Cantor: Improving Goodput in LoRa Concurrent Transmission

Dan Xu, Xiaojiang Chen, *Member, IEEE*, Nannan Zhang, Nana Ding, Jing Zhang, Dingyi Fang, *Member, IEEE*, and Tao Gu, *Senior Member, IEEE*

***Abstract*—Long range (LoRa) is an attractive low-power wide-area networks (LPWANs) technology for its features of low power, long range, and support for concurrent transmission. Our study reveals LoRa concurrent transmission suffer from the mismatch between the sender's reception (RX) and gateway's transmission (TX) window, which leads to the decline of goodput even the throughput is improved. Our experiment shows that goodput only accounts for two-fifths of the throughput in concurrent transmissions with 48 nodes at a duty cycle of 20%. This article presents a window match scheme named Cantor which improves the goodput of LoRa concurrent transmission by controlling the RX window size. Cantor does not require the frequent exchange of controlling information. Instead, it introduces a novel concurrent transmission model to estimate the downlink packet reception rate (PRR) with different network parameters, and a regression model is used to make the result more realistic. Then, we propose a simple optimization algorithm to select optimal RX window sizes in which nodes are able to receive acknowledgments. We implement and evaluate Cantor with commodity LoRa gateway and nodes, and conduct experiments in different scenarios. The experimental results show that Cantor increases the goodput by 70% and reduces energy consumption by 30% in LoRa concurrent transmissions with 48 nodes operate at a duty cycle of 20%.**

***Index Terms*—Concurrent transmission, goodput, long range (LoRa), window mismatch.**

I. Introduction

LPWANS have been playing a key role in the Internet of Things (IoT) [1]. As one of the major industry initiated low-power wide-area networks (LPWANs) technologies, Semtech's long-range (LoRa) technology and its open LoRaWAN protocol developed by LoRa alliance [2] offer an efficient, flexible, and economical solution to real-world problems [3]–[12], and several enterprises have built their LoRa ecosystems, such as Alibaba Cloud, MachineQ, and Vinduino, 140 million LoRa nodes to be deployed in 51 countries by the end of 2020 [13].

It is known that LoRa is popular for its features of long range, low power, and support for concurrent transmission. Concurrency increases throughput in LoRa significantly, however, from our real-world experiments in LoRa concurrent transmissions, we observe that the majority of improved throughput are not goodput, but duplicate packets as detailed in Section II. Transmitting duplicate packets not only wastes bandwidth but also consumes more energy.

We analyze this phenomenon and discover that the sustained concurrent uplink transmissions may block downlink transmissions, and any failure of downlink transmissions cause retransmissions, which decreases the goodput eventually. We analyze this phenomenon by giving an example, as illustrated in Fig. 1. Three nodes transmit data to the gateway randomly, the end times of these transmissions are different due to their different start times, packet sizes, and data rates. These differences may keep the gateway in the *RX* mode for a long period in order to receive uplink transmissions. When sender 1 completes its transmissions, it opens an RX window (i.e., RX1, in which node receives downlink transmissions, not the mode of the gateway) to receive downlink transmissions (i.e., ACK), but the gateway is busy receiving packets from sender 2 and cannot acknowledge sender 1. When sender 2 completes its transmission, since there are no incoming packets, the gateway is switched to the *TX* mode for sending an ACK for sender 1 (this period is also called as gateway's TX window in the following sections), but sender 1 has closed its RX1 window to save energy. When sender 1 reopens another RX window (i.e., RX2), the gateway has been occupied by sender 3's uplink transmissions. Sender 1 finally starts retransmission, but in turn, the retransmission will block the downlink transmissions for sender 3.

In a nutshell, when a single node transmits data to the gateway, the gateway can successfully acknowledge the sender because its TX window matches the sender's RX window mutually. However, when multiple nodes transmit data concurrently, the downlink transmissions to acknowledge senders can only be activated when the gateway completes receiving the current uplink transmission. Although the existing design in LoRa gateway has two radio chips and supports full duplex, its radio-frequency (RF) module is occupied by uplink transmissions (i.e., its both radios in the *RX* mode) while concurrent uplink transmissions occur. In this case, the gateway's TX window may miss the sender's RX windows, resulting in ACK lost

Manuscript received February 21, 2020; revised April 28, 2020 and June 22, 2020; accepted July 24, 2020. Date of publication July 31, 2020; date of current version January 22, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61802309, Grant 61672428, and Grant 61772422; in part by the ShaanXi Science and Technology Innovation Team Support Project under Grant 2018TD-026; in part by the Shaanxi International Joint Research Centre for the Battery Free Internet of Things under Grant 2018SD0011; and in part by the Australian Research Council (ARC) Discovery Project under Grant DP190101888. *(Corresponding author: Xiaojiang Chen.)*

Dan Xu, Xiaojiang Chen, Nannan Zhang, Nana Ding, Jing Zhang, and Dingyi Fang are with the School of Information Science and Technology, Northwest University, Xi'an 710069, China (e-mail: xjchen@nwu.edu.cn).

Tao Gu is with the School of Computer Science, RMIT University, Melbourne, VIC 3001, Australia.

