Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 6463097, 14 pages
https://doi.org/10.1155/2018/6463097

Research Article

TRILO: A Traffic Indication-Based Downlink Communication Protocol for LoRaWAN

Youngjune Oh[1], **Jongwon Lee**[2] **and Chong-Kwon Kim**[1]

[1]*Department of Computer Science & Engineering, Seoul National University, Seoul 08826, Republic of Korea*
[2]*Department of Computer Science & Electrical Engineering, Handong Global University, Pohang Z37554, Republic of Korea*

Correspondence should be addressed to Chong-Kwon Kim; ckim@snu.ac.kr

Received 27 April 2018; Revised 23 July 2018; Accepted 27 August 2018; Published 10 September 2018

Academic Editor: Andrea Zanella

Copyright © 2018 Youngjune Oh et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

LPWAN (Low-Power Wide Area Network) technologies such as LoRa and SigFox are emerging as a technology of choice for the Internet of Things (IoT) applications where tens of thousands of untethered devices are deployed over a wide area. In such operating environments, energy conservation is one of the most crucial concerns and network protocols adopt various power saving schemes to lengthen device lifetimes. For example, to avoid idle listening, LoRaWAN restricts downlink communications. However, the confined design philosophy impedes the deployment of IoT applications that require asynchronous downlink communications. In this paper, we design and implement an energy efficient downlink communication mechanism, named TRILO, for LoRaWAN. We aim to make TRILO be energy efficient while obeying an unavoidable trade-off that balances between latency and energy consumption. TRILO adopts a beacon mechanism that periodically alerts end-devices which have pending downlink frames. We implement the proposed protocol on top of commercially available LoRaWAN components and confirm that the protocol operates properly in real-world experiments. Experimental results show that TRILO successfully transmits downlink frames without losses while uplink traffic suffers from a slight increase in latency because uplink transmissions should halt during beacons and downlink transmissions. Computer simulation results also show that the proposed scheme is more energy efficient than the legacy LoRaWAN downlink protocol.

1. Introduction

The Internet of Things (IoT) interconnects "all" things including sensors, actuators, gateways, and everyday products such as watches and various wearable devices. The IoT ecosystem includes heterogeneous applications with diversified performance and functional requirements. To support diversified IoT applications operating on a broad spectrum of devices, multiple heterogeneous technologies should be harmoniously applied. The exigency of applying plural enabling techniques is particularly apropos in the area of networking; a single network technique cannot satisfy contradictive networking requirements of diverse IoT applications. In addition to traditional wireless network technologies such as cellular and WiFi networks, a plethora of WPAN (Wireless Personal Area Network) schemes including ZigBee, Z-Wave, IETF 6LowPAN, and Bluetooth Low Energy (BLE) can easily find areas of applications in the IoT ecosystem. Equipped with low-cost and power conserving capabilities, these network technologies compensate functional or economical cracks engendered by cellular or WiFi networks. However, most WPAN technologies are designed for short range communications and consume abundant energy to support low latency asynchronous bidirectional communications.

Recent years have witnessed explosive attentions on Low-Power Wide Area Network (LPWAN) technology as a means to fill the gap that existing network technologies leave behind. LPWAN targets to provide low-rate, high-latency communication capabilities to large numbers of IoT devices that are scattered over expanded areas such as campus, town, or city [1, 2]. Because it is difficult to manage widely scattered devices, LPWAN protocols are designed to sustain the lifetimes of devices more than ten years with a coin cell battery. To achieve both the long-range transmission and energy conservation goals, LPWAN adopts robust modulation and simple one-way communication protoscols, respectively.

Among many LPWAN alternatives, this paper deals with the LoRa [3] and LoRaWAN [4, 5] technologies. LPWAN

technologies can be classified into two categories: one operating in licensed bands and another in unlicensed bands. The 3GPP [6–9] has standardized two open standard protocols called LTE-M and NB-IoT [10, 11] that utilize the cellular infrastructure. Recently, Lauridsen et al. [12] reported extensive experimental performance results which show that NB-IoT and LTE-M schemes have contradictive characteristics and each has an advantage over the other depending on application areas. LoRa [3], SigFox [13], RPMA [14], and Weightless [15] are examples of noncellular technologies that operate in unlicensed spectrums. In addition to the free use of wireless spectrum resources, LPWAN technologies provide the advantages of network ownership and autonomous operation of networks. However, most of the LPWANs are proprietary techniques and cannot leverage the widely deployed cellular infrastructure.

Employing robust and less efficient modulation and sacrificing transmission speed, LoRa achieves long-range transmissions over 10 km. In addition, LoRaWAN achieves an unprecedented energy conservation capability by restricting downlink communications. LoRAWAN Class A end-devices stay in the sleep state as a norm and wake up only when they need to transmit uplink traffic. While LoRaWAN supports uplink transmissions like other protocols, downlink transmissions are allowed only as a response to uplink packets. As a result, LoRaWAN eliminates idle listening, a root cause of energy consumption in communications. These characteristics and limitations of LoRa and LoRaWAN indicate that they are optimized for applications that do not require high throughput or low latency data delivery.

In this paper, we embark on the design of an energy conservative downlink communication protocol on top of the LoRaWAN protocol. Let us start with a brief introduction of the LoRaWAN protocol. LoRaWAN forms a star-of-stars topology consisting of end-devices, gateways, and servers. A server is connected to multiple gateways each of which again is connected to multiple end-devices. A gateway assumes a role similar to a WiFi Access Point (AP). A gateway is connected to multiple end-devices via the LoRa wireless technology. Note that a LoRaWAN does not support multihop wireless transmissions; all end-devices are directly connected to gateways. LoRa is particularly designed for massive IoTs over a wide area, and a single LoRaWAN network can support tens of thousands of end-devices each of which emits short packets infrequently.

LoRaWAN defines three classes of end-devices depending on the level of energy conservation and the demands for downlink transmissions. A Class A node abnegates the reception of asynchronous downlinks by staying mostly in a sleep mode turning off its radio. It only wakes up for uplink transmissions as such demands occur. Class C end-devices operate in an always-on mode which allows asynchronous downlinks as well as uplinks. Class C end-devices assume the continuous supply of power, or their lifetimes would be much shorter than those of Class A devices. Class B end-devices also support downlinks in addition to uplink transmissions. Unlike uplink transmission that can be initiated at any time, downlinks should be synchronously scheduled according to the requests from end-devices. They use beacons to maintain time synchronization between a server and end-devices. Class B nodes wake up at scheduled intervals periodically even if there are no downlinks for them. We aim to devise a new downlink protocol which aims to be more energy efficient and flexible than the LoRaWAN's Class B downlink scheme.

The proposed downlink protocol, named TRILO, is based on the beacon mechanism. Like the IEEE 802.11 WLANs where an AP coordinates communications of multiple mobile stations, a gateway in a LoRaWAN can broadcast frames to all end-devices. Beacons are an efficient signalling mechanism because a broadcast beacon can alert multiple end-devices with a single message. In addition to beacons, a polling mechanism to announce the readiness to receive downlinks must be provided. A polling scheme should be designed carefully such that it fully exploits the physical characteristics of the underlying LoRa protocol. We consider two polling alternatives: concurrent polling and sequential polling. We choose sequential polling because a performance comparison indicates that sequential polling is more effective in power saving than concurrent polling (see Figure 6).

We implemented TRILO on commercially available LoRaWAN components. For a gateway, we used a RAK831 board which contains a SX1301 concentrator chip. The concentrator board is attached to a Raspberry Pi 3 board. An end-device is implemented on a NUCLEO-L073RZ board with an embedded LoRaWAN chip using 915Mhz antenna. The SX1272MB2DAS component is attached to a NUCLEO-L073RZ board with a modified LoRaMAC that includes our proposed scheme. Note that a network server and end-devices are communicating entities in the LoRaWAN architecture; gateways assume a passive role relaying frames between end-devices and a server.

We confirmed that the devised downlink protocol works correctly via real-world experiments. Extensive experiments were also conducted to collect important performance measures such as PDR (Packet Delivery Ratio), latency, and power consumption. An experimental network, which consists of one gateway and 15 end-devices, is deployed to our office building. Experimental results show that TRILO operates correctly. All end-devices receive beacons and downlink frames successfully. The PDR of downlinks is almost 100% in most experiment setups. Under an assumption that downlink traffic to each end-device is not heavy, the extra wake-up times required for maintaining and supporting downlink frame transmissions are minimal.

We also compare the performance of TRILO with that of Class B downlink protocol via computer simulations. Power consumption is an important performance metric in LoRaWAN. We measure the efficiency of downlink protocols which is the ratio of the actual data transmission time to the duty cycle. Simulation results show that the proposed protocol performs significantly better than the Class B scheme. The proposed scheme shows 3.7 times to about 14.6 times better efficiency than the Class B downlink protocol.

The contributions of this paper are as follows.

