Lemma 1.

Lemma 2: If logical slot indices are assigned by Algorithm 1, 2^i sequential logical indices from j are feasible.

Proof: When $i=1$, we prove that 2^1 logical indices, j and $j+1$ assigned according to Algorithm 1 are feasible. Assume that the logical index j was already assigned to any empty slot in the frame by Algorithm 1. Then, the whole frame is $S^0(j)$. To assign the logical index $j+1$, the algorithm divides $S^0(j)$ into two smaller sections, $S^1(j-1)$ and $S^1(j)$ and tries to assign $j+1$ to $S^1(j-1)$ that has the smaller maximum logical index. This guarantees that $(j, j+1)$ are sound. Since j alone is feasible and $(j, j+1)$ is recursively sound, $(j, j+1)$ is feasible by Definition 2.

When $i=k$, we assume that 2^k logical indices from j to $j+2^k-1$ are feasible. When $i=k+1$, we prove that 2^{k+1} sequential logical indices starting with j are feasible.

By assumption, the sequence $(j, j+1, \ldots, j+2^k-1)$is feasible. Therefore, by Lemma 1, $(j+1, j+2, \ldots, j+2^k-1,\ j+2^k)$ is recursively sound. Since $(j, j+1, \ldots, j+2^k-1)$ is feasible and $(j+1, j+2, \ldots, j+2^k-1,\ j+2^k)$ is recursively sound, $(j, j+1, \ldots, j+2^k-1, j+2^k)$ is feasible by Definition 2. In the same way, $(j, j+1, \ldots, j+2^k-1, j+2^k, j+2^k+1)$ is feasible. Continuing this, 2^{k+1} logical indices $(j, j+1, \ldots, j+2^k-1, j+2^k, \ldots, j+2^{k+1}-1)$ become feasible. Thus, we prove Lemma 2.

Theorem 1: Consider a frame of 2^N slots logically indexed by Algorithm 1. Given a task $\tau_i = (i, p_i), p_i = {2^N}/{2^k}$ slots, $(0 \le k \le N)$, task τ_i can meet its deadline if it takes slots with 2^k sequential logical indices starting with any logical index.

Proof: Suppose that 2^k sequential slots from j to $j + 2^k - 1$ are selected. Then, by Lemma 2, the selected logical indices are feasible. This implies that the task is assigned one slot in each section of ${2^N}/{2^k}$ slots. Thus, if the task sends one data packet in each of the selected slots, the packet can be transmitted within its period p_i or before its deadline that becomes the start of next period. Thus, we prove Theorem 1.

Lemma 3: Given two overlapping frames, the first frame is logically indexed from 1 to 2^N and the second frame is logically indexed from $2^N + 1$ to 2^{N+1} by Algorithm 1, they can be treated as a concatenated frame in scheduling under the constraint that a task can select at most 2^N slots.

Proof: Suppose that the first 2^N logical slot indices in the first frame are denoted as $a_1, a_2, \dots, a_{2^N}$ and the last 2^N logical indices in the second frame are denoted as $b_1, b_2, \dots, b_{2^N}$.

Suppose that 2^k $(0 \le k \le N)$ sequential logical slots are selected such that x $(x \le 2^k)$ slots $(a_{2^N-x+1}, a_{2^N-x+2}, \dots, a_{2^N})$ belong to the first frame and next $2^k - x$ slots $(b_1, b_2, \dots, b_{2^k-x})$ belong to the second frame. We need to prove two things: (1) none of the selected slots is overlapped with any others and (2) they are sound. Since a_i is overlapped with b_i , $2^k - x$ selected slots on the second frame are equivalent to $(a_1, a_2, \dots, a_{2^k-x})$ on the first frame by replacing b_i by a_i. They become the combination of $(a_1, a_2, \dots, a_{2^k-x})$ and $(a_{2^N-x+1}, a_{2^N-x+2}, \dots ,\ a_{2^N})$ on the first frame. Since $2^k - x <$

$2^N - x + 1$ with any k $(0 \leq k \leq N)$, the selected slots never overlap with any others. Thus, (1) is true.

As in Lemma 1, the k^{th} division guarantees that $(a_1, a_2, \dots, a_{2^k})$ is sound and a_{i+2^k} is assigned to the same section with a_i. Thus, $(a_{i+1}, a_{i+2}, \dots, a_{i+2^k})$ is also sound on the same sections in which $(a_1, a_2, \dots, a_{2^k})$ is sound. Consequently, if $i = 2^N - 2^k$, $(a_{2^N-2^k+1}, a_{2^N-2^k+2}, \dots, a_{2^N})$ is sound on the same sections that $(a_1, a_2, \dots, a_{2^k})$ is sound. Therefore, the combination of selected slots $(a_1, a_2, \dots, a_{2^k-x})$ and $(a_{2^N-x+1}, a_{2^N-x+2}, \dots, a_{2^N})$ are also sound at the same sections since $x < 2^k$. Thus, (2) is true and we prove Lemma 3.

Theorem 2: Suppose that k frames are logically indexed from 1 to $k \cdot 2^N$ by Algorithm 1. Given a task, $\tau_i = (i, p_i)$, $p_i = {2^N}/{2^k}$ slots, $(0 \leq k \leq N)$, task τ_i can meet its deadline if it takes the slots with 2^k sequential logical indices from any logical index.

Proof: By Lemma 3, k frames can be treated as one concatenated frame in task scheduling. Thus, 2^k selected logical indices are sound. This implies that the task τ_i is assigned one slot in each section of ${2^N}/{2^k}$ slots and the packet can be transmitted within its deadline. Thus, we prove Theorem 2.

Chapter 4. Real-time LoRa Protocol with Logical Frame Partitioning

The *real-time LoRa protocol with logical frame partitioning* (*RTLoRa-LFP*) is explained with the design of frame-slot architecture and the method of slot scheduling that utilizes the logical slot indices for periodic tasks. Then, the *two-level collision avoidance* scheme is presented for the reliable data transmission of aperiodic tasks. For convenience, important notations used in this chapter are summarized in Table 4.1.

Table 4.1. Summary of notations

Notation	Meaning
Ch_i-*frame*	A frame defined on channel i
N	A frame factor used in defining a frame of 2^N slots
G_i	A group of periodic tasks that are scheduled on Ch_i-*frame*
$nSch_i$	The number of scheduled slots on Ch_i-*frame*
$nUSch_i$	The number of unscheduled slots on Ch_i-*frame*
T_i	A set of periodic tasks that have the transmission period of 2^i slots
$sLSI$	A start logical slot index to schedule new tasks
GSI	As group scheduling information, $GSI = (g, sLSI, T_0, T_1, \ldots, T_N)$ where g indicates a group number
$uTSI$	As updated task scheduling information, $uTSI = (i, (x, sLSI))$ where i and x indicate a group number and a task identification number, respectively
$d(\tau)$	Slot demand of periodic task τ per each frame period
$D(T_i)$	Slot demand of T_i per each frame period

$preD(\tau)$	Total slot demand of all periodic tasks that precede task τ in GSI
psi	Physical slot index as the index of array
$SS(\tau)$	Slot schedule of periodic task τ that is expressed as a set of physical slot indices
CW_i	Contention window defined by an aperiodic task after the i^{th} failure to get a channel
CW_i^e	Extended contention window corresponding to CW_i

4.1 Network Model

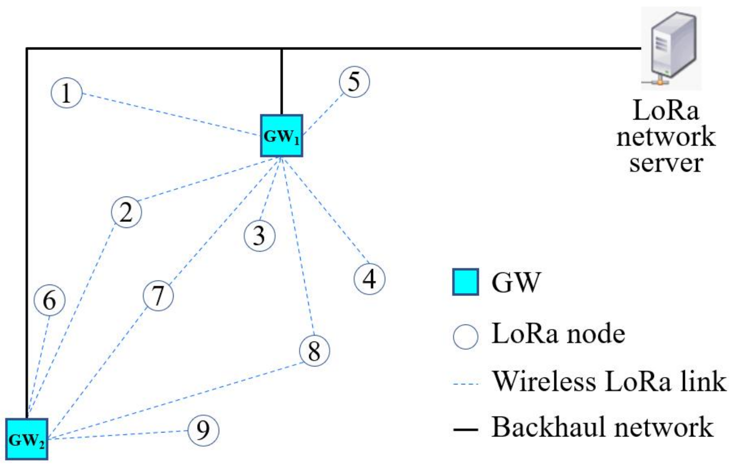


Figure 4.1. LoRa network model

A considered LoRa network consists of one LoRa network server, multiple gateways, and a number of end devices or nodes. A node can communicate directly with at least one of the GWs that connect to the LoRa server via a high speed backhaul network. Every node has one *periodic task* that transmits data periodically to GW and can activate an *aperiodic task* to transmit data when an event occurs. A periodic task has to transmit data before the beginning of the next period or deadline. Different tasks can have different transmission periods. If a server receives an identical data multiple times via different GWs, it takes one of them.

Figure 4.1 shows a simple LoRa network that includes one network server, two gateways as GW_1 and GW_2, and nine nodes. Three nodes 2, 7, and 8 are covered by both GW_1 and GW_2 connected to the server while other nodes are covered by one of them.