Digital Object Identifier 10.1109/JIOT.2020.3013315

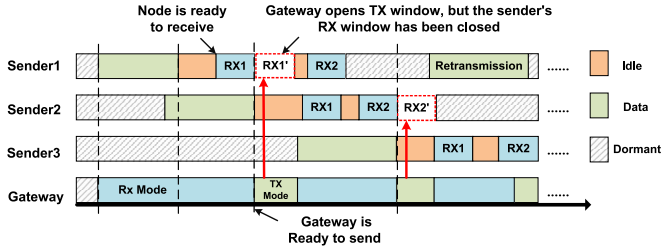


Fig. 1. Window mismatch problem in LoRa.

which triggers retransmission. The mismatch between gateway's TX window and the sender's RX window is essentially due to the fact that sustained concurrent uplink transmissions block downlink transmissions.

To address the window mismatch problem, negative acknowledgment (NAK) [14] can be used to avoid acknowledging all received packets, and the gateway only responds to a node if its packets are lost. However, such downlink responses may still be blocked by uplink transmissions. To make it worst, the lost packets may have been flushed from the node's memory.

Designing an effective window match control scheme for LoRa concurrent transmission is not a trivial task. From our experimental studies detailed in Section II, we observe that although uplink transmissions from 48 nodes with a duty cycle of 20% occupy most of the time slots of the gateway, the gateway can still find available time slots for downlink transmissions. If the sender's RX windows can be controlled to match downlink transmissions, the window mismatch problem can be significantly alleviated or avoided completely. For example, as illustrated in Fig. 1, sender 1 may postpone its RX1 to RX1′ to receive downlink transmissions, and sender 2 may do likewise.

However, controlling the RX window requires frequent exchange of control messages between the gateway and nodes, which will inevitably introduce extra overhead. Thereby, it is not feasible in LoRa for the following constraints: 1) LoRa node is low power, and frequent control information exchange consumes more energy and 2) it is more likely that the control information to nodes will be blocked by uplink transmissions, similar to the case of ACK.

In addition, prolonging RX window size may also enhance the probability of matching RX windows and downlink transmissions without frequent information exchange. However, it may increase the whole energy consumption of nodes due to longer RX windows consume more energy. This article essentially studies the theoretical model of the relationship between the RX window size and the reception probability of downlink transmissions, and on this basis, improves the reception probability of downlink transmissions without increasing additional energy consumption of nodes.

In this article, we propose a control scheme named Cantor to address the window mismatch problem in LoRa concurrent transmission. The design of Cantor stems from the following key insight: the node access time distribution follows a uniform distribution, which is the result of probability statistics. Based on this insight, Cantor constructs a theoretical concurrent transmission model to estimate the downlink PRR with different network parameters and obtain a realistic result through regression. Based on this model, the gateway selects an appropriate RX window size to achieve optimal goodput. An appropriate RX window size can be sent to each node during the initial network phase, avoiding exchanging control information frequently during the operation of the network.

The contributions of this article are summarized as follows.

Contributions:

1) We discover the window mismatch problem in LoRa concurrent transmission and its negative impact on goodput through our extensive experiments. Our analysis shows that this problem can be significantly alleviated if the RX window size can be well controlled.
2) We propose a novel window match control scheme named Cantor to address the challenges in controlling RX window size, which consists of: a) a LoRa concurrent transmission model based on the node access time distribution, to estimate theoretical downlink PRR and b) a simple optimization algorithm named wane and wax (WW) to select an appropriate RX window size in advance.
3) We implement Cantor on LoRa devices and deploy experiments in four different scenarios, including both indoor and outdoor. Our results reveal that with 48 LoRa nodes deployed, Cantor achieves about 70% goodput gain and decreases energy consumptions by 30% over prior LoRa transmission systems.

II. Preliminary and Motivation

In this section, we first present our motivation behind the design of Cantor, then conduct a thorough experimental study to reveal the issues that exist in LoRa's concurrent transmission.

A. Preliminary

In LoRa, the gateway can scan eight channels (*IF*0–*IF*7) for preambles at all times. Besides, packets using different spreading factors (SFs) ($SF = 6 \sim 12$) can be demodulated simultaneously even in the same channel ($SF = 6$ is used for specific circumstances, and we do not consider it in concurrency).

Moreover, the existing design in LoRa gateway has two radio chips and supports full duplex, but it cannot start downlink transmissions while concurrent uplink transmissions occur since its RF module is occupied by uplink transmissions (i.e., both radios in the *RX* mode). In this case, the gateway has to switch at least one of the two radios from *RX* to *TX* in order to start a downlink. The change in mode would disable all uplink traffics in the channels that are connected to the radio. On the other hand, if the uplink transmission is on continuously, the downlink ACK will miss the RX windows of nodes.

B. Downlink Transmission Versus Concurrent Uplink Transmission

In a wireless network, throughput and goodput are two important metrics to measure transmission performance.

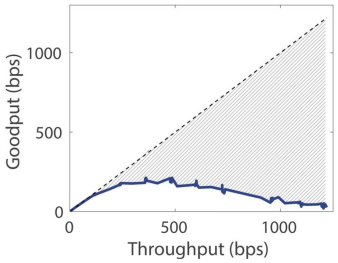


Fig. 2. Throughput versus goodput.