(i) We propose a new energy efficient downlink protocol, called TRILO, for LoRaWAN. TRILO is customized to the

physical and network properties of LoRa and LoRaWAN. In particular, we aim to devise a protocol that is more energy efficient than synchronous downlink mechanism developed for LoRaWAN's Class B.

(ii) We devise two polling mechanisms called concurrent polling and sequential polling. Concurrent polling best exploits the LoRa's capability that supports simultaneous receptions of multiple packets. However, the analysis indicates that sequential polling is more energy efficient than concurrent polling, and we adopt sequential polling.

(iii) TRILO is implemented on commercially available LoRaWAN components. Modified LoRaWAN components operate properly in real-world experiments.

(iv) We conduct experiments with modified LoRaWAN devices. Our performance study shows that PDR is almost 100% for both uplink and downlink traffic and the extra wake-up times to maintain and support beacons and downlink packets are minimal.

(v) We conduct computer simulations to compare the performances of TRILO and the LoRaWAN's Class B downlink protocol. Our performance study shows that the proposed scheme, which minimizes unnecessary receive window allocations, is significantly more energy efficient than the legacy LoRaWAN.

The rest of the paper is organized as follows. We begin with the related work highlighting several low-power MAC mechanisms. We then give a brief primer on LoRa and LoRaWAN focusing on the features of LoRa and LoRaWAN that are closely related to the design of downlink protocols. Next, we describe the proposed beacon based downlink protocol. Two polling methods, concurrent polling and sequential polling, are explained. We compare the wake-up times of the two polling methods and show that sequential polling is more energy efficient than concurrent polling. Then, the implementation details of TRILO and performance results obtained from real-world experiments are described. We also illustrate the performance comparison of TRILO and the LoRaWAN's downlink protocol. Lastly, conclusion summarizes the paper and discusses several future research problems.

2. Related Work

Energy is one of the precious resources that should be managed carefully in battery powered mobile devices. Many researchers have studied energy conservation schemes in various scenarios. This section reviews several exemplary power saving schemes.

Basically, almost all energy conservation schemes try to minimize the duty cycle. The literature can be partitioned into two categories depending on the type of communications: one to one communications and one to many communications. Usually, one to one communication is between two devices of the equal capabilities. On the other hand, in most one to many networks, a central node enjoys unlimited power supply from a powerline while many others are battery powered mobile devices. For example, a WiFi network consists of one AP (Access Point) and many mobile stations. As most prior works, we focus on the energy conservation for battery powered devices.

Many clever low-power MAC mechanisms have been designed for WSNs (Wireless Sensor Networks) which implement multihop wireless networks over WiFi networks or the IEEE 802.15.4 standard protocols. Even though base network technologies support one to many communications, neighbour nodes in mesh networks assume peer to peer communications. The energy conservation mechanisms for WSNs can be classified by several criteria such as synchronicity and sender- or receiver-initiated. The two most widely deployed protocols, ContikiMAC [16, 17] and TinyOS LPL [17, 18], are asynchronous sender-initiated protocols and are implemented on two *de facto* standard operating systems, Contiki [16] and TinyOS [18], respectively. In sender-initiated protocols, a sender transmits a packet repeatedly for a time longer than a predetermined duty cycle interval. A receiver operates in the duty cycling mode; it wakes up at every duty cycle interval and listens for a possible transmission from the sender.

Bluetooth Low Energy (BLE) [19] is another network technology that requires one to one energy conserving MAC. BLE aims for the lifetime from several months to years with a coin cell battery [19]. The BLE neighbour (device) discovery mechanism is an asymmetric scheme where each device plays the role of either an advertiser or a scanner. To bind with another BLE device, an advertiser transmits advertising packets periodically at every scheduled advertising event. A scanner also turns its radio on repeatedly at every scanning events. Applying the Chinese Remainder Theorem, Kandhalu, Xhafa, and Hosur [20] devised a BLE neighbour discovery mechanism that guarantees eventual rendezvous of advertisers and scanners. To avoid the ill-fated case that both advertiser and scanner choose the same prime number, each device selects two prime numbers and wakes up at every integer multiples of the primes.

Several researchers have proposed new neighbour discovery schemes that reduce discovery latency, power consumption, and reliability of the BLE device discovery mechanism. Unlike the legacy BLE devices whose roles are fixed to either advertiser or scanner at the production time, [21] assumed that devices could change their roles dynamically. They devised a new neighbour discovery protocol customized for such environments. Contrary to the prior methods that use time slots, BLEnd [22] demises the slot structure and reduces the discovery delay. Nihao [23] improves the latency and reduces energy consumption by increasing advertisements and decreasing scanning times.

One notable energy conservation protocol for one to many communications is the IEEE 802.11 PSM (Power Saving Mode). An Access Point (AP) periodically broadcasts beacons to notify mobile stations that have pending downstream frames. Inspecting a beacon frame, idle stations without buffered downlinks turn their radio off immediately until the next beacon interval. Backlogged stations request for downstream packet transmissions by sending poll messages to the AP. Upon the delivery of downlink frames, backlogged stations also fall back to the sleep state until the next beacon.

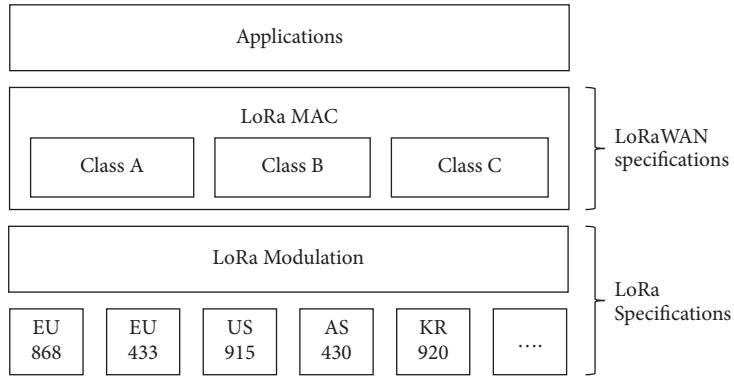


Figure 1: LoRa and LoRaWAN specifications.

Table 1: Frequency bands and bandwidths allocated for LoRa.

Location	Europe	North America	Korea	China	Japan
Frequency band(MHz)	867-869	902-928	920-923	470-510/779-787	920-925
Channels	10	64+8+8	13	-	-
Uplink CH BW(KHz)	125/250	125/500	125	-	-
Downlink CH BW(KHz)	125	125	125	-	-
Tx power Up (dBm)	+14 dBm	+20~+30 dBm	+10~+14 dBm	-	-
Tx power Down (dBm)	+14 dBm	+27 dBm	+23 dBm	-	-

3. A Primer on LoRa and LoRaWAN

This section contains a brief overview of LoRa and LoRaWAN technologies. Specifically, technical properties that influence or limit the design space of LoRaWAN based downlink protocol will be addressed.

Figure 1 shows the protocol stack of LoRa and LoRaWAN [4]. LoRa, developed and owned by Semtech [3], is a propriety technique that defines a physical layer customized for LPWA communications. LoRaWAN, which defines a MAC layer on top of LoRa, is an open standard developed by a nonprofit organization called LoRa alliance. LoRa operates on the 900 MHz ISM spectrum and adopts the CSS (Chirp Spread Spectrum) modulation technique. CSS [24, 25], based on chirped-FM modulation, yields high processing gains and enables long transmissions at the expense of data rates. LoRa supports Adaptive Data Rate (ADR) which varies data rates adapting to transmission distances and channel conditions. When the SNRs are low, LoRa lowers the data rate to several hundred bits per second by using large SF (Spreading Factor) and robust CR (Coding Rate). When SF and bandwidth (BW) are selected, the data rate, R_{SF}, is determined as $SF/2^{SF} \bullet BW$. The lowest data rate of LoRa, determined at SF=12 and BW=125 KHz, is approximately 300 bits per second. The highest data rate is approximately 11 Kbps and is obtained at SF=7 and BW=250 KHz. High data rates can be achievable when channel conditions are good; SF=7 requires -120 dBm while SF=12 requires only -136 dBm.

Each country allocates different frequency bands and bandwidth for LoRa. Table 1 illustrates the frequency bands, bandwidth, and transmission powers defined by the regulatory bodies at several regions and countries. It is worthwhile to note that a device that locks on one SF can successfully receive a signal even though signals of different SF exist at the same time on the same subchannel. Moreover, gateways can simultaneously receive multiple transmissions if either subchannels or SFs are different. Currently, the SX1301 based gateway supports concurrent demodulations up to 8 signals.

LoRa's CSS modulation enables packet receptions even when the signal power is much less than the absolute noise floor. The high reception sensitivity makes carrier sensing practically impossible. Lack of carrier sensing forces the original LoRa to use the Aloha MAC that suffers from excessive collisions. Even though some countries require LBT (Listen Before Talk), it is quite difficult to sense on-going LoRa signals without synchronizing with preambles. This paper assumes that all devices use the Aloha MAC.