4.2 Frame-Slot Architecture

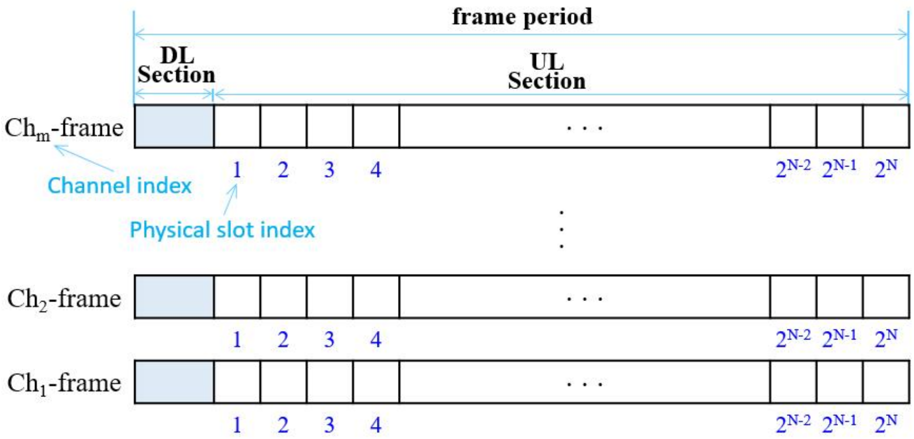


Figure 4.2. A multi-channel frame-slot architecture

With *m* available channels, *m* overlapping frames are defined for each frame period and each frame consists of a downlink (DL) section and an uplink (UL) section. A frame defined by channel Ch_i is denoted by Ch_i*-frame.* A server uses the DL section or DL slot to transmit a downlink message to nodes while nodes use the UL section to send periodic or aperiodic data. The UL section is further sliced into 2^N *data slots* where *N* as a *frame factor,* is an integer constant and each data slot is large sufficiently to send one data packet. Each frame is identified by a distinct channel, the data slots in the UL section of each frame are identified by the *physical slot indices*, numbered from 1 to 2^N sequentially, and each of the frames is used for the scheduling of tasks independently.

A server uses one specified channel to broadcast a DL message during the DL section while all nodes have to wait for the message on the channel. Note that the channel hoping can be easily used to improve the transmission reliability of the DL message. A frame-slot architecture using m channels is illustrated in Figure 4.2.

4.3 Logical Frame Partitioning

Apart from the physical slot indices, the data slots in each of the frames is assigned the same logical slot indices by the LSI algorithm. With m channels, all periodic tasks are divided into m groups as $G_1, G_2, \ldots, G_m$ and $G_i = (\tau_{i,1}, \tau_{i,2}, \ldots, \tau_{i,n_i})$ where $\tau_{i,j}$ is the j^{th} task of group G_i and n_i is the number of periodic tasks in G_i. If all periodic tasks in group G_i are scheduled, $nSch_i$ as the total number of scheduled slots of G_i, is given as follows:

$$nSch_i = \sum_{j=1}^{n_i} d(\tau_{i,j}) \leq 2^N \quad (4.1)$$

where $d(\tau_{i,j})$ is the slot demand of task $\tau_{i,j}$ per each frame period. Then, $nUSch_i$ as the total number of unscheduled slots on *Ch$_i$-frame* can be given as follows: $nUSch_i = 2^N - nSch_i$.

By the slot scheduling of periodic tasks, the data slots in *Ch$_i$-frame* are said to be *logically partitioned* such that $nSch_i$ data slots corresponding to the logical slot numbers from 1 to $nSch_i$ are scheduled and $nUSch_i$ data slots from $nSch_i + 1$ to 2^N are unscheduled. Note that $nSch_i$ implies the last scheduled slot in *Ch$_i$-frame*, and this way of partitioning the slots is referred to as a *logical frame partitioning*.

4.4 Slot Scheduling

Since tasks in one group are independent of ones in other groups, a slot scheduling of periodic tasks for all groups is identical. Thus, we describe a scheduling for only one group in this subsection. A server generates *group scheduling information GSI* for the slot scheduling of a group as follows:

$$GSI = (g, sLSI, T_0, T_1, \ldots, T_N)$$

where *g* indicates a group number, *sLSI* indicates the *start logical slot index* to start the generation of a slot schedule for the tasks, and T_i indicates a set of periodic tasks that has the same transmission period of 2^i slots expressed as follows:

$$T_i = \left(i, n_i, \left(\tau_{i,1}, \tau_{i,2}, \ldots, \tau_{i,n_i}\right)\right)$$

where $\tau_{i,j}$ is the j^{th} task of T_i and n_i is the number of tasks in T_i. Note that *sLSI* is always *unity* since the slot scheduling of every group starts with an empty schedule at the beginning.

For the slot scheduling of periodic tasks, a server distributes *GSI*. Upon receiving GSI, if a task finds itself in group *g*, it generates its slot schedule. $SS(\tau_{i,j})$ as a slot schedule for task $\tau_{i,j}$ can be obtained as follows:

$$SS(\tau_{i,j}) = \begin{pmatrix} psi\left(preD(\tau_{i,j}) + sLSI\right), psi\left(preD(\tau_{i,j}) + sLSI + 1\right), \\ \ldots, psi\left(preD(\tau_{i,j}) + sLSI + d(\tau_{i,j}) - 1\right) \end{pmatrix} \quad (4.2)$$

where $psi(v)$ indicates the *physical slot index* corresponding to the logical slot index v and $preD(\tau_{i,j})$ indicates the total slot demand of all tasks preceding task $\tau_{i,j}$ in *GSI* as follows:

$$preD(\tau_{i,j}) = \sum_{k=0}^{i-1} D(T_k) + \sum_{k=1}^{j-1} d(\tau_{i,k}) \quad (4.3)$$

where $D(T_k)$ indicates the total slot demand of all tasks in T_k.

Consider that a certain group includes 50 tasks (two bytes for each task identification number) with 5 classes of periods (1 byte for each class). Then, the size of GSI will be just 107 bytes.

If the size of GSI exceeds the size of data that a server can transmit in one DL slot, GSI can be divided into *k* subgroups, *sG*(1), *sG*(2), …, *sG*(*k*) with $GSI(1), GSI(2), \ldots, GSI(k)$, $k > 1$, respectively, to be transmitted separately. In this case, a server calculates *sLSI*(*i*) as the start logical slot index of subgroup *i* and includes it in *GSI*(*i*) before sending.

$$sLSI(i) = \sum_{j=1}^{i-1} D(sG(j)) \text{ and}$$
$$GSI(i) = (g, sLSI(i), T_0, T_1, \ldots, T_N) \tag{4.4}$$

where $D(sG(j))$ indicates the total slot demand of subgroup *i*. Upon receiving *GSI*(*i*), a task can generate its slot schedule by using (4.2) after replacing *sLSI* with *sLSI*(*i*).

Let us see an example for slot scheduling on a frame of 16 slots. Consider group g that has five periodic tasks such as task A with transmission period 4, tasks B and C with transmission period 8, and tasks D and E with transmission period 16. If $sLSI = 1$, *GSI*(*g*) is represented as follows:

$$GSI(g) = \big(g, 1, (4,1,(A)), (8,2,(B,C)), (16,2,(D,E))\big)$$

When a server distributes *GSI*(*g*), every node can generate its own slot schedule as given in Table 4.2.

Table 4.2. An example of a slot schedule

Task ID	Transmission period	Slot demand	Scheduled slot
A	4	4	1 to 4
B	8	2	5 to 6

C	8	2	7 to 8
D	16	1	9
E	16	1	10

The slot schedule for group *g* on the frame is illustrated in Figure 4.3. Note that each of tasks can satisfy its deadline by transmitting one data in each of the schedule slots.

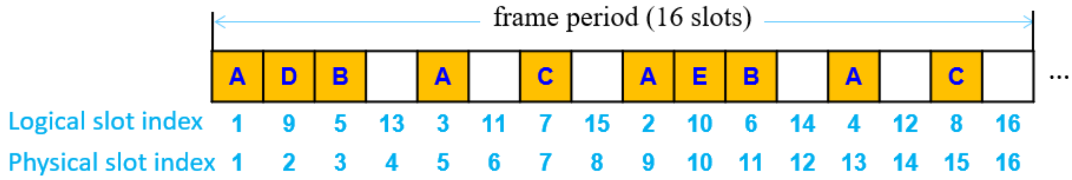


Figure 4.3. Slot scheduling example with group g of five periodic tasks (sLSI = 1).

4.5 Data Transmission and Maintenance

4.5.1 Periodic Data Transmission

During network initialization, a node registers with a server by sending its task information (for *x*, $\tau_x = (x, p_x)$) and waits for a DL message on the specified channel to receive the scheduling information from the server. If the server broadcasts GSI, every task in the group generates its own slot schedule in the corresponding frame and transmits data according to the slot schedule.