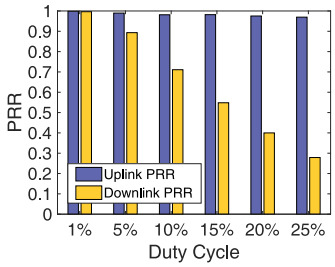


Fig. 3. Uplink versus downlink.

Throughput refers to the end-to-end number of bytes received and goodput refers to the effective bytes that exclude retransmitted packets. Besides, PRR is another important metric, and we consider both uplink and downlink PRRs.

In this section, we present the preliminary results about goodput and downlink PRR in LoRa concurrent transmission, which motivate the design of an effective window match control scheme.

Experimental Settings: We deploy a LoRa gateway and 48 LoRa nodes in different environments (i.e., a 15 m × 18 m office, a 7.5 m × 6.6 m lab, a playground, and an open area near our school) for experiments. These nodes operate in eight different channels and each channel has six different SFs, the size of RX window is set to 20 ms, the initial duty cycle of node is set to 0.01, and then gradually increase to 0.25. Each experiment lasts about 2 h. The detailed information about our experiment settings is shown in Table I.

Throughput and Goodput: Fig. 2 depicts the relation between goodput and throughput in LoRa concurrent transmissions. Unlike what we expect that both should grow linearly, the goodput drops significantly with the throughput increases. We analyze this phenomenon and discover that the main reason is that retransmitted packets account for a large proportion of the traffic.

PRR of Uplink and Downlink: Fig. 3 shows the PRR of uplink and downlink with different duty cycles. These results further explain the reason why there is a difference between throughput and goodput, i.e., uplink transmissions block a significant amount of downlink traffics in order to finish the ongoing concurrent transmissions. It hence results in higher uplink PRR but lower downlink PRR.

C. Node's RX Window Matches Gateway's TX Window

The above results reveal that the goodput is much lower than throughput in LoRa concurrent transmission. This section presents our analysis of the results.

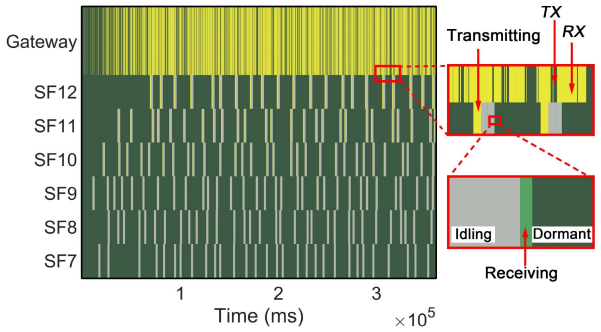


Fig. 4. States of gateway and nodes.

We first study both uplink and downlink transmissions of LoRaWAN and then analyze the changes of gateway and nodes' states. The gateway has two states: 1) receiving (*RX* mode) and 2) transmitting (*TX* mode), and the node has four states: 1) sending; 2) idling; 3) receiving; and 4) dormant. As a specific result, Fig. 4 shows the states of the gateway and nodes. From the first glance of Fig. 4, the gateway is almost fully occupied by uplink transmissions. Therefore, it is crucial to know if the gateway has no chance to make downlink transmissions.

We now analyze this example and explain the reason behind it. In the gateway's state result, the total time of *RX* mode accounts for 71.2%, and only 98 303 ms are left for downlink transmissions. During this experiment, 1279 packets are received by the gateway, and 15 × 1279 = 19 185 ms are needed for acknowledging these packets, which is just a tiny fraction in terms of gateway's total *TX* time ($[19185/98303] \approx 0.195$). It implies that the overall time of the *TX* mode is enough for the gateway to acknowledge all senders.

We further analyze the CDF of gateway's *RX* and *TX* duration. Fig. 5(a) plots the CDF of *TX* durations, and the *TX* durations that less than 15 ms only account for 6.54%, and most *TX* durations are more than 15 ms (in these experiments, the length of ACK is about 15 ms), it means that each *TX* duration is short, but it is able to afford the transmission of ACK.

We plot the CDF of *RX* durations in Fig. 5(b), which shows that 90% of gateway's *RX* durations fall between 45 and 1000 ms. Considering the node's receive delays are about 1000 ms and exceed most of the gateway's *RX* durations, it means that the node is able to match at least one gateway's *TX* duration.

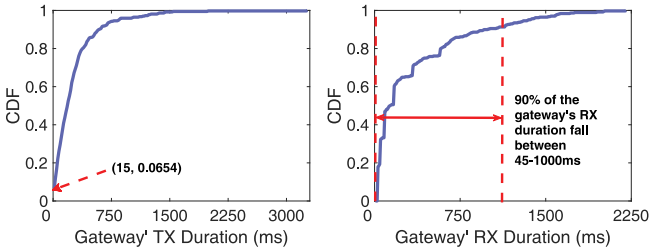
From the above analysis, we conclude that even 48 nodes send packets to the gateway concurrently with a duty cycle of 20%, the gateway has enough time to acknowledge all the nodes.

III. Network Model and Problem Formulation

We consider a multichannel, multi*SF* LoRa network which consists of one gateway and n LoRa nodes. There are an uplink and a downlink between each node and the gateway. The network can be represented by a star with the gateway at the center and nodes connected to the gateway directly.