The LoRaWAN end-devices are classified into three classes, Classes A, B, and C. A Class A device—bidirectional end-devices—can initiate uplink transmissions only. It can receive downlink frames, but only as a response to uplink frames that it has initiated. Figure 2 describes the timing of an initial uplink transmission and following downlink transmissions. The timing of downlink transmissions should be carefully coordinated; the server should schedule a downlink such that its transmission interval is coincident

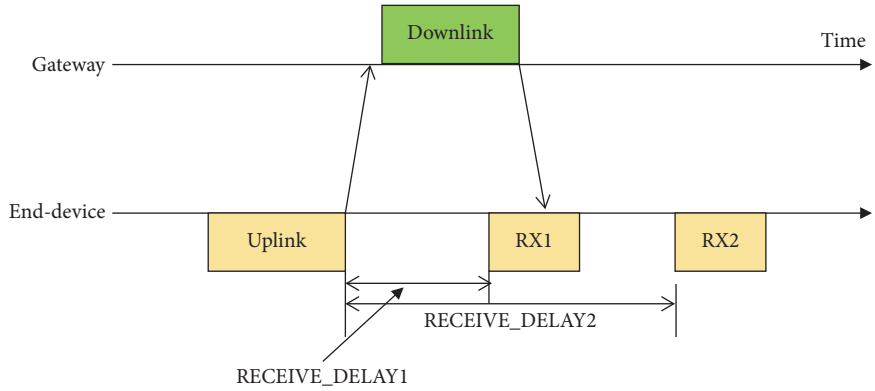


Figure 2: Class A end-device initiated uplink and responding downlinks.

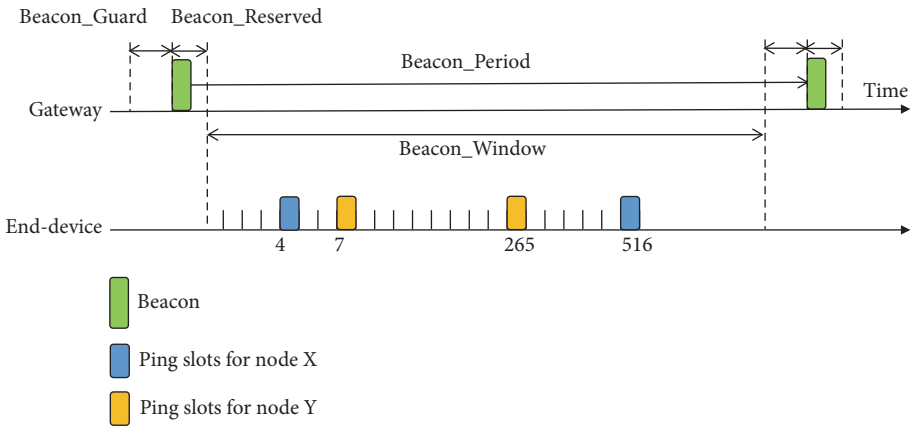


Figure 3: Beacon based Class B downlink scheme.

with predetermined receive windows. Class A end-nodes sacrifice asynchronous downlink transmissions but attain long lifetimes as a reward. It is expected that battery powered Class A end-devices enjoy ten to twenty years of lifetimes in reasonable use cases. Class A is suitable for sensor nodes that report sensing data periodically or when the readings satisfy predetermined criteria.

Unlike a Class A end-device, Class B and C devices support downlink communications. A Class C end-device operates in an always-on mode and wastes significant amounts of energy for idle listening required for the receptions of asynchronous downlinks. Class B end-devices—bidirectional end-devices with scheduled receive slots—support *synchronous* server-initiated downlinks. Class B end-devices must be time synchronized with the gateway which transmits beacons at every beacon period of 128 seconds. Once being synchronized, Class B end-devices wake up at each beacon period to maintain the synchronization. LoRa discretizes the time into so-called *ping slots* each of which is 30 mSec long. One beacon period contains 4096 ping slots, and a device can request 2^k ($0 \leq k \leq 7$) ping slots per beacon period. The first ping slot position is determined randomly, and p-th ping slot starts at $t_{BEACON\_RESERVED} + (r_0 + (p-1) \bullet 2^{12-k}) \bullet 30$ where $t_{BEACON\_RESERVED}$ is the time when the BEACON_RESERVED finishes; r_0 is the first offset randomly selected. Figure 3 shows the beacon timing and an example of Class B end-devices' ping slot schedules (nodes X and Y) whose parameters are $k = 3$ and $r_0 = 4$, and $k = 4$ and $r_0 = 7$, respectively.

The downlink protocol for Class B end-devices incurs significant overheads. In addition to beacons, they must blindly wake up at their reserved ping slots regularly even though there are no buffered frames for them. Also, devices consume energy to maintain a precise clock to catch the interrupts during ping slots. We design and implement a new downlink scheme to reduce such overheads.

4. Downlink Communications Scheme

There can be many alternative methods for downlink transmissions. One simple method is to utilize the Class A uplink-downlink transaction; an end-device periodically transmits probe messages to the server, and if there is a downlink message for the device, the server sends it as a response to the uplink probe. This method is effective and energy efficient if end-devices can guess the existence of downlink messages correctly. However, exact prediction of downlinks is practically impossible. Because end-devices cannot predict when downlink packets are ready, they can issue either too many unnecessary probes when the probing interval is short or suffer from long latencies when the probing interval is long.

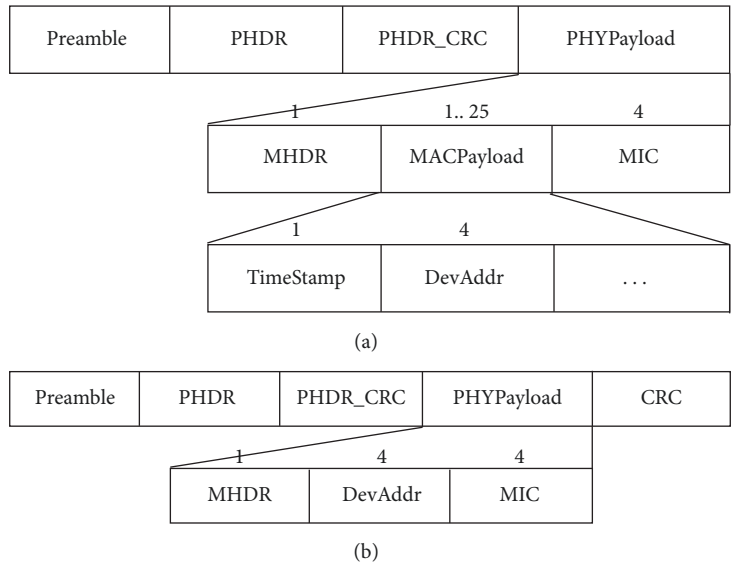


Figure 4: Beacon and poll message formats. (a) Beacon. (b) Poll.

Broadcasting is an effective mechanism to disseminate control information in a network that supports one to many communications. Indeed, IEEE 802.11 WLANs [26]—where an AP and multiple stations form a one to many network—adopt the periodic broadcast scheme to alert backlogged mobile stations in the Power Saving Mode (PSM). A WLAN AP periodically (usually every 100 mSec) broadcasts a beacon message that contains Traffic Indication Map (TIM). A TIM enumerates mobile stations that have buffered downlink packets. Stations in the sleep mode wake up periodically at every beacon interval to listen TIMs. Backlogged stations then send PS-Poll frames requesting downlink packet transmissions. As soon as a station receives a buffered frame, it falls back to the sleep mode until the next beacon. Stations without buffered downlink frames immediately return to the sleep state to minimize energy consumption.

Because a LoRaWAN forms a star (more precisely, a star-of-stars topology) and supports broadcasting like IEEE 802.11 WLAN, it is natural to adopt a downlink mechanism similar to the IEEE 802.11 PSM. In addition, as explained in the previous section on LoRaWAN primer, the LoRaWAN standard already supports a beacon mechanism for time synchronization. We can easily modify the structure of a LoRaWAN's beacon frame to augment a field that specifies backlogged end-devices. This additional field is similar to the TIM field of the WLAN. The message formats of beacon and poll frames are shown in Figure 4. Compared to beacon transmission, polling mechanisms for LoRaWAN ask for a careful design because the MAC protocols in LoRaWAN are practically limited to the unslotted Aloha scheme. Given that carrier sensing is prohibited, a beacon triggers synchronized medium access from backlogged end-devices and causes excessive collisions. On the other hand, a LoRaWAN is robust against collisions because it supports multiple simultaneous receptions of different SFs and different subbands. In short, because polling can affect the performance of overall systems significantly, it is worthwhile to carefully analyse all aspects that may affect the performance.

We consider two polling mechanisms: concurrent polling and sequential polling. Before delving into the details of polling schemes, let us first introduce the notations used for the following analysis (Table 2).