Suppose that a new periodic task, say *x*, wants to register with the server after network initialization. Then, if a server finds $d(x)$ sequential logical slot indices whose corresponding slots are all unscheduled in any frame, say Ch_i-*frame*, it gets *sLSI* as the first one of those logical slot indices and generates the following updated task scheduling information, *uTSI*:

$$uTSI = (i, (x, sLSI)\,)$$

where *sLSI* can be either $nSch_i + 1$ or a certain logical slot index k if $d(x)$ slots with the sequential logical slot numbers starting with *k* were previously freed due to the removal of a periodic task. Then, a server broadcasts *uTSI* so that task *x* can generate its slot schedule *SS*(*x*) as follows:

$$SS(x) = (psi(sLSI), psi(sLSI + 1), \dots, psi(sLSI + d(x) - 1)) \quad (4.5)$$

4.5.2 Aperiodic Data Transmission

Since an aperiodic task transmits data in unscheduled slots, it has to know the start logical slot index of unscheduled slots in the frame to which it belongs. For this, a server includes the logical frame partitioning information, $LFPi = (nSch_1, nSch_2, \dots, nSch_m)$ in a DL message so that an aperiodic task can know the last scheduled logical slot index in the frame to which it belongs.

An aperiodic task may have to compete with other aperiodic tasks within an unscheduled slot. In low-rate wireless networks, collision is costly because it can not only waste the precious bandwidth, but also can increase network traffic by retransmission. The IEEE 802.15.4 standard [20] deals with this problem by using the Slotted CSMA/CA mechanism based on an exponential backoff algorithm that uses the notion of *backoff period* as a delay unit (i.e., a slot corresponding to a time length of 20 symbols). Before transmitting a packet, every node takes $random(0, 2^{BE} - 1)$ backoff periods where *BE* is a backoff exponent, and then performs Channel Clear Assessment (*CCA*) twice to improve the correctness of checking due to the presence of the interframe space. A node that detects a busy channel takes another random backoff by incrementing *BE*. This process is repeated a specified number of times.

The collision problem becomes more serious in extremely low-rate LoRa networks characterized by a long packet *time on air* (*ToA*). It can be severer if more data slots have been scheduled for periodic tasks. Therefore, this requires a collision-avoidance scheme for the reliable data transmission of aperiodic tasks.

a) Contention Window and Delay Slot

Definition 3. *Given that a data slot is divided into a number of delay slots or delayslots numbered sequentially, the delayslot is the smallest time span such that if two neighboring nodes ready for data transmission generate delay slot numbers by the difference of unity and wait for the number of delayslots corresponding to the delay slot number before sending data, one with the larger number can overhear the signal issued by another.*

To effectively handle the collision problem in LoRa networks, this dissertation introduces the notion of *contention window* and *delayslot*. A *contention window* (*CW*) is a set of forthcoming unscheduled data slots that is determined after the arrival of an aperiodic task. The aperiodic task doubles the CW size if it fails to acquire a channel within CW. $|CW_i|$ as the size of contention window after the i^{th} failure to acquire a channel, is expressed as follows:

$$|CW_i| = \min(|CW_{i-1}| \times 2, MaxCW) \quad (4.6)$$

where *MaxCW* is the maximum size of contention window.

Table 4.3. Delayslots for different SFs (ms)

SF	7	8	9	10

k_i	2	2	4	4
$SymTime_i$	1.024	2.048	4.096	8.192
$delayslot_i$	2.048	4.096	16.384	32.768

BW=125kHz, CR=4/5

The *delayslot* includes the time to turn the radio chip on, the time to transfer a packet from a microprocessor to a radio chip buffer, and the time to perform *Channel Activity Detection* (CAD). The former twos are negligible compared to the last one in LoRa devices. Furthermore, according to the technical report from Semtech [50], experiments indicate that the CAD operation for only one symbol time can fail to detect the status of a channel correctly. Hence, they recommend the CAD operation for the time of multiple symbols depending on the spreading factors. Thus, $delayslot_i$ as delay slot with the use of SF i is given as follows:

$$delayslot_i = k_i \times SymTime_i \tag{4.7}$$

where k_i is a coefficient for SF i and $SymTime_i$ is the symbol time (in milliseconds) when SF i is used.Table 4.3 shows the values of *delayslot* for different SFs.

b) Two-Level Collision Avoidance Scheme

A *two-level collision avoidance* scheme is proposed based on contention window and *delayslot*. The *first level collision avoidance* (*L1-CA*) scheme employs an exponential backoff algorithm that uses a data slot as a unit backoff. If an aperiodic task is activated, it first generates the initial *contention window* (CW_0) and selects one data slot randomly within CW_0. However, if multiple tasks select the same unscheduled slot, their transmissions can result in collision. Thus, an aperiodic task is asked to acquire a channel before sending data. This channel contention process is said to be the *second level collision avoidance* (*L2-CA*) scheme. Let us defer the discussion of L2-CA for the time being. Whenever a task fails to acquire a channel within the selected data slot, it defers data transmission and doubles its contention window according to (4.6). A task gives up transmission after a specified number of failures. If it succeeds in sending, it changes its contention window back to CW_0.

Now, let us see how the L2-CA scheme works. The L2-CA scheme uses the notion of *rdelay*, that is defined as follows:

$$rdelay = \text{random}(0, MaxDelayCnt) \times delayslot \quad (4.8)$$

where $MaxDelayCnt$ indicates a maximum delay count. To avoid collision, every task takes delay for *rdelay* from the start of the selected data slot, and then performs CAD for one more delay slot. If the channel is idle, it sends data; otherwise, it defers data transmission and restarts L1-CA with the doubly increased contention window. Since the winning task has to complete data transmission within the remaining time of the selected slot, L2-CA does not use backoff. Instead, it limits the maximum delay time within the data slot by using the constant *MaxDelayCnt*. The operation of L2-CA is illustrated in Figure 4.4.

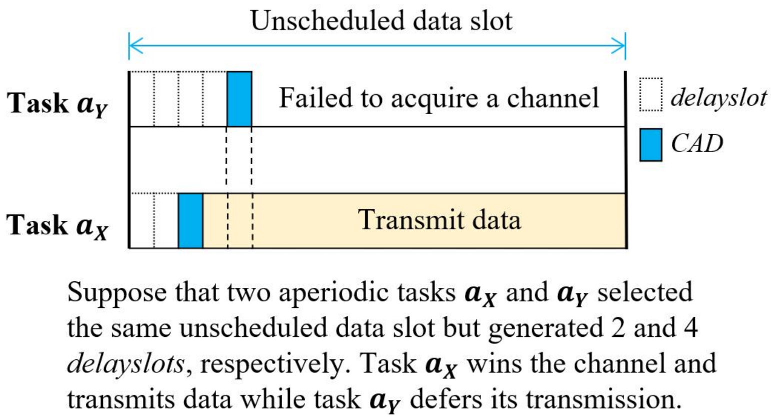


Figure 4.4. The operation of the L2-CA scheme

3) Extended Contention Window

Frames have the different distributions of unscheduled slots if groups have different slot demands. An aperiodic task can transmit data on any channel. Once an aperiodic task arrives, let us see how *CW* is calculated.

Given an aperiodic task $a_x = (x, t)$,

$$S(a_x) = \{u | startT(u) \geq t, u \text{ } is \text{ } an \text{ } unscheduled \text{ } slot\} \quad (4.9)$$

where $startT(u)$ *indicates the start time of slot u.*

In (4.9), $S(a_x)$ includes all the unscheduled slots that appear after time *t* on all frames. When $|CW_0| = k$, CW_i of an aperiodic task a_x after the *i^th^* failure to get a channel is calculated as follows:

$$CW_i \text{ } is \text{ } a \text{ } set \text{ } of \text{ } (2^i \times k) \text{ } unscheduled \text{ } slots \text{ } such \text{ } that \text{ } CW_i \subseteq S(a_x)$$
$$and \text{ } \forall u \in CW_i, \forall v \in (S(a_x) - CW_i), \text{ } startT(u) \leq startT(v). \quad (4.10)$$

According to (4.10), a contention window may have to exclude some unscheduled slots that have the same start time, but belong to different frames. However, it seems that such exclusion is not desirable. Thus, CW_i^e as an extended *CW* after a_x takes the i^{th} failure is calculated as follows:

CW_i^e *of aperiodic task* a_x *is a set of unscheduled slots such that*

$$CW_i^e = \{u | u \in S(a_x), startT(u) \leq \ maxStartT\} \quad (4.11)$$

where $maxStartT = max\{startT(u) |\ u \in \ CW_i\}$

In implementation, CW_i^e can be expressed as a set of pairs of *channel* and *psi*, and obtained as follows. If an aperiodic task arrives at time *t*, each unscheduled slot after time *t* is examined for each channel *ch* and (*ch*, *psi*) is included in CW_i^e if the logical slot index corresponding to *psi* is greater than the last scheduled logical slot index. This process repeats until the size of CW_i^e is greater than or equal to the current contention window size. This is given in Algorithm 2.