TABLE I
DETAILED EXPERIMENTAL SETTINGS

SF	Air time of each packet (ms)	Active time (s)	Data rate of each SF node with different duty cycles (packet/s)					
			1%	5%	10%	15%	20%	25%
7	46	2.086	0.0048	0.0240	0.0479	0.0719	0.0959	0.1198
8	93	2.136	0.0047	0.0234	0.0468	0.0702	0.0936	0.1170
9	164	2.204	0.0045	0.0227	0.0454	0.0681	0.0907	0.1134
10	323	2.363	0.0042	0.0212	0.0423	0.0635	0.0846	0.1058
11	660	2.7	0.0037	0.0185	0.0370	0.0556	0.0741	0.0926
12	1150	3.19	0.0031	0.0157	0.0313	0.0470	0.0627	0.0784
Total data rate of all 48 nodes with different duty cycles (packet/s)			0.2007	1.0033	2.0067	3.0100	4.0133	5.0166



(a) (b)

Fig. 5. CDF of gateway's state duration. (a) *TX* duration. (b) *RX* duration.

TABLE II
NOTATIONS IN PROBLEM FORMULATION

Notation	Meaning
μ	Duty cycle
T_{sf}	Work cycle duration for different SF nodes
δ	Time slot
a_{sf}	Active time
$l_{sf}\delta$	The packet air time
$\gamma\delta$	The ACK message air time
t_1	Receive delay for the first RX window
t_2	Receive delay for the second RX window
t_a	The size of RX window
R_{max}	The maximum number of retransmission

The total number of nodes with different SFs is $n_{sf}, sf \in [7, 12]$, respectively, each node works with a certain duty cycle of $\mu = (a_{sf}/T_{sf})$, Table II lists notations used in this article.

Energy Consumption: A LoRa node has four states: 1) dormant; 2) idle; 3) receiving; and 4) sending. According to the datasheet of SX1278 [15], in the dormant state, radio module and MCU are all off, the energy consumption in this state is $I_{\text{dormant}} < 0.2\ \mu$A; in the idle state, RC oscillator is enabled, the energy consumption is about $I_{\text{idle}} = 1.6$ mA; in the receiving and sending states, MCU and radio modules are on, and the energy consumed is more than that in other states, they are about $I_{\text{receive}} = 12$ mA and $I_{\text{send}} = 120$ mA (maximum power).

Therefore, for a node that sends data with $SF = x$, the consumed energy in a work cycle duration T_x is $ET_x^{(t_1,t_2,t_a)}$

$$ET_x^{(t_1,t_2,t_a)} \approx t_{\text{dormant}} \cdot I_{\text{dormant}} + (t_1 + t_2) \cdot I_{\text{idle}} + 2t_a \cdot I_{\text{receive}} + l_x\delta \cdot I_{\text{send}}. \quad (1)$$

Problem Formulation: Our main goal is to send data and receive ACK at proper durations that maximizes the expected goodput, while the energy consumed for sending a packet successfully does not increase. According to the analysis of Section II-B, the main reason for the decrease of goodput is unnecessary retransmissions (unnecessary retransmission is the retransmission initiated when the gateway has received the packet from the node, but the node has not received the ACK from the gateway). Although maximizing downlink PRR can reduce unnecessary retransmissions and improve goodput directly, we define the objective function as minimization of the expected transmission times, to introduce the energy consumption into our optimization goal through expected transmission times. Therefore, the optimization problems can be formulated as follows:

$$\text{Min} \quad \mathbb{E}(Rt) = \sum_{i=1}^{R_{\max}} i \cdot p_r(1-p_r)^{i-1}, i \in [1, R_{\max}] \quad (2)$$

$$\text{Subject to} \quad t_1 \geq t_{\text{process}} \quad (3)$$

$$t_1 + t_2 + 2t_a + l_x\delta = t_{\text{active}} \quad (4)$$

$$\Delta E_x = E_x^{(t_1',t_2',t_a')} - E_x^{(t_1,t_2,t_a)} \leq 0 \quad (5)$$

where $\mathbb{E}(Rt)$ is the expected transmission times of node for a certain packet, i is transmission times, and p_r is the probability that the sender receives the ACK from the gateway. $E_x^{(t_1,t_2,t_a)}$ is the average energy consumed for transmitting a packet successfully (it means that not only the gateway receives the packet but also the sender node receives the ACK) with parameters t_1, t_2, and t_a, $E_x^{(t_1,t_2,t_a)}$ can be expressed as

$$E_x^{(t_1,t_2,t_a)} = \mathbb{E}(Rt) \cdot ET_x^{(t_1,t_2,t_a)} \quad (6)$$

where t_1', t_2', and t_a' are the updated t_1, t_2, and t_a, the detailed update steps are shown in Algorithm 1. ΔE_x is the difference between the updated energy consumption and the one before update.

Although the problem has been formulated, we cannot solve this optimization problem directly because the downlink PRR is unknown. Therefore, our solution is to design a model to calculate the downlink PRR under each network setting, and then find out the appropriate parameters that satisfy the objective function and constraints.