Concurrent polling is designed to best exploit the characteristics of the LoRa's PHY layer. Note that a gateway can receive multiple signals simultaneously if their channels or SFs are different. Commercially available gateways support concurrent demodulation up to 8 signals; however, let us assume that the number of simultaneous receptions is increased multiplicatively by deploying multiple gateways. For example, if a LoRa network uses 10 channels and 6 different SFs as in Europe, then the maximum of 60 polls can be delivered simultaneously when more than or equal to 8 gateways are deployed. Let *busy* and *idle* devices be end-devices with and without pending downlink frames, respectively. Hearing a beacon, all busy end-devices transmit polls to the server simultaneously as shown in Figure 5(a). Polls that are in the same subchannel and of the same SF will experience collisions and will be lost. For example, polls from devices $E_1, E_2, and\ E_3$ are transmitted successfully. However, polls from devices $E_4, and\ E_5$ will collide and will be lost.

Let n and m be the number of the maximum simultaneous frames that the network allows and the number of frames that arrive during a beacon period (i.e., the number of polls assuming that there is at most one frame per end-device). The expected number of successful polls is $m \cdot e^{-m/n}$, approximately. Because collided polls will compete again at the next beacon interval, the actual number of polls competing at each beacon is approximated as $m' = m + m(1 - e^{-m/n})$.

Idle devices fall back to the sleep state as soon as they recognize there are no frames for them. The maximum

Table 2: Notations.

Notation	Meaning
D_b	Size of a beacon except preamble, the minimum size is 17+4• m bytes where m is the number of end-devices with buffered downlinks
D_p	Size of a poll frame except preamble. The minimum size can be 4
D_d	Average size of a downlink frame except preamble
t_c	Time to compensate clock drift. According to the LoRaWAN standard, it is set to 13 mSec.
t_g	Time gap between two consecutive frames during downlink operation. Frames include poll and data.
r_b	Beacon broadcasting data rate. The standard specifies using SF=12 and the data rate is 250 bps
r_p	Average poll frame transmission data rate
r_d	Average downlink frame transmission data rate

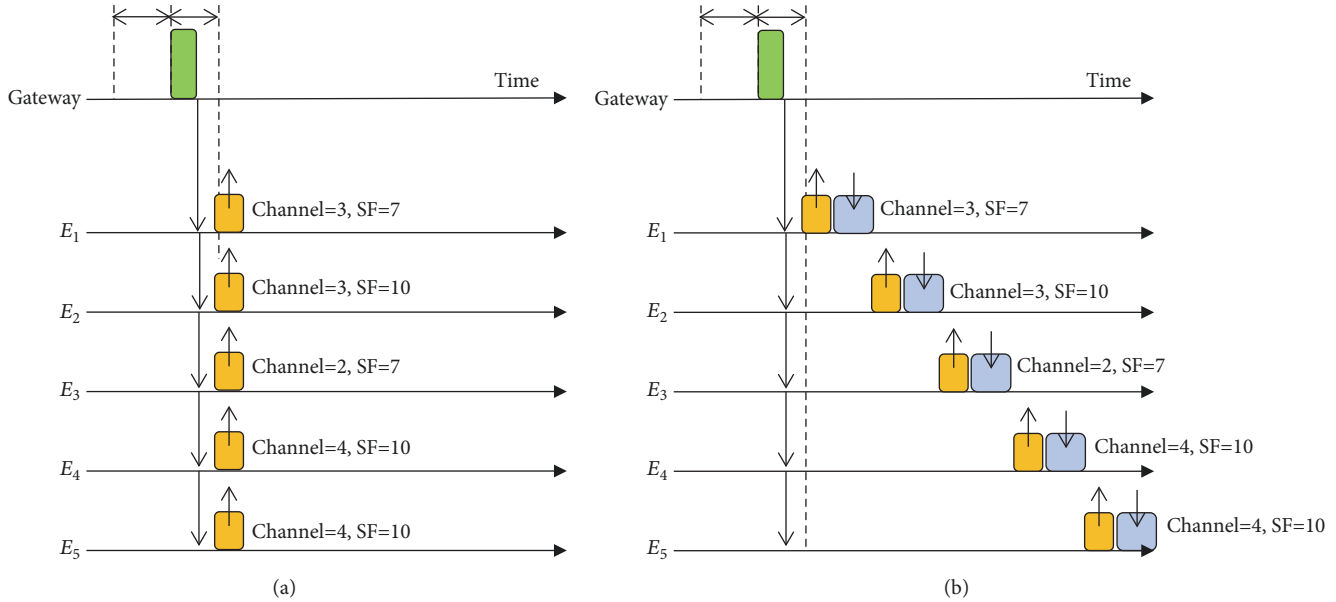


Figure 5: Example of polling mechanisms. (a) Concurrent polling. (b) Sequential polling.

wake-up time of an idle device during a beacon period, W_I^C, is computed as follows.

$$W_I^C = t_c + \frac{D_b}{r_b} \tag{1}$$

A busy end-device transmits a poll and waits until it receives its downlink. As soon as it receives the downlink, it turns its radio off until the next beacon. Because the gateway transmits downlinks only to the end-devices whose polls are successfully received, on the average, $m' \bullet e^{-m'/n}$ downlink frames will be transmitted. The average maximum wake-up time of a busy device whose poll is delivered successfully to the gateway is

$$W_{BS}^C = t_c + \frac{D_b}{r_b} + \frac{D_p}{r_p} + \frac{1}{2} \bullet \left(t_g + \frac{D_d}{r_d}\right) \bullet m' \bullet e^{-m'/n} \tag{2}$$

A busy device whose poll fails to reach to the gateway must wait until all downlink transmissions are completed, and its wake-up time is

$$W_{BF}^C = t_c + \frac{D_b}{r_b} + \frac{D_p}{r_p} + \left(t_g + \frac{D_d}{r_d}\right) \bullet m' \bullet e^{-m'/n} \tag{3}$$

Under a simple assumption that all end-devices are equally probable to have a pending downlink frame, the sum of wake-up times of all M devices under the synchronous polling scheme is

$$\left(M - m'\right) \bullet W_I^C + m' \bullet e^{-m'/n} \bullet W_{BS}^C + m' \bullet \left(1 - e^{-m'/n}\right) \bullet W_{BF}^C \tag{4}$$

The average time that a node should turn on its battery to receive a downlink is derived by dividing (4) by the multiplication of M and the average number of successful downlinks per beacon period.

Another scheme that we have considered is sequential polling. Unlike the contention based concurrent polling, end-devices poll and receive downlink in the same order that their addresses appeared in the DevAddr fields of a beacon frame. Figure 5(b) shows an example of sequential polling. Receiving a beacon, busy devices remember their positions in the DevAddr field and all devices except the first busy device fall back to the sleep state. The first busy device transmits a poll and receives a downlink frame. The second busy device wakes

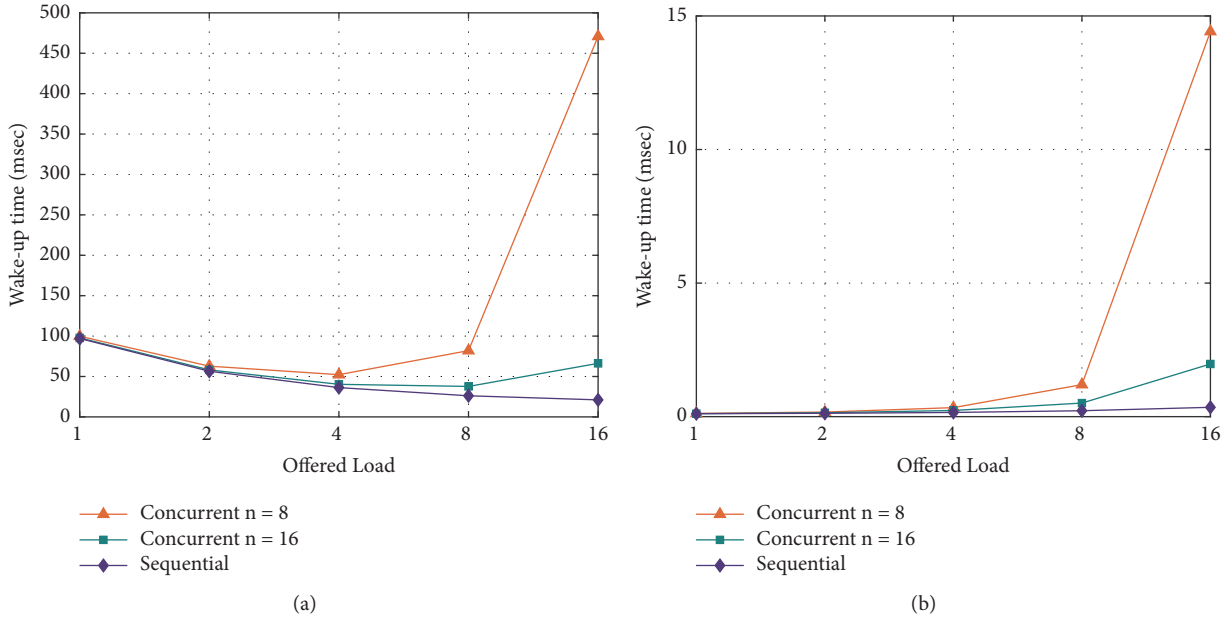


FIGURE 6: Performance of concurrent polling and sequential polling. (a) Average wake-up time of both idle and busy devices. (b) Average wake-up time of busy devices.