Algorithm 2. Get extended contention window

// CH = a set of available channels
// CWsize = contention window size after the i^{th} *failure*
// firstpsi = the first physical slot index right after the i^{th} *failure*
// lsi(k) = logical slot index corresponding to physical slot index k
1. $CW_i^e = \emptyset; psi = firstpsi;$
2. **while** $|CW_i^e| < CWsize$ **do**
3. $\quad CW_i^e = CW_i^e \cup \{(c, psi) \mid lsi(psi) > nSch_c, c \in CH\};$
4. $\quad psi = psi + 1;$
5. $\quad$ **if** $psi > 2^N$ **then** $psi = 1;$
6. **endWhile**

For example, consider task a_x that arrives at the down-arrowed time on node *x* as shown in Figure 4.5. Let us calculate CW_0^e with the contention window size of 4. Figure 4.5 illustrates two frames over two continuous frame periods. Each frame consists of 8 slots in which four logical slots, 1, 2, 3, and 4, and two logical slots, 1 and 2, are scheduled for periodic tasks in G_1 and G_2, respectively. Suppose that an aperiodic task a_x arrives at time *t*. Then, according to Algorithm 2, *firstpsi* is 7 and $CW_0^e = \{(2,7),(1,8),(2,8),(1,2),(2,2)\}$.

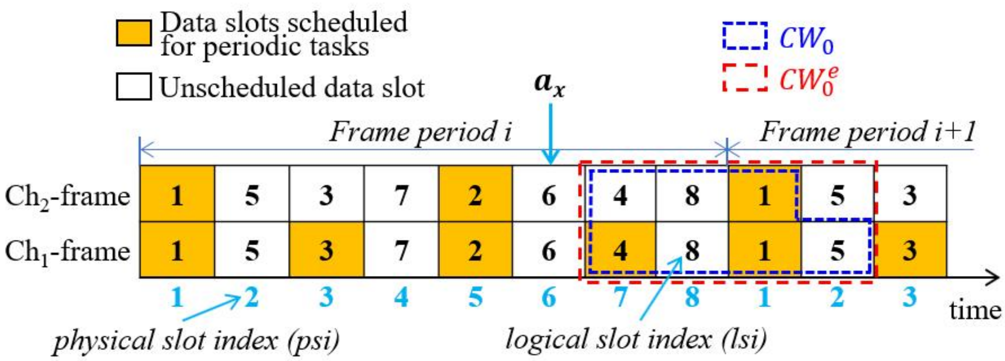


Figure 4.5. Extended contention window on multichannel frames with the initial contention window size of 4.

If CW_0^e includes elements past one frame period, some elements can be duplicated. This problem can be easily resolved by including *frame period (fp)* in a tuple as (*fp*, *ch*, *psi*).

Figure 4.6 illustrates the two-level collision avoidance scheme when two aperiodic tasks a_X and a_Y compete for data transmission. Suppose that tasks a_X and a_Y arrive at the beginning of the frame period. According to the L1-CA scheme, if they select different unscheduled data slots within CW_0^e, both will succeed in transmission. Even though they happen to select the same unscheduled data slot, say the third one in the figure, they can have another chance to avoid collision by using the L2-CA scheme.

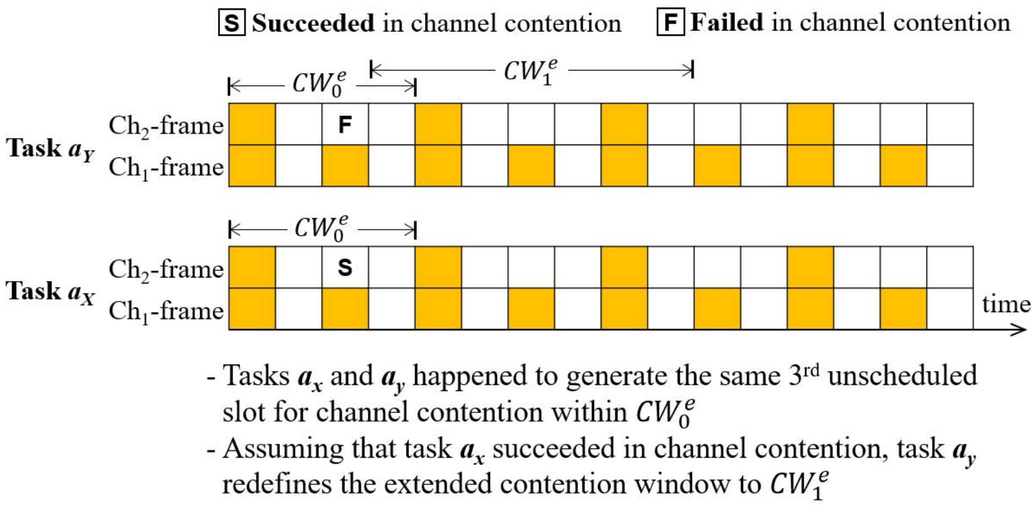


Figure 4.6. Illustration of the two-level collision avoidance scheme with the initial contention window size of 4.

If task a_X generates *rdelay* smaller than task a_Y, it acquires channel 2 (*Ch2-frame*) and succeeds in sending data while task a_Y defers its transmission and executes the L1-CA scheme again. If the two tasks generate the same *rdelay* during the L2-CA, they will transmit data concurrently and one of them can be received correctly by GW due to the capture effect.

4.6 Slot utilization

Let us compare the *slot utilization* (*SU*) of periodic tasks for two approaches, ZFD and SFD. In this comparison, we consider only one group since different groups take the same scheduling process. To enable a real-time slot scheduling for every periodic task, frame period T_{frm} using ZFD is determined as the shortest one of the transmission periods of periodic tasks as follows:

$$T_{frm}(ZFD) = \min\{p_i |\ \tau_i \ \in G\} \tag{4.12}$$

where $p_i = \alpha_i \times T_{frm}(ZFD)$, $\alpha_i \in N^+$. Given *n* tasks in G, $SU(ZFD)$ with ZFD can be calculated as follows:

$$SU(ZFD) = {}^{1}\!/_{\mathrm{n}} \sum_{j=1..n} {}^{1}\!/_{\alpha_j} \tag{4.13}$$

Just for the convenience of analysis, we assume that a frame period with SFD is the longest one of the transmission periods of tasks as follows:

$$T_{frm}(SFD) = \max\{p_i \mid \tau_i \in \mathrm{G}\} \tag{4.14}$$

Then, each task acquires the exactly required number of slots within the frame period by using the LSI algorithm. Thus, no transmission slot is wasted regardless of the differences of transmission periods. Thus, since SFD does not waste any slot, $SU(SFD)$ is given *unity*.

$$SU(SFD) = 1 \tag{4.15}$$

Let us take an example to compare the channel utilization for the two approaches. Consider two task groups as follows:

$$G_1 = (\tau_A, \tau_B, \tau_C) = \big((\mathrm{A}, 8), (\mathrm{B}, 8), (\mathrm{C}, 8)\big) \text{ and}$$

$$G_2 = (\tau_X, \tau_Y, \tau_Z) = \big((\mathrm{X}, 8), (\mathrm{Y}, 16), (\mathrm{Z}, 32)\big).$$

The frame period and slot utilization of each approach are calculated in Table 4.4.

Table 4.4. Slot utilization of ZFD and SFD for two task groups.

	$T_{frm}(ZFD)$	$T_{frm}(SFD)$	$SU(ZFD)$	$SU(SFD)$
G_1	8	8	1	1
G_2	8	32	0.58	1

Since the transmission periods of all tasks in G_1 are 8 slots long, both ZFD and SFD set T_{frm} to 8. Thus, both approaches can fully utilize the slots. With G_2, ZFD sets T_{frm} to 8 while SFD does it to 32. Thus, with ZFD, task τ_Y and task τ_Z uses one slot every two and four frame periods, wasting one and three slots, respectively. Meanwhile, SFD still can achieve maximum slot utilization.

Chapter 5. Performance evaluation

5.1 Periodic Data Transmission

5.1.1 Experimental Setup

Experiment was performed in the testbed of one GW and fifteen nodes with STM32 microcontroller and SX1276 radio chip. It is claimed that the use of one channel and one SF is sufficient to examine the advantages of the protocols comparatively. Nodes were deployed over one three-floor building at the University of Ulsan. Three protocols, *LoRaWAN*, *Slotted Aloha*, and *RT-LoRa-LFP*, were compared for their PDRs. The key parameters and values used in the experiments are summarized in Table 5.1 where the packet size includes a payload of 25 bytes and the header of 8 bytes. Each node has one periodic task that transmits one packet to GW with transmission period, p_{tx}, of 1.5 s, 3 s, 6 s, or 12 s without external interference.

Table 5.1. Experimental parameters and values

Parameter	Value	Parameter	Value
Number of nodes	15	Spreading factors	SF_7
Packet size	33 bytes	Bandwidth	125 kHz
Number of channels	1	Code rate	4/5
Tx power	14 dBm	Preamble	8 symbols

Since this experiment is concerned with internal interference, SF_7 only was used and every node was checked beforehand to make sure that its link to GW is good.