IV. CONCURRENT TRANSMISSION CONTROL SCHEME

A. Cantor Overview

We design an effective control scheme named Cantor, aiming to achieve high goodput in LoRa concurrent transmission. Cantor operates at the gateway and can essentially increase downlink PRR at the node through searching the optimal transmitting parameters. Cantor has three major components: 1) a theoretical concurrent transmission model; 2) a regression

Algorithm 1 Pseudo of Cantor

Require:
Number of node with different *SF*s: n_{sf}, $sf \in [7, 12]$;
The packet air time of different nodes: $l_{sf}\delta$;
Active time of different nodes: a_{sf};
Duty cycle of nodes: μ;
Statistical downlink PRR of different nodes: $p^{sf}_{r\_r}$;
Ensure: the optimized t_2 and t_a;
1: Initialization: $t_1 = t_2 = 1000$ ms, $t_a = 20$ ms, time step: $i = 1$;
2: Calculate the theoretical downlink PRRs of different nodes p^{sf}_r using (18);
3: Establish the realistic downlink PRRs p^{sf} using regression model;
4: Calculate the expected transmission times $\mathbb{E}(Rt)$ using (2);
5: Calculate the consumed energy of different nodes $ET^{(t_1,t_2,t_a)}_{sf}$ using (1);
6: **while** (3), (4), (5) **do**
7: $t_a = t_a + i$;
8: $t_2 = t_2 - 2i$;
9: Update p^{sf}_r using (18);
10: Update p^{sf} using regression model;
11: Update $ET^{(t_1,t_2,t_a)}_{sf}$ and $\mathbb{E}(Rt)$ using (1) and (2);
12: **end while**
13: **Return** t_2 and t_a;

model; and 3) an optimizer. To understand how Cantor works, Fig. 6 depicts these three components, and we describe each component as follows.

1) *Theoretical Concurrent Transmission Model:* This model is constructed to calculate the downlink PRR. In the initial stage of the network, the gateway collects packets from nodes, Cantor analyzes these packets to obtain the network parameters (such as the total number of nodes with different SFs, the distribution of nodes in each channel, the duty cycles, the size of data, receive delay of t_1 and t_2, and the size of RX window) and set these parameters in the model to calculate the theoretical downlink PRR. How to build the model is detailed in Section IV-B.
2) *Concurrent Transmission regression Model:* Cantor calculates the downlink PRR through the theoretical model, however, the theoretical model does not consider clock skew and channel quality which may vary in real scenarios, as a result, the theoretical PRR obtained may not be accurate. Therefore, a regression model should be applied to get a more realistic PRR before the optimizer. How to train the regression model is detailed in Section IV-C.
3) *Optimizing the Network Parameters:* Based on the above two models, Cantor estimates the real downlink PRR with different network parameters, the optimization algorithm called WW could select the optimal transmission parameters that satisfy the objective function for nodes. This process is described in Section IV-D.

B. Theoretical Concurrent Transmission Model

We analyze LoRa concurrent transmission with multichannel and multi*SF*. Specifically, multiple nodes with different SFs transmit data concurrently on the same

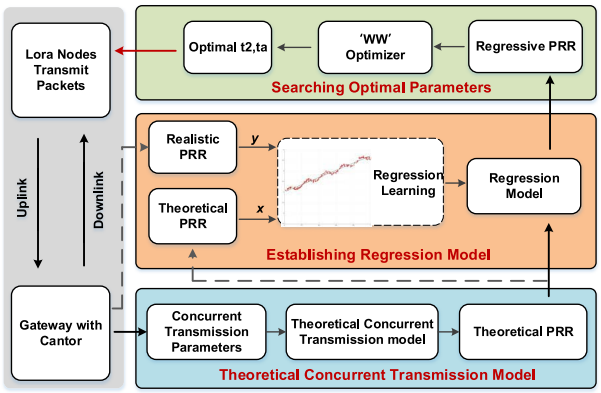


Fig. 6. Overview of cantor.

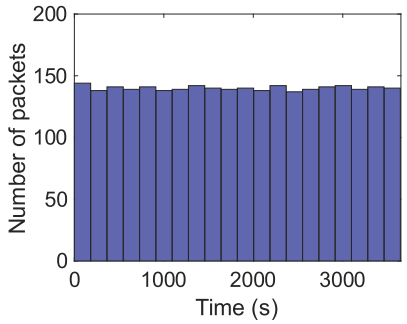


Fig. 7. Packets' access times distribution.

channel, and on the other hand, nodes occupy different channels that may transmit data concurrently with any *SF*.

Total Number of Nodes Accessing Gateway: In order to analyze concurrent transmission, the access process of nodes should be carefully modeled. Here, we study the distribution of node access time through an experiment that is conducted in Section II-B, and we record nodes' access times and find that the numbers of packets in each time period are similar. As shown in Fig. 7, we divide 1 h experiments' data into 20 durations to imply the packets' access times distribution, the result depicts that the node access time fits uniform distribution.

As shown previous, the node operates with a duty cycle of μ in a work cycle T_{sf}, a node accesses the gateway randomly within T_{sf}, and the access time is a random variable B^{sf}_i, $i = 1, 2, \ldots, n_{sf}$. Each random variable is independently and identically distributed, thus we use the uniform distribution to describe node access time: $B^{sf}_i \sim U[0, C_{sf}]$, $C_{sf} = T_{sf} - \alpha_{sf}\delta$.