up after one poll-data frame exchange time to poll and receive its downlink. The same procedure repeats for succeeding busy devices. Equation (5) is the average maximum wake-up time of a busy device.

$$W_B^S = t_c + \frac{D_b}{r_b} + \frac{D_p}{r_p} + \left(t_g + \frac{D_d}{r_d}\right) \tag{5}$$

Moreover, the sum of all wake-up times during a beacon period is

$$(\mathrm{M} - \mathrm{m}) \bullet W_I^S + m \bullet W_B^S \tag{6}$$

5. Comparison of Two Polling Schemes

We compared the two polling methods by varying the values of parameters that may affect performance. The performance metric is the average wake-up time required for the reception of one downlink frame. We varied the total number of end-devices (M) in a network, the number of downlink messages per beacon interval, and the network capacity for concurrent transmissions (n). Figure 6 shows the average wake-up times of concurrent and sequential polling as functions of the offered load. The unit of the offered load is the number of new downlink frames generated per beacon interval. Figures 6(a) and 6(b) are the wake-up time of all end-devices including both idle and busy devices and the wake-up time of busy devices only, respectively. When calculating the wake up time of busy devices only, terms $(M-m')\bullet W_I^C$ and $(\mathrm{M}-\mathrm{m})\bullet W_I^S$ in (4) and (6) are excluded. We used two values (n=8 and n=16) of network capacity to analyse the effect of maximum simultaneous receptions. For simplicity, we assume that reception paths of gateways are ideally utilized as each gateway does not lock on the same packet. In both Figures 6(a) and 6(b), we can observe that sequential polling is more effective in power conserving than concurrent polling. When the offered load is small, the performance differences are not great. However, as the offered load increases, the gap increases also and at m=16, concurrent polling consumes 3 to 22 times more energy than sequential polling in the case of all devices (Figure 6(a)). Moreover, in the case of busy devices only (Figure 6(b)), concurrent polling consumes 5.5 to 41 times more energy than sequential polling. The number of total end-devices is fixed at 1000. We performed the same analysis by varying the value of M and obtain the similar results.

We also can observe that concurrent polling at n=16 is better than that at n=8. As n increases, the probability of collision decreases. Collisions affect the performance in two ways. A device, whose poll fails to reach a gateway, wastes energy because it should wait until all downlinks are completed. Also, collisions increase the offered load at the following beacon period, and an inflated load again increases the collision probability. Generally, the wake-up time of all devices decreases as the offered load increases up to 4 (Figure 6(a)). However, as the offered load exceeds network capacity (n), wake-up time increases sharply.

Because our analytic study reveals the performance advantage of sequential polling over concurrent polling, we adopt sequential polling for our downlink communication scheme. Figure 7 illustrates TRILO in a greater detail. *Beacon_Window* is divided into downlink reception time and uplink transmission time. After a beacon, busy devices receive downlink frames sequentially during downlink time. After that, nodes transmit uplink messages like plain Class A devices. One problem is that uplink transmissions or the downlinks responding to uplinks may overlap with the following beacon. To avoid the problem, the *Beacon_Guard* is introduced; end-devices refrain from initiating new uplink

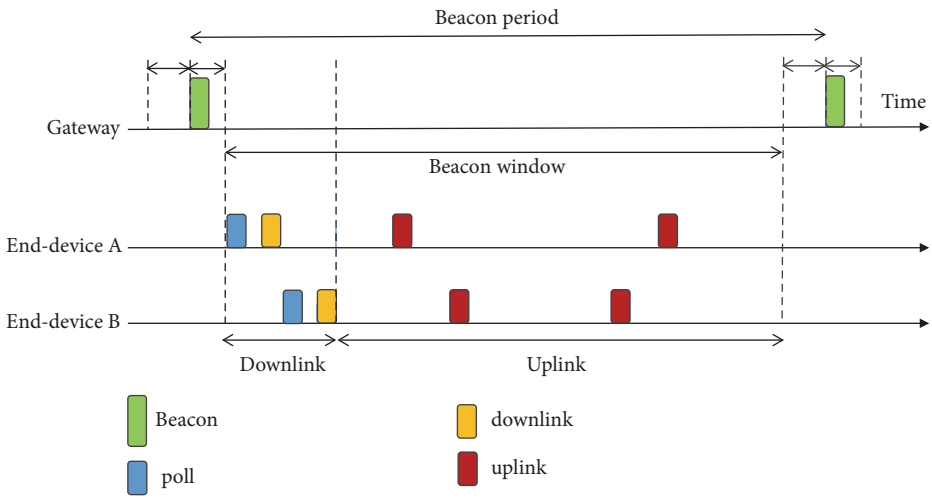


Figure 7: Proposed downlink communication scheme.

(a) (b)

Figure 8: Devices used for implementation. (a) End-device with SX1272MB2DAS attached to a NUCLEO-L073RZ board. (b) Gateway with RAK831 attached to a Raspberry Pi 3.

transmissions if remaining time to the next beacon is less than *Beacon_Guard*. Note that LoRaWAN standard limits maximum payload size, so that beacon payload cannot convey a large number of DevAddrs; however, it can be easily extended depending on network conditions by using low SFs or allocating short network addresses.

6. Implementation

We implemented TRILO using commercially available components. We built a gateway on a RAK831 board [27] which contains an SX1301 concentrator chip. The gateway board is attached to a Raspberry Pi 3 board as shown in Figure 8(b). For our implementation, we employed the Semtech libraries which contain LoRaWAN packet forwarder and hardware abstraction layer for SX1301.

In the LoRaWAN architecture, a gateway plays a simple role of a bridge between end-devices and a network server, and a network server performs all beacon and poll message processings. We utilize an open source project [28], targeted for private LoRaWAN network production in implementing a network server. Figure 9 describes the message handler modules that we implemented. The beacon handler creates a beacon payload consisting of the timestamp and buffered downlink frame information. Then it passes the payload to the packet processor which assembles the payload into a beacon packet. Since a beacon needs to be transmitted in a greater time precision than data packets, we modified the gateway such that beacons are transmitted immediately. The poll handler sends a pending downlink frame in response to a poll message. Note that this mechanism is transparent to the LoRaWAN MAC processor and the application server because we implement modules at the packet processor level.

An end-device is implemented on a NUCLEO-L073RZ board with an embedded LoRaWAN chip using a 915Mhz antenna (see Figure 8(a)). We attached the SX1272MB2DAS component to a NUCLEO-L073RZ board with the modified LoRa MAC. Note that end-device logics are compiled and stored as firmware in an end-device. Time synchronization between nodes and a network server is based on the local time of a network server instead of GPS time because we implemented the beacon handler at the server, not using the beacon frame of the Class B standard. We modified the end-device's software by adding the beacon handling and polling functions. Note that each device which has a pending frame waits for its turn, and processing times of each downlink vary

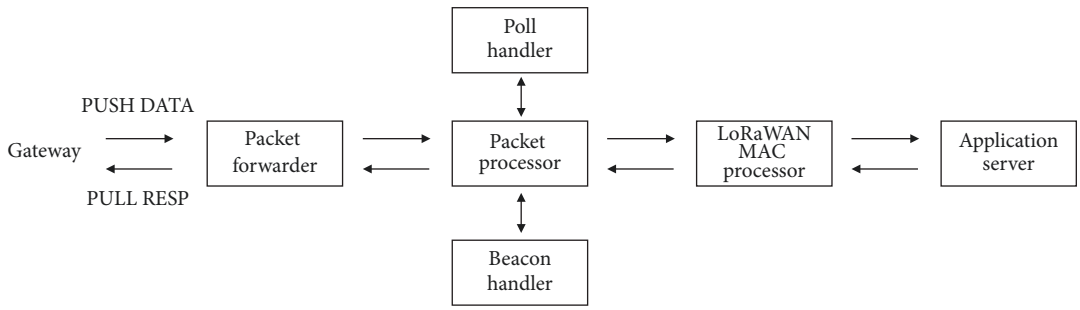


Figure 9: Block diagram of a network server that handles beacon and poll messages.