5.1.2 Results and Discussion

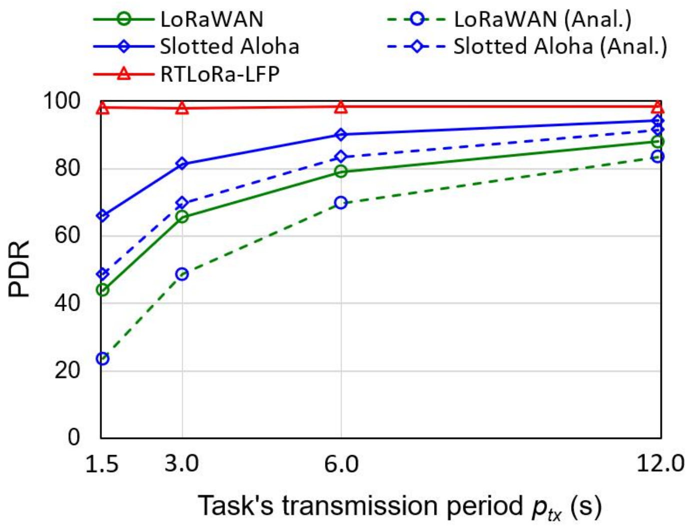


Figure 5.1. Comparison of PDRs for different protocols with varying node Tx interval without interfering nodes

Figure 5.1 compares PDRs for three protocols. It is shown that RTLoRa-LFP achieves PDR of 100% while LoRaWAN and Slotted Aloha decrease PDR sharply as traffic increases. Note that the PDR of LoRaWAN decreases down to 44% when every node has Tx interval of 1.5 s. The figure also examines the soundness of the experimental data by comparing them with the analytical data for LoRaWAN and for Slotted Aloha (that given in (A1.3) and (A1.4), respectively, in Appendix 1) with packet time-on-air $T_P = 72\ ms$ and the frame period $T_{frm} = p_{tx}$. Two dashed curves indicate the analytical data for LoRaWAN and Slotted Aloha, respectively. Analytical graphs for LoWaWAN and Slotted Aloha show their PDRs less than their experimental data by 45% and 27%, respectively at the high traffic of $Tx\ interval = 1.5$. This is because the analytical model does not take into account capture effect. According to one study [32], it was shown that PDRs by analytical model are lower than ones by simulation that reflects capture effect by up to 40% depending on traffic. However, note that they have similar decreasing patterns.

It was shown by the experiment results that RTLoRa-LFP could far outperforms two compared protocols, LoRaWAN and Slotted Aloha, in transmitting periodic data. RTLoRa-LFP can achieve almost 100% PDR since it relies on slot scheduling for data transmission while other ones that employ the Aloha protocol suffer from collisions.

5.2 Aperiodic Data Transmission

In this subsection, we focus on evaluating the performance of aperiodic data transmission. The proposed protocol, RTLoRa-LFP, that uses the SFD approach, was compared with ILoRa [23] and RT-LoRa [24] that use the ZFD approach. The compared protocols are implemented as follows. The ILoRa protocol uses the Pure Aloha approach for the transmission of aperiodic data. Whereas, the RT-LoRa protocol employs the Slotted Aloha approach in which if an aperiodic task arrives within CFP, it selects a transmission slot randomly within the forthcoming CAP to avoid collision and transmits data in the selected slot. On the contrary, if an aperiodic task arrives within CAP, it selects the next slot to transmit data due to the nature of random arrival.

5.2.1 Simulation Model

For performance evaluation, a simulation tool was developed for network simulation based on the well-known discrete event simulator, Simpy [51]. Since Simpy does not provide LoRa simulation model, we borrowed the LoRa communication model that defines communication range and collision behavior from LoRaSim [32]. In LoRaSim, the communication range for both gateway and end node are modeled using the following log-distance path loss model:

$$L_{pl}(d) = \overline{L_{pl}}(d_0) + 10\gamma log\left(\frac{d}{d_0}\right) + X_\sigma \qquad (5.1)$$

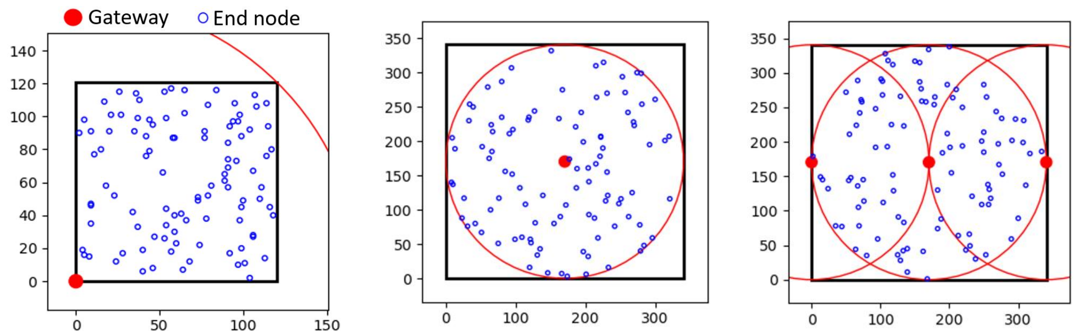
where *d* is the distance from the node to a gateway, $L_{pl}(d)$ is the path loss in dB, $\overline{L_{pl}}(d_0)$ is the mean path loss at the reference distance d_0, γ is the path loss exponent, and X_σ takes the normal distribution with zero mean and σ^2 variance to account for shadowing, represented as $X_\sigma \sim N(0, \sigma^2)$. The empirical measurements on a real node with $d_0 = 40$ give the parameter values in (5.1): $\overline{L_{pl}}(d_0) = 127.41\ dB, \gamma = 2.08$ and $\sigma = 3.57$. In this model, σ is assumed to be zero for simplicity. If a node transmits data at the distance of *d* from GW, GW calculates the reception power based on the transmission power and $L_{pl}(d)$ from (5.1). Then, it is determined that GW receives data successfully if its reception power is greater than the sensitivity threshold of the receiver.

We also borrow the collision behavior *C*(*x*, *y*) model of two concurrently transmitted data *x* and *y* as follows:

$$C(x,y) = O(x,y) \wedge C_{freq}(x,y) \wedge C_{sf}(x,y) \wedge C_{pwr}(x,y) \wedge C_{cs}(x,y) \qquad (5.2)$$

where $O(x,y)$ is true if their reception intervals overlap in any amount, $C_{freq}(x,y)$ is true if the overlapping of their frequencies is larger than the threshold value, $C_{sf}(x,y)$ is true if they use the same spreading factor, $C_{pwr}(x,y)$ is true if the difference of their reception powers is larger than the threshold value, and $C_{cs}(x,y)$ is true if the overlapping of their reception intervals is larger than the specified value. This model implies that packets *x* and *y* always collide if all terms in the right side of (5.2) are true; otherwise, one or two of them can be received depending on the condition of each term.

5.2.2 Simulation Setup



(a) S1 (b) S2 (c) S3

Figure 5.2. Three deployment scenarios for performance evaluation

One key factor that degrades the reliability of transmission in LoRa networks is the high probability of collision. To resolve this problem, the proposed approach employs the collision avoidance scheme using the LBT mechanism. However, the LBT mechanism has limitation in dealing with the hidden node problem. Another factor to consider is *node density*, as the number of nodes (*nNodes*) divided by the size of dimension, that determines data traffic. In industrial monitoring applications, nodes are often distributed densely within the limited working area. The use of multiple GWs may alleviate the packet loss problem by high traffic.

As depicted in Figure 5.2, three scenarios, S1, S2, and S3 were used to examine how well the compared LoRa protocols can deal with node density and the hidden node problem. In S1, GW is located at the corner and end nodes are randomly distributed in a square area such that one GW covers all end nodes, where the red curve line indicates the communication range of the GW. This scenario excludes the hidden node problem completely that occurs when two nodes transmit data packet to GW, but are not within a mutual communication range. In S2 with the bigger dimension, GW is located at the center and end nodes are distributed randomly in a circled area such that GW covers all the nodes. In S3, two GWs are additionally placed at the diametrically opposite points on the circumference of a circle to reduce the number of nodes exposed to the hidden node problem.

To better examine the capability of different MAC protocols in dealing with aperiodic tasks, we introduce one parameter to show a scheduled slot index, *ssi*, that is used to indicate the portion of scheduled slots to total slots as follows:

$$ssi = \frac{nSch_i}{2^N} \leq 1 \quad (5.3)$$

Since the higher *ssi* value implies the smaller number of unscheduled slots and thus will intensify the channel contention of aperiodic tasks, it can better reveal a good scheme in handling the collision and the hidden node problem.

Table 5.2. Simulation parameters and values.

Parameter	Value	Parameter	Value
nNodes	50 ~ 600	*N*	8
m	1	UL slot size	100 ms
Spreading factor	7	DL section	200 ms
Code rate	4/5	Frame period	25.8 s
Bandwidth	125 kHz	k_i (SF7)	2
Tx Power	14 dBm	$(\lvert CW_0\rvert, MaxCW)$	(4, 64)
Packet size	35 bytes	*MaxDelayCnt*	10
$1/\lambda$	25.8 (s)	CE threshold	3 dBm

Other parameters and values are summarized in Table 5.2. The frame period is 25.8 seconds, and the number of UL slots and the number of DL slots are 256 (25,600 milliseconds) and 1 (200 milliseconds), respectively. It is assumed that each node generates one aperiodic data whose arrival rate follows the Poisson distribution with a specified rate λ. The capture effect can be enabled or disabled according to the objective of evaluation, and if enabled, its threshold is specified to judge the occurrence of collision. In this study, the threshold was set to 3 dBm according to [10] while LoRaSim uses 6 dBm.