Size of RX Window: We now analyze the effect of the RX window size. As shown in Fig. 8, the RX window size is t_a that is larger than ACK transmission time $\gamma\delta$. Thus, the sender has $(t_a/\delta) - \gamma_s + 1$ time slots (assuming a time slot is 1 ms) to start receiving ACK within the RX window.

Probability of Receiving ACK in RX1: If a node with $SF = s$ accesses the gateway at t_0 and receives the gateway's reply in the first RX window (RX1), the following requirements must be satisfied.

1) No node with $SF > 7$ accesses the gateway in the time period $[t_0 + l_s\delta + t_1 - l_7\delta + (i-1)\delta, t_0 + l_s\delta + t_1 + (i-1)\delta]$,

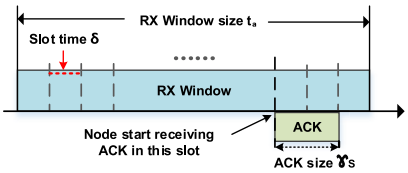


Fig. 8. ACK reception window (RX window).

and no node with $SF = 7$ accesses the gateway in the time period $[t_0+l_s\delta+t_1-l_7\delta+\delta+(i-1)\delta, t_0+l_s\delta+t_1+(i-1)\delta]$, which $i \in [1, (t_a/\delta) - \gamma_s + 1]$. This probability is calculated in (7). If the ACK receiving is started at the ith slot of RX1, and the time moment of this slot is $t_0 + l_s\delta + t_1 + (i-1)\delta$ (we use $t^{(1)}$ to simplify the expression), it means that there is no uplink transmission at this moment or an uplink transmission is just finished. If there is an uplink transmission with $SF = 7$ is just finished at $t^{(1)}$ [we consider other uplink transmission cases with $SF > 7$ in requirement 2)], the latest start time of this transmission is $t^{(1)}-l_7\delta$, otherwise, the transmission cannot finish at $t^{(1)}$, and the ACK receiving is blocked. Therefore, there is no node with $SF > 7$ accessing the gateway in the time period $[t^{(1)}-l_7\delta, t^{(1)}]$, and no node with $SF = 7$ accessing the gateway in the time period $[t^{(1)}-l_7\delta+\delta, t^{(1)}]$. As shown in Fig. 9, the longer gray rectangle presents this period. The probability $p^s_{cr1\_1}$ is calculated as follows:

$$p^s_{cr1\_1} = \left[1 - P_7\left(t^{(1)} - l_7\delta + \delta \le t \le t^{(1)}\right)\right] \times \prod_{j=8}^{12}\left[1 - P_j\left(t^{(1)} - l_7\delta \le t \le t^{(1)}\right)\right]. \quad (7)$$

In (7), $1-P_7(t^{(1)} - l_7\delta + \delta \le t \le t^{(1)})$ denotes the probability that no node with $SF = 7$ accesses the gateway in the time period $[t_0+l_s\delta+t_1-l_7\delta+\delta+(i-1)\delta, t_0+l_s\delta+t_1+(i-1)\delta]$, and $\prod_{j=8}^{12}[1-P_j(t^{(1)}-l_7\delta \le t \le t^{(1)})]$ denotes the probability that no node with $SF > 7$ accesses the gateway in the time period $[t_0+l_s\delta+t_1-l_7\delta+(i-1)\delta, t_0+l_s\delta+t_1+(i-1)\delta]$.

2) No node with $SF = x$ accesses the gateway in the time period $[t_0+l_s\delta+t_1-l_x\delta+\delta+(i-1)\delta, t_0+l_s\delta+t_1-l_{(x-1)}\delta+(i-1)\delta]$, and no node with $SF = x+1 \sim 12$ accesses the gateway in the time period $[t_0+l_s\delta+t_1-l_x\delta+(i-1)\delta, t_0+l_s\delta+t_1-l_{(x-1)}\delta+(i-1)\delta]$, where $x \in [8, 12]$. This probability is calculated in (8). If there is an uplink transmission with $SF = x, x \in [8, 12]$ which is just finished at $t^{(1)}$, the latest start time of this transmission is $t^{(1)} - l_x\delta$. Hence, in order to ensure the ACK receiving at time $t^{(1)}$, there are no node with $SF > x+1$ accesses the gateway in the time period $[t^{(1)}-l_x\delta, t^{(1)}-l_{x-1}\delta]$, and no node with $SF = x$ accesses the gateway in the time period $[t^{(1)}-l_x\delta+\delta, t^{(1)}-l_{x-1}\delta]$. Besides, if there exists x such that $t^{(1)} - l_x\delta \le 0$, let $x_{\max} = x$. As shown in Fig. 9, the shorter gray rectangle depicts this period. The probability $p^s_{cr1\_2}$ is calculated as follows:

$$p^s_{cr1\_2} = \prod_{x=8}^{x_{\max}}\left[1 - P_x\left(t^{(1)} - l_x\delta + \delta \le t \le t^{(1)} - l_{x-1}\delta\right)\right] \times \prod_{m=x}^{12}\left[1 - P_m\left(t^{(1)} - l_x\delta \le t \le t^{(1)} - l_{x-1}\delta\right)\right]. \quad (8)$$