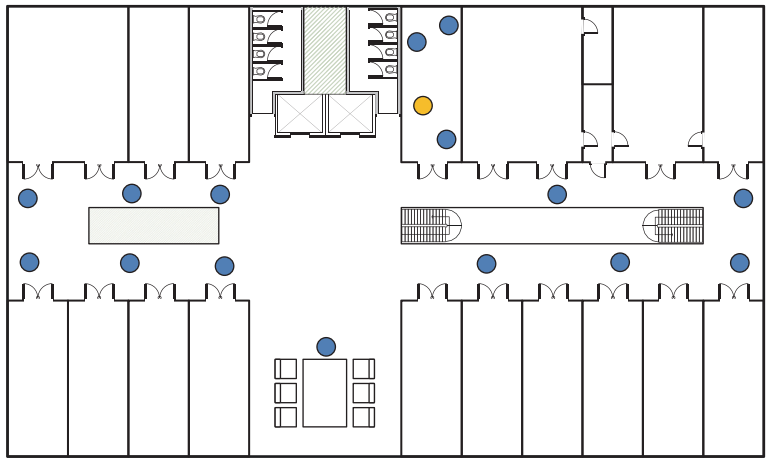


Figure 10: Experiment environment. One gateway (yellow dot) and 15 end-devices (blue dots) are deployed in our office building.

due to different SFs. In a downlink period, we configured the time for each poll and downlink based on SF 12 which has the longest airtime. However, the experimental results show that the proposed scheme can be deployed with high performance even with the conservative configuration. It is possible to optimize the waiting times for each downlink more compactly based on the information about SFs of each node; however, we leave this issue as future work.

We tuned radios of the components to obey the Korean regulation which allocates frequency bands from 920Mhz to 923Mhz for LoRaWAN. We also added the listen before talk (LBT) function as the regulation specifies. Note that the current LoRaWAN specifications for the KR920-923 ISM band do not have duty cycle limits but require adopting the LBT channel access rule [4].

7. Experimental Results

We evaluate the performance of TRILO by deploying modified components in our office building. We placed the gateway and three end-devices in our lab. Also, we deploy 12 more end-devices on the same floor of the building as shown in Figure 10. The beacon periods are set to either 128 sec. or 256 sec. We varied the uplink and downlink offered loads from 1, 2, 4 to 8 frames per beacon period. The offered traffic is unrealistically large for such a small network; however, note that we are trying to emulate the operating conditions of large scale LoRAWAN networks consisting of tens of thousands of end-devices. Testbeds of such scale are not available, so our experiment emulates large networks with 15 end-devices. Uplink and downlink messages should be interpreted as an aggregated traffic generated by thousands of end-devices. Message interarrival times of uplink and downlink traffic follow the exponential distribution, and the destinations of downlink messages are randomly and uniformly selected. We used primitive Adaptive Data Rate (ADR) for SF and power allocation, and all nodes converged to SF 8 and default EIRP output power (14dBm). The performance of real-world LoRaWAN networks depends on the distributions of SF and transmission power. However, we ignore these factors because improvement of uplink transmissions is not a goal of this work.

The performance metrics used in this study include Packet Delivery Ratio (PDR), duty cycle, and latency. To obtain reliable results, we repeat experiments with the same parameter values ten times. The length of each experiment run is 10 beacon periods (i.e., 1,280 or 2,560 sec). Because the destinations of downlinks and the origins of uplinks are randomly selected, it is expected that all end-devices and messages experience the average performance.

First of all, we measure the PDR of uplink and downlink frames. We observe that all downlink frames are delivered successfully without exception. The result is rather surprising because TRILO employs beacon broadcasting and polling

Table 3: Packet delivery ratio of uplink traffic at various uplink and downlink offered loads.

Packet generation rate	$\lambda_d = 1$	$\lambda_d = 2$	$\lambda_d = 4$
$\lambda_u = 2$	98.64%	98.27%	97.72%
$\lambda_u = 8$	96.74%	96.03%	95.10%

in addition to downlink data frames. A failure in any one component may lead to unsuccessful data transmissions. Unlike downlink frames, uplink traffic experiences occasional transmission failures. We measure the PDR by varying the offered loads of both uplink and downlink traffic. Note that all nodes use the same SF, and collisions occur if frames are transmitted on the same channel simultaneously. Table 3 illustrates the PDR of uplink traffic at different offered loads. Both uplink and downlink traffic affect the PDR of uplink traffic. The PDR decreases as the offered load of downlink traffic increases; at $\lambda_u = 2$, the PDR decreases from 98.64% to 97.72% as the offered load of downlink traffic increases from 1 to 4. In addition, the uplink traffic intensity also affects the PDR of uplink traffic; for example, at $\lambda_d = 2$, the PDR decreases from 98.27% to 96.03% as the offered load of uplink traffic increases from 2 to 8.

Note that uplink transmissions are prohibited during polling and following downlink frame transmissions. Uplink frames which arrive during downlink operation should wait until the current downlink operation is completed. This waiting procedure synchronizes backlogged end-devices and may cause collisions. Even though the LoRa PHY supports multiple simultaneous transmissions of different SFs or channels, multiple attempts increase the collision probabilities and the PDR decreases as both uplink and downlink traffic increase. To reduce the bloated collisions due to synchronous transmissions, persistent mechanisms such as 0-persistence or p-persistence are required. However, persistence schemes increase the complexity of the network protocol and may cause other side effects such as increased energy consumption.

Next, we assess the additional power consumed for downlink communications. Because gateways operate in an always-on mode, we only measure the energy consumed by end-devices. We use the duty cycle, additional time required for downlink communications, as the performance metric. The duty cycle is the sum of wake-up times used for beacon reception, for possible poll transmission, and for downlink data frame receptions. Figure 11 shows the average duty cycle at different downlink traffic loads. The duty cycle increases as the downlink offered load increases; it increases from 0.75% to 1.1% as λ_d increases from 1 to 4. Note that all end-devices consume the minimal fixed duty cycle for beacon reception regardless of the intensity of downlink traffic. In addition to the fixed duty cycle, end-devices with backlogged downlinks continue to stay in the wake-up state for polling and data reception.

Figure 12 illustrates the average latency of downlink traffic and uplink traffic as functions of offered load, respectively. As shown in Figure 12(a), the average latency of downlink traffic increases in proportion to the offered load. As explained

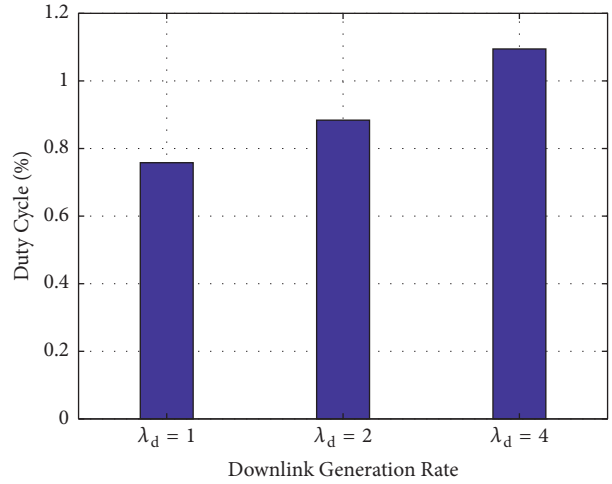


Figure 11: Duty cycles required for downlink operation including beacon, poll, and downlink data transmissions.

before, downlink frames are generated according to a Poisson process. Therefore, the probability that a downlink frame occurs in a small time period follows uniform distribution and, on average, each downlink frame waits one-half of the beacon interval. In addition, frames that arrive during a downlink processing are postponed to the next beacon. In Figure 12(a), we can observe that the effect of uplink traffic on the average latency of downlink is minimal. This result is not surprising because uplink and downlink communications are well isolated in the proposed scheme.

Like the average downlink latency, the average uplink latency increases as the downlink offered load increases. Uplink frames that arrive during downlink periods should wait until the end of the current downlink period. The average latency of uplink traffic as well as the duty cycle increase as the downlink offered load increases. Contrary to Figure 12(a) where the average downlink latency is seldom affected by the uplink offered load, the average uplink latency increases proportionally as the uplink offered load increases. As the uplink communication demands increase, a queue starts to form, and this causes the increase in waiting time. The queueing time is most distinctive at $\lambda_d = 4$ when the service time for uplink communications is minimal.

8. Simulation Results

We also compared the performance of the Class B downlink scheme and that of TRILO. Because commercial LoRaWAN devices do not support Class B functions yet, we relied on computer simulations for the performance comparison. The simulation code is written primarily in C/C++, and we conduct extensive simulations in a typical LoRaWAN network and in a heavy downlink traffic condition, respectively. To compare energy consumed for downlink communications, we define the efficiency of each protocol as a performance metric. The efficiency, defined as the ratio of data transmission time to the duty cycle, is the fraction of time used for useful work.