5.2.3 Results and Discussion

a) Effect of Contention Window

Simulation was performed with S1 to investigate the sensitivity of RTLoRa-LFP with varying *nNodes* from 50 to 600 and with varying the maximum allowable number of channel contentions (*MaxCCs*) from 1 to 7 before giving up data transmission. This scenario prevents the hidden node problem. The *ssi* is set to *zero*, implying that periodic tasks do not exist.

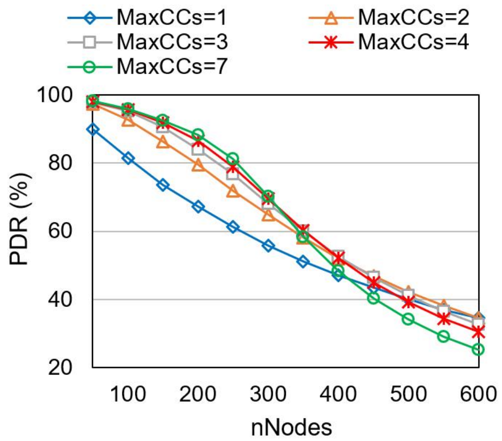


Figure 5.3. Evaluation of PDR of RT-LoRa-LFP with varying nNodes and MaxCCs (S1, ssi = 0)

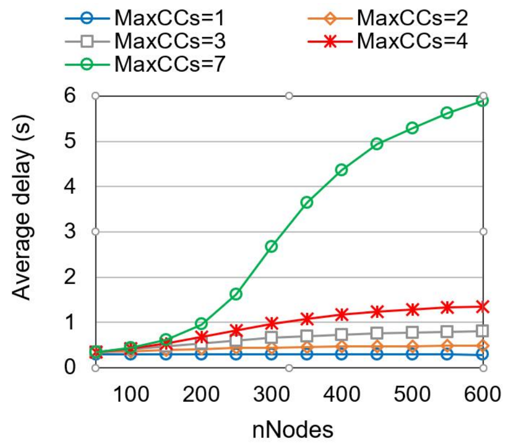


Figure 5.4. Evaluation of average delay of RT-LoRa-LFP with varying nNodes and MaxCCs (S1, ssi = 0)

Referring to Figure 5.3 and Figure 5.4, as *nNodes* goes over 250, the larger *MaxCCs* makes both PDR and delay worse drastically. This comes from the principle "the more the number of reattempts of channel contention, the more the number of competing nodes". On the contrary, the protocol with *nNodes* of 256 or less shows the higher PDR for the bigger *MaxCCs* since it gives more chances to a node that have failed channel contention. If *nNodes* is 100 and *MaxCCs* is 3, the protocol can achieve PDR over 95% while it limits the delay below 300 milliseconds. Considering PDRs and delay, the optimal value of *MaxCCs* is chosen as 3 or 4. *MaxCCs* is set to 4 for the rest of discussion.

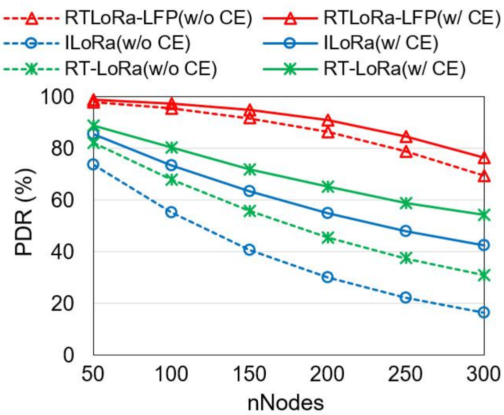


Figure 5.5. PDRs of different protocols with capture effect enabled or disabled (S1, ssi = 0)

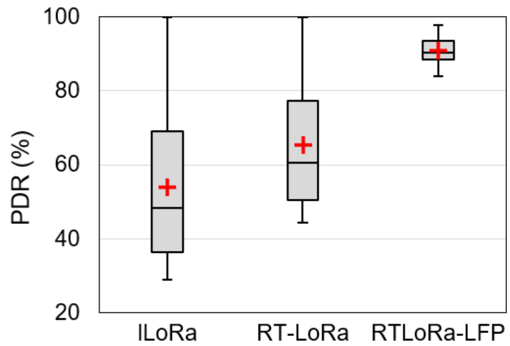


Figure 5.6. The PDR distribution of nodes for different protocols (nNodes = 200, S1, ssi = 0)

Figure 5.5 shows which protocol can better avoid collision among three protocols, RTLoRa-LFP, ILoRa, and RT-LoRa. Each protocol shows two graphs corresponding to two modes, one with the capture effect enabled (w/ CE) and another with the capture effect disabled (w/o CE). Recall that if two or more nodes transmit data concurrently, GW demodulates only the data with the strongest signal by the capture effect. RTLoRa-LFP shows a small gap of PDRs between two modes while the other protocols show the big gap.

This indicates that RTLoRa-LFP does not allow concurrent transmissions much. It is worth noting that the other approaches rely on the capture effect considerably.

Figure 5.6 shows that the PDR distribution of nodes varies from 29% to 100% and 44% to 100% with ILoRa and RT-LoRa, respectively, while that varies from 84% to 98% with RTLoRa-LFP. This implies that RTLoRa-LFP achieves high fairness in data transmission.

b) Effect of Logical Frame Partitioning

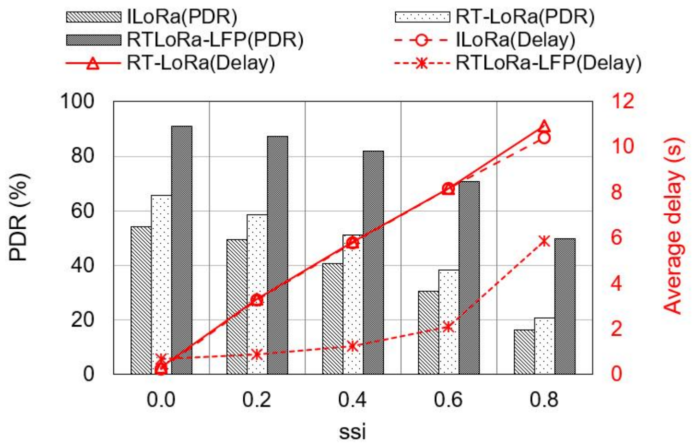


Figure 5.7. PDRs and average delay according to the increase of ssi (nNodes = 200, S1)

Simulation was performed to evaluate the effect of the frame partitioning scheme by using different *ssi* values. As shown in Figure 5.7, all three protocols show the decreasing PDRs by increasing the *ssi* value since the reduced number of unscheduled slots will increase the degree of channel contention. However, RTLoRa-LFP outperforms other approaches clearly owing to the use of the two-level collision avoidance scheme.

As for the transmission delay, when *ssi* = 0.0, both ILoRa and RT-LoRa show delay slightly lower than RTLoRa-LFP since they can transmit aperiodic data immediately at their arrival times. As the size of CFP increases with the increase of *ssi*, aperiodic data that arrive during CFP cannot be transmitted before the start of CAP, thereby increasing delay. However, with the RTLoRa-LFP protocol, since unscheduled slots are distributed over the frame, the waiting times are much smaller than those of the other two's on average. Note that the RTLoRa-LFP curve of average delay is sustained almost flat till *ssi* increases to 0.6. For the high *ssi* value of *0.9*, we summarize PDR and delay in Table 5.3 due to the scaling problem of the large values.

Table 5.3. PDRs and average delay with ssi = 0.9.

	ILoRa	**RT-LoRa**	**RTLoRa-LFP**
PDR	8.1	10.2	31.7
Delay	8.0	10.5	43.0

If *ssi* increases to the extremely large value of 0.9, the number of unscheduled slots is only 25 slots in a frame of 256 slots used in this simulation. The average delay of RTLoRa-LFP increases sharply up to 43 seconds while that of the other ones decreases slightly. This is because RTLoRa-LFP tries to transmit data over multiple frame periods, thereby achieving relatively high PDR and long delay; however, the other ones achieve low PDRs because most of packets are lost early due to collisions.

c) Effect of Multiple GWs

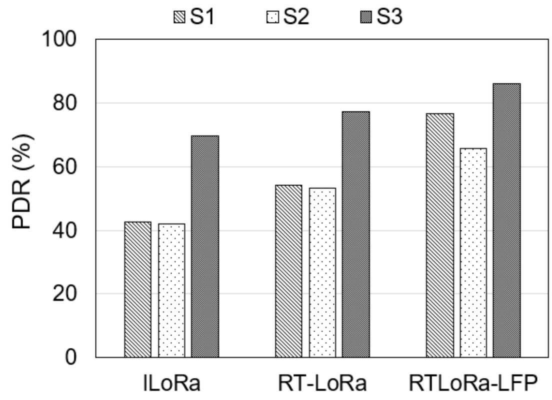


Figure 5.8. PDRs for different scenarios (nNodes = 200)

Simulation for three protocols was performed to evaluate the effect of using multiple GWs with different scenarios S1, S2, and S3. Note that a node succeeds in data transmission if any of GWs receives the data in LoRa networks. Therefore, with the use of more GWs, nodes better overcome two interference types, one by neighboring nodes and another by hidden nodes.