In (8), $1 - P_x(t^{(1)} - l_x\delta + \delta \le t \le t^{(1)} - l_{x-1}\delta)$ denotes the probability that no node with $SF = x$ accesses the gateway in the time period $[t_0 + l_s\delta + t_1 - l_x\delta + \delta + (i-1)\delta, t_0 + l_s\delta + t_1 - l_{(x-1)}\delta + (i-1)\delta]$, and $\prod_{m=x}^{12}[1 - P_m(t^{(1)} - l_x\delta \le t \le t^{(1)} - l_{x-1}\delta)]$ denotes the probability that no node with $SF = x+1 \sim 12$ accesses the gateway in the time period $[t_0 + l_s\delta + t_1 - l_x\delta + (i-1)\delta, t_0 + l_s\delta + t_1 - l_{(x-1)}\delta + (i-1)\delta]$.

3) When $i \ge 2$, it indicates that the ACK transmission start time is delayed, as shown in Fig. 8, meaning that the first $i-1$ slot of RX has been occupied, and a data transmission has just finished at slot $i-1$, and that is, at least one type of SF node accesses gateway for data transmission in the time period $[t_0+l_s\delta+t_1-l_x\delta+(i-2)\delta, t_0+l_s\delta+t_1-l_x\delta+(i-1)\delta]$. If x exists such that $t_0+l_s\delta+t_1-l_x\delta+(i-2)\delta \le 0$, let $x_{\max\_1} = x$, if not, let $x_{\max\_1} = 12$. The probability $p^s_{cr1\_3}$ is calculated in

$$p^s_{cr1\_3} = 1 - \prod_{x=7}^{x_{\max\_1}}\left[1 - P_x\left(t^{(1)} - l_x\delta - \delta \le t \le t^{(1)} - l_x\delta\right)\right] \quad (9)$$

where $\prod_{x=7}^{x_{\max\_1}}[1 - P_x(t^{(1)} - l_x\delta - \delta \le t \le t^{(1)} - l_x\delta)]$ denotes the probability that no node with $SF = 7 \sim x_{\max\_1}$ accesses the gateway in the time period $[t_0 + l_s\delta+t_1-l_x\delta+(i-2)\delta, t_0+l_s\delta+t_1-l_x\delta+(i-1)\delta]$, $1 - \prod_{x=7}^{x_{\max\_1}}[1 - P_x(t^{(1)} - l_x\delta - \delta \le t \le t^{(1)} - l_x\delta)]$ denotes that there is at least one kind of SF node accesses gateway for data transmission in this time period, and this is the reason why the start time of ACK is delayed. So far, the probability of receiving the gateway's reply for node with $SF = s$ in the first RX window is calculated in

$$p^s_{cr1} = \{p^s_{cr1\_1} * p^s_{cr1\_2} | i = 1\} + \sum_{i=2}^{\frac{t_a}{\delta}-\gamma_s+1} p^s_{cr1\_1} * p^s_{cr1\_2} * p^s_{cr1\_3} \quad (10)$$

where $\{p^s_{cr1\_1} * p^s_{cr1\_1} | i = 1\}$ is the probability that the ACK transmission is started in the first slot of $RX1$ window, and $\sum_{i=2}^{(t_a/\delta)-\gamma_s+1} p^s_{cr1\_1} * p^s_{cr1\_2} * p^s_{cr1\_3}$ means the ACK transmission is postponed to the following slots.

Probability of Receiving ACK in RX2: If a node with $SF = s$ accesses the gateway at time t_0 and receives the gateway's reply in the second RX window (RX2), the following requirements must be satisfied.

1) No node with $SF > 7$ accesses gateway in the time period $[t_0 + l_s\delta + t_1 + t_a + t_2 - l_7\delta + (i-1)\delta, t_0 + l_s\delta + t_1 + t_a + t_2 + (i-1)\delta]$, and no node with $SF = 7$ accesses gateway in the time period $[t_0 + l_s\delta + t_1 + t_a + t_2 - l_7\delta +$

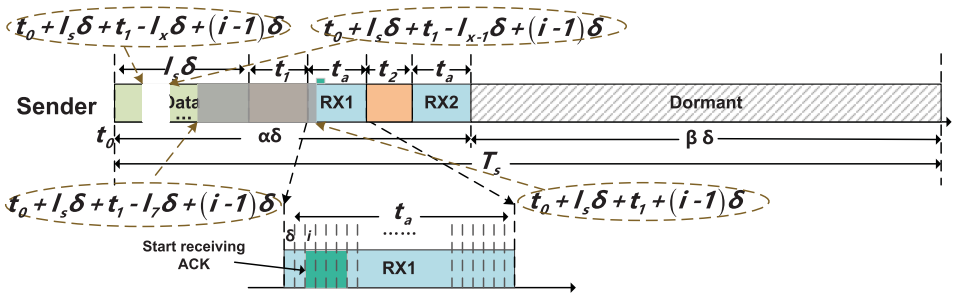


Fig. 9. Sender receives reply in RX1.