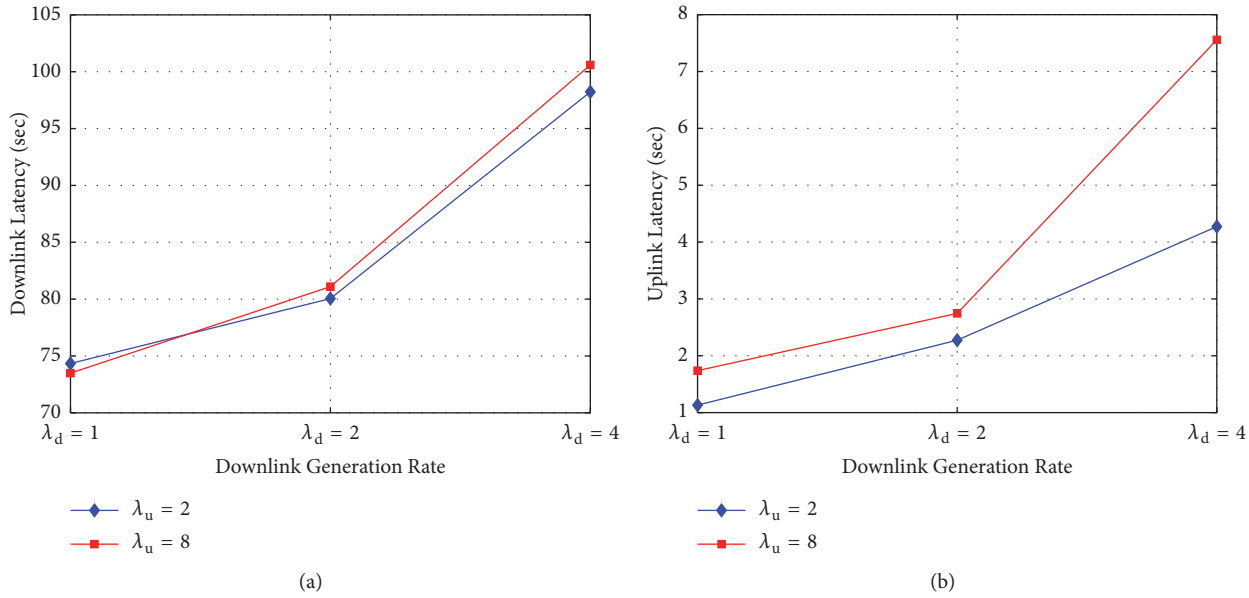


Figure 12: Average latency of traffic. (a) Downlink latency. (b) Uplink latency.

Table 4: Notations used for Class B downlink protocol and TRILO.

Notation	Meaning
D_T	Duty cycle of TRILO
D_B	Duty cycle of the Class B downlink protocol
T^T_{Beacon}	Beacon transmission time of TRILO
T^B_{Beacon}	Beacon transmission time of the Class B downlink protocol
T_{poll}	Average poll frame transmission time
T_{down}	Average downlink frame transmission time
k	Parameter for the Class B's ping slots
I_E	Indicator function representing a node is backlogged or not

According to the LoRaWAN standard, implementation of the Class B downlink scheme discretizes the time into so-called *ping slots*. Each end-device requests 2^k $(0 \le k \le 7)$ ping slots per beacon period. We use the average case of the Class B where each end-device randomly selects a value of the parameter k between 0 and 7. Let us first introduce the notations for the analysis of LoRaWAN downlink protocol (Table 4).

We used SF 9 for beacon transmissions according to the LoRaWAN specification, KR920-923 [29]. The duty cycle of TRILO, D_T, is

$$D_T = \frac{T^T_{Beacon} + I_E \cdot \left(T_{poll} + T_{down}\right)}{Beacon\ Period} \tag{7}$$

Note that unlike T^B_{Beacon}, T^T_{Beacon} increases in proportion to the number of busy end-devices. The duty cycle of a Class B device, D_B, is

$$D_B = \frac{T^B_{Beacon} + 2^k \cdot T_{down}}{Beacon\ Period} \tag{8}$$

Class B devices have different duty cycles depending on randomly selected values of parameter k. Then, the efficiency is computed as

$$Efficiency = \frac{Downlink\ transmission\ time}{Duty\ cycles} \tag{9}$$

We measured the efficiency of TRILO and the Class B downlink protocol by varying the number of nodes from 50 to 4000 and the downlink offered load from 2 to 32 frames per beacon period. Each performance point is the average of 10,000 runs. We limit the maximum downlink traffic to 32 frames per beacon interval because uplink traffic is greater than downlink traffic in many LoRaWAN applications. Note that at $\lambda_d = 32$, about a half of beacon period is consumed for downlink communications if we assume that the downlink transmission time is 2 seconds on average.

The efficiency of each scheme is shown in Figure 13. Note that the offered load is aggregated load and the traffic intensity of individual node decreases as the number of nodes increases. The total times for downlink communications are fixed at each given offered load. However, the sum of duty cycles increases as the number of nodes increases. Therefore, in both protocols, the maximum efficiencies are realized at small networks, and the efficiency decreases as the network size increases. The results illustrate that efficiencies of TRILO are higher than those of the Class B downlink protocol throughout all downlink generation rates. The Class B scheme performs well in heavy traffic cases. However, the proposed scheme outperforms the Class B scheme even in heavy traffic conditions. In small networks, the performance gap decreases from about 13.9 times to about 3.3 times as λ_d increases from 2 to 32. Also, each gap in each offered load increases as the network size increases. In the case of TRILO, the average duty cycle decreases inversely proportional to the number of nodes because only busy devices are active during downlink

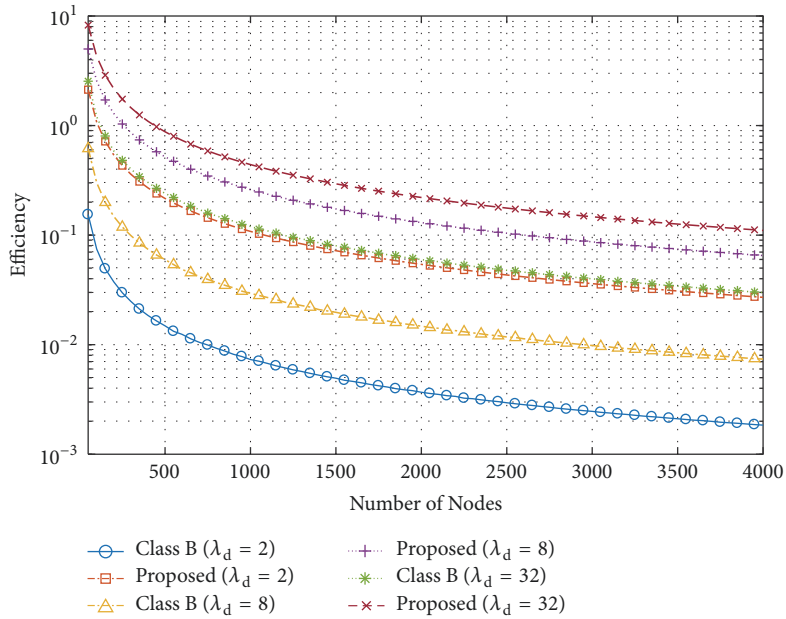


Figure 13: Efficiency of TRILO and Class B downlink protocol.

periods. On the contrary, Class B protocol allocates 31 ping slots on average regardless of downlink demands. In short, the number of receive windows wasted by the Class B scheme varies depending on the amount of downlink traffic; however, only busy devices use extra duty cycle for downlink traffic in TRILO.

The Class B protocol does not coordinate traffic between uplinks and downlinks, so that several uplinks can be disrupted by downlink transmissions due to the half-duplex characteristic of a LoRaWAN gateway; all reception paths can be blocked by downlink transmissions. It can obstruct the proper operations of typical LoRaWANs where uplink transmissions are predominant. In contrast, the proposed scheme is free from this problem because it isolates the time for uplinks and downlinks periods.

9. Conclusions

A plethora of LPWAN techniques such as LoRa, SigFox, RPMA, and Weightless have emerged as a new enabling network technology for IoT applications that require low-rate, low-cost, and long lifetimes. LPWANs allow end-device initiated uplinks. However, to maximize the lifetime of battery powered end-devices, nodes are put into the sleep state such that rather restricted downlink communications are permitted. In this paper, we proposed a new downlink protocol, named TRILO, for LoRaWAN.

Because LoRaWAN assumes a star topology where multiple end-devices are connected to a single gateway, we adopted a beacon based downlink mechanism similar to the IEEE 802.11 PSM standard. A gateway broadcast beacons periodically at every beacon period and end-devices are time synchronized. Beacons alert nodes with buffered downlinks in addition to synchronizing end-devices' clocks. Customized to the physical characteristics of LoRa, we designed two polling schemes: concurrent polling and sequential polling. An analysis indicates that sequential polling is more energy efficient than concurrent polling.

We implement the proposed beacon based downlink protocol using commercially available components and conduct experiments with modified devices. Our performance study shows that Packet Delivery Ratio (PDR) is almost 100% for both uplink and downlink traffic, and the extra wake-up time to receive beacons and downlinks is minimal. Simulation results also show that our proposed protocol can adaptively allocate receive windows resulting in lower energy consumption.