First, let us compare ILoRa and RT-LoRa in S1 and S2. ILoRa and RT-LoRa employs Pure Aloha and Slotted Aloha, respectively, that do not distinguish the interference types. Thus, referring to Figure 5.8, it is obvious that RT-LoRa achieves PDR higher than ILoRa while each of them achieves almost the same PDRs for S1 and S2. Second, as for RTLoRa-LFP, PDR in S2 is lower than that in S1 by almost 10%. This is because many nodes in S2 are exposed to the hidden node problem. It is also observed that RTLoRa-LFP achieves much higher PDR than the other ones. This implies that the two-level collision avoidance scheme can handle the collision problem. On the other hand, three protocols all improve PDRs in S3 considerably compared to those in other two scenarios. For ILoRa and RT-LoRa, the improvements come from the use of two additional GWs and capture effect while for RTLoRa-LFP, the reason is that many nodes are disengaged from the possibility of the hidden node problem due to two additional GWs, and the data transmissions of those nodes are treated by the two-level collision avoidance scheme.

In summary, it was shown by simulation that the proposed protocol could far outperform the other ones in reliability and average transmission delay, and also could greatly improve the unfairness of aperiodic data transmission that is caused by the capture effect. Furthermore, simulation results showed that the use of multiple GWs could alleviate the hidden node problem considerably.

Chapter 6. Concluding Remarks

6.1 Dissertation conclusion

In this dissertation, a real-time LoRa protocol with logical frame partitioning, RTLoRa-LFP, was proposed for the use in industrial monitoring and control applications that require the support for both periodic and aperiodic data transmission. It used a real-time slot scheduling algorithm, supported by the *logical slot indexing* algorithm. The logical slot indices enable the efficient generation of a schedule for periodic real-time tasks. The RTLoRa-LFP protocol not only guarantees that periodic tasks finish their data transmission within their respective time constraints, but also allows aperiodic tasks to transmit their data with high reliability and short delay. The former real-time support is achieved by using the real-time slot scheduling while the latter improvements are achieved by employing the two-level collision avoidance scheme using contention window and delay slot. The proposed protocols are expected to satisfy the requirements of the target application with some salient achievements as follows:

- The RTLoRa-LFP protocol not only guarantees that periodic tasks finish their data transmission within their respective time constraints, but also allows aperiodic tasks to transmit their data with high reliability and short delay. The former real-time support is achieved by using the real-time slot scheduling while the latter improvements are achieved by employing the two-level collision avoidance scheme using contention window and delay slot.

- The experiment results show that RTLoRa-LFP could far outperforms two compared protocols, LoRaWAN and Slotted Aloha, in transmitting periodic data. RTLoRa-LFP can achieve almost 100% PDR for periodic data transmission since it relies on slot scheduling for data transmission.
- The simulation results show that the proposed protocol could far outperform the other ones in reliability and average transmission delay. Specifically, with 200 aperiodic tasks that transmit data at 25.8 seconds in average, RTLoRa-LFP still can achieve 90% PDR while two compared protocols, ILoRa and RTLoRa, can achieve only 57% and 65% PDR, respectively.
- It also shows that the proposed protocol can better improve the transmission delay of aperiodic task as the size of CFP increases. As the number of scheduled slot (in CFP) increases from 0% to 40%, RTLoRa-LFP remains flat average delay curve with maximum delay is 1.9 seconds. Meanwhile, the other protocols show sharply increased delay curve with maximum delay is 5.8 seconds.
- Finally, the simulation results also show that RTLoRa-LFP could greatly improve the unfairness of aperiodic data transmission that is caused by the capture effect. The PDR distribution of nodes of RTLoRa-LFP varies within a small range from 84% to 98% while that of ILoRa and RT-LoRa vary from 29% to 100% and 44% to 100%, respectively.

It was shown with various experiment and simulation scenarios that the proposed protocol can not only guarantee the reliability and timely delivery of periodic data, but also deal with aperiodic data with high reliability, fairness, and low delay compared with other recent protocols. This approach can be considered for the standardization process of real-time IoT networks since it provides a real-time data transmission service in LoRa networks.

6.2 Future research direction

Even though the design of a real-time LoRa protocol has gain a lot of attention by researchers in the last few years, many interesting aspects of the real-time LoRa protocol still remain to be investigated. There is still a lot of room to deeply dig in the problem of the transmission of both periodic and aperiodic tasks.

Our target in the future is to implement the data transmission of aperiodic task on the real device to further investigate the performance of the network. Moreover, we are researching to find the algorithms to improve the protocol so that it can better support the downlink message transmission from the server to end devices. Finally, another possible research direction is the integration of the LoRa network and the wireless sensor network to support more applications in industrial fields.

Appendix

Appendix 1

In the Poisson process, given a packet arrival rate λ , $P(k)$, the probability that k packets will arrive in time t, is given as:

$$P(k\ attempts\ in\ time\ t)\ = \frac{(\lambda t)^k}{k!} e^{-\lambda t} \tag{A1.1}$$

Let T_P denote one packet transmission time that is assumed to be equal to a slot time. Suppose that each node transmits one packet during a frame period, T_{frm}. Then, this transmission model can be transformed into the Poisson process with the mean probability of T_P/T_{frm}. Then, for *n* nodes, the packet arrival rate λ can be expressed as:

$$\lambda = \frac{nT_P}{T_{frm}} \tag{A1.2}$$

Since LoRaWAN follows the Aloha protocol in data transmission, the vulnerable periods of a data packet using LoRaWAN and Slotted Aloha correspond to $2T_P$ and T_P, respectively [52]. Let $P_0(X)$ denote the probability that a packet is transmitted successfully to a server for protocol X. By applying the average arrival rate in (A1.2) to (A1.1), we get $P_0(X)$ as follows:

$$P_0(LoRaWAN) = P(k = 0\ in\ 2\ time\ slots) = e^{-\frac{2nT_P}{T_{frm}}} \tag{A1.3}$$

and

$$P_0(Slotted\ Aloha) = P(k = 0\ in\ 1\ time\ slot) = e^{-\frac{nT_P}{T_{frm}}} \tag{A1.4}$$

Publications

Journals

1. Q. L. Hoang, W. Jung, T. Yoon, D. Yoo, and H. Oh, "A Real-Time LoRa Protocol for Industrial Monitoring and Control Systems," *IEEE Access,* vol. 8, pp. 44727-44738, 2020.
2. Q. L. Hoang, H. P. Tran, W.-S. Jung, S. H. Hoang, and H. Oh, "A Slotted Transmission with Collision Avoidance for LoRa Networks," *Procedia Computer Science,* vol. 177, pp. 94-101, 2020/01/01/ 2020.
3. C. T. Ngo, Q. L. Hoang, and H. Oh, "Evaluation of a Compound Node Selection Function for a Greedy Routing in Urban Vehicular Ad Hoc Networks," *Procedia Computer Science*, vol. 134, pp. 259-266, 2018.

International Conferences

1. Q. L. Hoang, V. N. Q. Bao, H. Oh, "Performance Evaluation of LoRa Networks for Confirmed Messages, ", *in 2021 International Conference on Advanced Technologies for Communications (ATC)*, 2021, pp. 162-166.
2. C. T. Ngo, Q. L. Hoang, and H. Oh, "Bssa ch: a big slot scheduling algorithm with channel hopping for dynamic wireless sensor networks," in *International Conference on Ad-Hoc Networks and Wireless*, 2017, pp. 359-366: Springer.

Bibliography

[1] M. Bor, J. Vidler, and U. Roedig, "LoRa for the Internet of Things," *EWSN,* pp. 361-366, 02/15 2016.

[2] B. Reynders and S. Pollin, "Chirp spread spectrum as a modulation technique for long range communication," in *2016 Symposium on Communications and Vehicular Technologies (SCVT)*, 2016, pp. 1-5.

[3] M. A. Ertürk, M. A. Aydın, M. T. Büyükakkaşlar, and H. Evirgen, "A Survey on LoRaWAN Architecture, Protocol and Technologies," *Future Internet,* vol. 11, no. 10, p. 216, 2019.

[4] J. Haxhibeqiri, A. Karaagac, F. V. d. Abeele, W. Joseph, I. Moerman, and J. Hoebeke, "LoRa indoor coverage and performance in an industrial environment: Case study," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1-8.

[5] J. d. C. Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic, and A. L. L. Aquino, "LoRaWAN — A low power WAN protocol for Internet of Things: A review and opportunities," in *2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*, 2017, pp. 1-6.

[6] J. Petäjäjärvi, K. Mikhaylov, M. Pettissalo, J. Janhunen, and J. Iinatti, "Performance of a low-power wide-area network based on LoRa technology: Doppler robustness, scalability, and coverage," *International Journal of Distributed Sensor Networks,* vol. 13, no. 3, p. 1550147717699412, 2017/03/01 2017.

[7] M. Saelens, J. Hoebeke, A. Shahid, and E. D. Poorter, "Impact of EU duty cycle and transmission power limitations for sub-GHz LPWAN SRDs: an overview and future challenges," *EURASIP Journal on Wireless Communications and Networking,* vol. 2019, no. 1, p. 219, 2019/09/02 2019.

[8] N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. J. L. a. Hersent, "Lorawan specification," 2015.

[9] J. P. S. Sundaram, W. Du, and Z. Zhao, "A Survey on LoRa Networking: Research Problems, Current Solutions, and Open Issues," *IEEE Communications Surveys & Tutorials,* vol. 22, no. 1, pp. 371-388, 2020.

[10] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, and I. Tinnirello, "Impact of LoRa Imperfect Orthogonality: Analysis of Link-Level Performance," *IEEE Communications Letters,* vol. 22, no. 4, pp. 796-799, 2018.

[11] D. De Guglielmo, S. Brienza, and G. Anastasi, "IEEE 802.15.4e: A survey," *Computer Communications,* vol. 88, pp. 1-24, 2016/08/15/ 2016.

[12] D. Chen, M. Nixon, and A. W. Mok, "Real-Time Mesh Network for Industrial Automation," ed: Springer: Berlin/Heidelberg, Germany, 2010.

[13] S. Petersen and S. J. I. I. E. M. Carlsen, "WirelessHART versus ISA100. 11a: The format war hits the factory floor," vol. 5, no. 4, pp. 23-34, 2011.

[14] H. Oh and C. T. Ngo, "A Slotted Sense Multiple Access Protocol for Timely and Reliable Data Transmission in Dynamic Wireless Sensor Networks," *IEEE Sensors Journal,* vol. 18, no. 5, pp. 2184-2194, 2018.

[15] G. R. Hiertz, D. Denteneer, L. Stibor, Y. Zang, X. P. Costa, and B. J. I. C. M. Walke, "The IEEE 802.11 universe," vol. 48, no. 1, pp. 62-70, 2010.

[16] A. Malik, J. Qadir, B. Ahmad, K.-L. Alvin Yau, and U. Ullah, "QoS in IEEE 802.11-based wireless networks: A contemporary review," *Journal of Network and Computer Applications,* vol. 55, pp. 24-46, 2015/09/01/ 2015.

[17] E. Ferro and F. Potorti, "Bluetooth and Wi-Fi wireless protocols: a survey and a comparison," *IEEE Wireless Communications,* vol. 12, no. 1, pp. 12-26, 2005.

[18] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks,* vol. 38, no. 4, pp. 393-422, 2002/03/15/ 2002.

[19] R. S. Sinha, Y. Wei, and S.-H. Hwang, "A survey on LPWA technology: LoRa and NB-IoT," *ICT Express,* vol. 3, no. 1, pp. 14-21, 2017/03/01/ 2017.

[20] P. Baronti, P. Pillai, V. W. C. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards," *Computer Communications,* vol. 30, no. 7, pp. 1655-1695, 2007/05/26/ 2007.

[21] G. Anastasi, M. Conti, and M. D. Francesco, "A Comprehensive Analysis of the MAC Unreliability Problem in IEEE 802.15.4 Wireless Sensor Networks," *IEEE Transactions on Industrial Informatics,* vol. 7, no. 1, pp. 52-65, 2011.

[22] R. Daidone, G. Dini, and G. Anastasi, "On evaluating the performance impact of the IEEE 802.15.4 security sub-layer," *Computer Communications,* vol. 47, pp. 65-76, 2014/07/01/ 2014.

[23] L. Leonardi, F. Battaglia, G. Patti, and L. L. Bello, "Industrial LoRa: A Novel Medium Access Strategy for LoRa in Industry 4.0 Applications," in *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, 2018, pp. 4141-4146.

[24] L. Leonardi, F. Battaglia, and L. L. Bello, "RT-LoRa: A Medium Access Strategy to Support Real-Time Flows Over LoRa-Based Networks for Industrial IoT Applications," *IEEE Internet of Things Journal,* vol. 6, no. 6, pp. 10812-10823, 2019.

[25] T. Polonelli, D. Brunelli, A. Marzocchi, and L. Benini, "Slotted ALOHA on LoRaWAN-Design, Analysis, and Deployment," *Sensors,* vol. 19, no. 4, 2019.

[26] "IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer," *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011),* pp. 1-225, 2012.

[27] L. Zheng, "Industrial wireless sensor networks and standardizations: The trend of wireless sensor networks for process autometion," in *Proceedings of SICE Annual Conference 2010*, 2010, pp. 1187-1190: IEEE.

[28] V. C. Gungor and G. P. Hancke, "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches," *IEEE Transactions on Industrial Electronics,* vol. 56, no. 10, pp. 4258-4265, 2009.

[29] B. Reynders, W. Meert, and S. Pollin, "Range and coexistence analysis of long range unlicensed communication," in *2016 23rd International Conference on Telecommunications (ICT)*, 2016, pp. 1-6.

[30] A. Augustin, J. Yi, T. Clausen, and M. W. Townsley, "A Study of LoRa: Long Range & Low Power Networks for the Internet of Things," *Sensors,* vol. 16, no. 9, 2016.

[31] J. Haxhibeqiri, F. Van den Abeele, I. Moerman, and J. Hoebeke, "LoRa Scalability: A Simulation Model Based on Interference Measurements," *Sensors,* vol. 17, no. 6, p. 1193, 2017.

[32] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa Low-Power Wide-Area Networks Scale?," presented at the Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, Malta, Malta, 2016.

[33] O. Georgiou and U. Raza, "Low Power Wide Area Network Analysis: Can LoRa Scale?," *IEEE Wireless Communications Letters,* vol. 6, no. 2, pp. 162-165, 2017.

[34] K. Mikhaylov, P. Juha, and T. Haenninen, "Analysis of Capacity and Scalability of the LoRa Low Power Wide Area Network Technology," in *European Wireless 2016; 22th European Wireless Conference*, 2016, pp. 1-6.

[35] F. V. d. Abeele, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, "Scalability Analysis of Large-Scale LoRaWAN Networks in ns-3," *IEEE Internet of Things Journal,* vol. 4, no. 6, pp. 2186-2198, 2017.

[36] T. Polonelli, D. Brunelli, A. Marzocchi, and L. Benini, "Slotted ALOHA on LoRaWAN-Design, Analysis, and Deployment," (in eng), *Sensors (Basel, Switzerland),* vol. 19, no. 4, p. 838, 2019.

[37] Q. L. Hoang, H. P. Tran, W.-S. Jung, S. H. Hoang, and H. Oh, "A Slotted Transmission with Collision Avoidance for LoRa Networks," *Procedia Computer Science,* vol. 177, pp. 94-101, 2020/01/01/ 2020.

[38] C. Pham, "Robust CSMA for long-range LoRa transmissions with image sensing devices," in *2018 Wireless Days (WD)*, 2018, pp. 116-122.

[39] T. To and A. Duda, "Simulation of LoRa in NS-3: Improving LoRa Performance with CSMA," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1-7.

[40] A. Bachir, M. Dohler, T. Watteyne, and K. K. Leung, "MAC Essentials for Wireless Sensor Networks," *IEEE Communications Surveys & Tutorials,* vol. 12, no. 2, pp. 222-248, 2010.

[41] A. S. Althobaiti and M. Abdullah, "Medium Access Control Protocols for Wireless Sensor Networks Classifications and Cross-Layering," *Procedia Computer Science,* vol. 65, pp. 4-16, 2015/01/01/ 2015.

[42] C. Ebi, F. Schaltegger, A. Rüst, and F. Blumensaat, "Synchronous LoRa Mesh Network to Monitor Processes in Underground Infrastructure," *IEEE Access,* vol. 7, pp. 57663-57677, 2019.

[43] R. Piyare, A. L. Murphy, M. Magno, and L. Benini, "On-Demand LoRa: Asynchronous TDMA for Energy Efficient and Low Latency Communication in IoT," (in eng), *Sensors (Basel, Switzerland),* vol. 18, no. 11, p. 3718, 2018.

[44] C. Liao, G. Zhu, D. Kuwabara, M. Suzuki, and H. Morikawa, "Multi-Hop LoRa Networks Enabled by Concurrent Transmission," *IEEE Access,* vol. 5, pp. 21430-21446, 2017.

[45] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with Glossy," in *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2011, pp. 73-84.

[46] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, and M. Gidlund, "Using LoRa for industrial wireless networks," in *2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS)*, 2017, pp. 1-4.

[47] D. Zorbas and B. O. Flynn, "Autonomous Collision-Free Scheduling for LoRa-Based Industrial Internet of Things," in *2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 2019, pp. 1-5.

[48] Q. L. Hoang, W. Jung, T. Yoon, D. Yoo, and H. Oh, "A Real-Time LoRa Protocol for Industrial Monitoring and Control Systems," *IEEE Access,* vol. 8, pp. 44727-44738, 2020.

[49] L. Tessaro, C. Raffaldi, M. Rossi, and D. Brunelli, "Lightweight Synchronization Algorithm with Self-Calibration for Industrial LORA Sensor Networks," in *2018 Workshop on Metrology for Industry 4.0 and IoT*, 2018, pp. 259-263.

[50] Semtech, "Application Note: SX126x CAD Performance Evaluation," November 2019 2019.

[51] N. J. D. Matloff, CA. Dept of Computer Science. University of California at Davis. Retrieved on August, "Introduction to discrete-event simulation and the simpy language," vol. 2, no. 2009, pp. 1-33, 2008.

[52] A. Brand and H. Aghvami, *Multiple access protocols for mobile communications GPRS, UMTS and beyond*. New York: John Wiley, 2002.