For further work, we plan to investigate the impact of our scheme on the networks with multiple gateways. Also, detailed investigations of impact of the number of downlinks, and the certain distributions of SFs on the performance seem interesting. Optimization techniques leveraging SF and power allocation are expected to affect the performance of the proposed scheme.

Data Availability

The data used to support the findings of this study are included within the article.

Disclosure

The funding sponsors had no role in the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; and in the decision to publish the results.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIP) (No. 2016R1A5A1012966), Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2015-0-00557, Resilient/Fault-Tolerant Autonomic Networking Based on Physicality, Relationship and Service Semantic of IoT Devices), and the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2015-0-00378) supervised by the IITP (Institute for Information & Communications Technology Promotion). Also, the snu-samsung smart campus research center at Seoul National University provides research facilities for this study.

References

[1] M. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa low-power wide-area networks scale?" in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM 2016*, pp. 59–67, Malta, November 2016.

[2] D. Magrin, M. Centenaro, and L. Vangelista, "Performance evaluation of LoRa networks in a smart city scenario," in *Proceedings of the 2017 IEEE International Conference on Communications, ICC 2017*, France, May 2017.

[3] LoRa, https://www.lora-alliance.org/What-Is-LoRa/Technology.

[4] "LoRaWAN 1.1 specification," LoRa Alliance, 2017, https://lora-alliance.org/resource-hub/lorawantm-specification-v11.

[5] M. Bor, J. Vidler, and U. Roedig, "LoRa for the internet of things," in *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN '16)*, pp. 361–366, Junction Publishing, 2016.

[6] 3GPP, "Study on provision of low-cost Machine-Type Communications User Equipments based on LTE," TR 36.888 V12.0.0, 06, 2013.

[7] 3GPP, "Cellular system support for ultra-low complexity and low throughput Internet of Things," *TR 45.820 V13.1.0, 11*, 2015.

[8] J. Gozalvez, "New 3GPP Standard for IoT," *IEEE Vehicular Technology Magazine*, vol. 11, no. 1, pp. 14–20, 2016.

[9] R. Ratasuk, N. Mangalvedhe, and A. Ghosh, "Overview of LTE enhancements for cellular IoT," in *Proceedings of the 26th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communications, PIMRC 2015*, pp. 2293–2297, China, September 2015.

[10] H. Shariatmadari, R. Ratasuk, S. Iraji et al., "Machine-type communications: current status and future perspectives toward 5G systems," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 10–17, 2015.

[11] S. Landstrom, J. Bergstrom, E. Westerberg, and D. Hammarwall, "NBIoT: a sustainable technology for connecting billions of devices," *Ericsson Technology Review*, vol. 4, pp. 2–11, 2016.

[12] M. Lauridsen, I. Z. Kovacs, P. Mogensen, M. Sorensen, and S. Holst, "Coverage and capacity analysis of LTE-M and NB-IoT in a rural area," in *Proceedings of the 2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pp. 1–5, Montreal, QC, Canada, September 2016.

[13] Sigfox, http://www.sigfox.com.

[14] Weightless open standard, http://www.weightless.org.

[15] Ingenu RPMA, http://www.ingenu.com/technology/rpma/.

[16] A. Dunkels, "The ContikiMAC radio duty cycling protocol," Technical Report T2011:13, Swedish Institute of Computer Science, 2011.

[17] J. Ko, J. Eriksson, N. Tsiftes et al., "Industry: Beyond interoperability - Pushing the performance of sensor network IP stacks," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems, SenSys 2011*, pp. 1–11, USA, November 2011.

[18] D. Moss, J. Hui, and K. Klues, "TEP 105: low power listening," http://www.tinyos.net/tinyos-2.x/doc/pdf/tep105.pdf, 2008.

[19] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: an emerging low-power wireless technology," *Sensors*, vol. 12, no. 9, pp. 11734–11753, 2012.

[20] A. A. Kandhalu, A. E. Xhafa, and S. Hosur, "Towards bounded-latency bluetooth low energy for in-vehicle network cable replacement," in *Proceeding of the International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 635–640, Las Vegas, NV, USA, 2013.

[21] A. Montanari, S. Nawaz, C. Mascolo, and K. Sailer, "A study of bluetooth low energy performance for human proximity detection in the workplace," in *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications, PerCom 2017*, pp. 90–99, Kona, HI, USA, March 2017.

[22] C. Julien, C. Liu, A. L. Murphy, and G. P. Picco, "BLEnd: Practical continuous neighbor discovery for bluetooth low energy," in *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN 2017*, pp. 105–116, New York, NY, USA, April 2017.

[23] Y. Qiu, S. Li, X. Xu, and Z. Li, "Talk more listen less: energy-efficient neighbor discovery in wireless sensor networks," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications, IEEE INFOCOM 2016*, USA, April 2016.

[24] A. J. Berni and W. D. Gregg, "On the utility of chirp modulation for digital signaling," *IEEE Transactions on Communications*, vol. 21, no. 6, pp. 748–751, 1973.

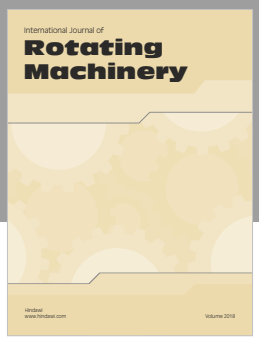
[25] A. Springer, W. Gugler, M. Huemer, L. Reindl, C. C. W. Ruppel, and R. Weigel, "Spread spectrum communications using chirp signals," in *Proceedings of the IEEE/AFCEA Information Systems for Enhanced Public Safety and Security, EUROCOMM 2000*, pp. 166–170, Germany.

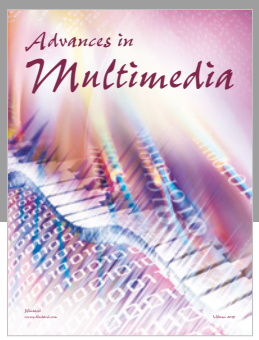
[26] IEEE Std 802.11-2007, "IEEE Standard for Information Technology—Telecommunications and Information Exchange between Systems—Local and Metropolitan Area Networks—Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, June 2007".

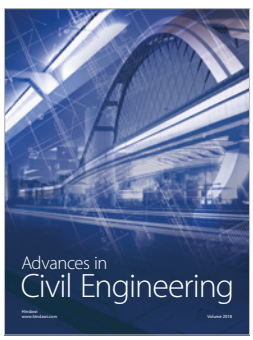
[27] RAK 831, http://www.rakwireless.com/en/WisKeyOSH/RAK831.

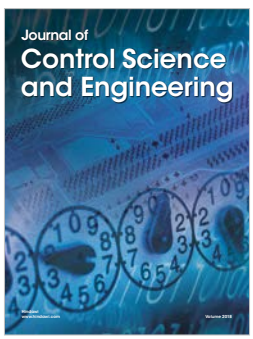
[28] LoRaWAN network server, https://github.com/gotthardp/lorawan-server.

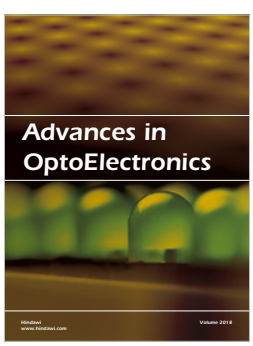
[29] "LoRaWAN 1.1 Regional Parameters," LoRa Alliance, 2017, https://lora-alliance.org/resource-hub/lorawantm-regional-parameters-v11ra.

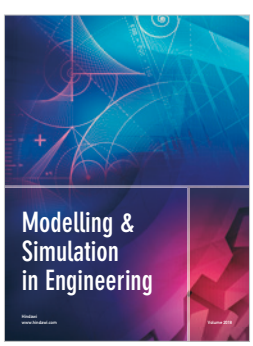


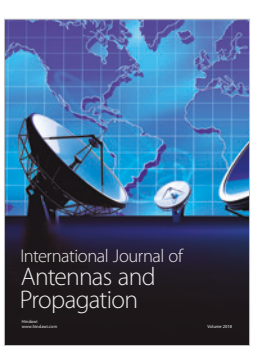


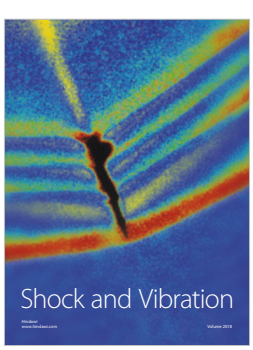


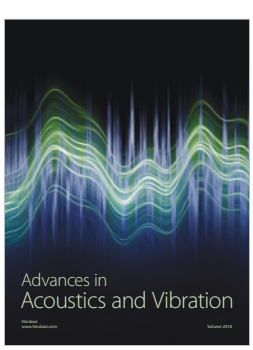












Hindawi

Submit your manuscripts at
www.hindawi.com