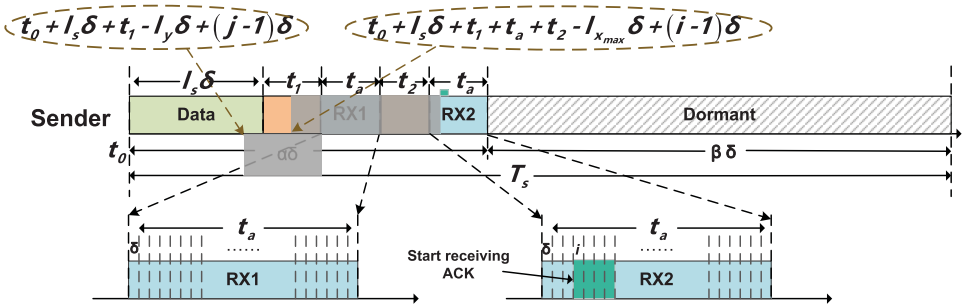


Fig. 10. Sender receives reply in RX2.

$\delta + (i-1)\delta, t_0 + l_s\delta + t_1 + t_a + t_2 + (i-1)\delta]$ where $i \in [1, (t_a/\delta) - \gamma_s + 1]$. If the ACK receiving is started at the ith slot in RX2, and the time moment of this slot is $t_0 + l_s\delta + t_1 + t_a + t_2 + (i-1)\delta$ (we use $t^{(2)}$ to simply this time moment). The analysis and derivation are similar to requirement 1) in RX1, and thus we do not repeat them here. As shown in Fig. 10, the longer gray contains this period. The probability $p^s_{r2\_1}$ is calculated as follows:

$$p^s_{cr2\_1} = \left[1 - P_7\left(t^{(2)} - l_7\delta + \delta \le t \le t^{(2)}\right)\right] \times \prod_{k=8}^{12}\left[1 - P_k\left(t^{(2)} - l_7\delta \le t \le t^{(2)}\right)\right]. \quad (11)$$

2) No node with $SF = x$ accesses the gateway in the time period $[t_0 + l_s\delta + t_1 + t_a + t_2 - l_x\delta + \delta + (i-1)\delta, t_0 + l_s\delta + t_1 + t_a + t_2 - l_{(x-1)}\delta + (i-1)\delta]$, and no node with $SF = x+1 \sim 12$ accesses the gateway in the time period $[t_0 + l_s\delta + t_1 + t_a + t_2 - l_x\delta + (i-1)\delta, t_0 + l_s\delta + t_1 + t_a + t_2 - l_{x-1}\delta + (i-1)\delta]$, where $x \in [8, 12]$. Requirement 2)'s analysis and derivation are also similar to RX1's. If there is an x such that $t^{(2)} - l_x\delta \le 0$, let $x_{\max} = x$, if does not exist, then let $x_{\max} = 12$. As shown in Fig. 10, the longer gray is this period. The probability $p^s_{r2\_2}$ is

$$p^s_{cr2\_2} = \prod_{x=8}^{x_{\max}}\left[1 - P_x\left(t^{(2)} - l_x\delta + \delta \le t \le t^{(2)} - l_{x-1}\delta\right)\right] \times \prod_{m=x}^{12}\left[1 - P_m\left(t^{(2)} - l_x\delta \le t \le t^{(2)} - l_{x-1}\delta\right)\right]. \quad (12)$$

3) When $i \ge 2$, there are at least one type of SF node accessing gateway for data transmission in the time period $[t_0 + l_s\delta + t_1 + t_a + t_2 - l_x\delta + (i-2)\delta, t_0 + l_s\delta + t_1 + t_a + t_2 - l_x\delta + (i-1)\delta]$, the green rectangle in Fig. 10 explains this case. If x exists such that $t_0 + l_s\delta + t_1 + t_a + t_2 - l_x\delta + (i-2)\delta \le 0$, let $x_{\max\_2} = x$, if not, let $x_{\max\_2} = 12$. The probability $p^s_{r2\_3}$ is

$$p^s_{cr2\_3} = 1 - \prod_{x=7}^{x_{\max\_2}}\left[1 - P_x\left(t^{(2)} - l_x\delta - \delta \le t \le t^{(2)} - l_x\delta\right)\right]. \quad (13)$$

In addition, the node receives reply in RX2, indicating that RX1 has been occupied by uplink transmissions. We now analyze how to calculate the probability that RX1 is occupied. Assuming j is the jth slot in RX1, $j \in [1, (t_a/\delta) - \gamma_s + 1]$, and the ACK is started to receive at the ith slot in RX2. According to requirement 2), the minimum time moment at which the node is not allowed to access the gateway is $t^{(2)} - l_{x_{\max}}\delta$, we compare this time moment with RX1's each time moment $t_0 + l_s\delta + t_1 + (j-1)\delta$ (this time moment is expressed by $t^{(3)}$) that do not allow nodes to access gateway, where $y \in [7, 12]$. Here, we introduce function $f(y, j)$

$$f(y, j) = t^{(2)} - l_{x_{\max}}\delta - \left[t^{(3)} - l_y\delta\right]. \quad (14)$$

For each j, if there exist minimal y_j such that $f(y_j, j) > 0$, y_j is the minimal SF that its uplink transmission is forbidden at the jth slot in RX1, and then y_j is added to set Y

$$Y = \left\{y_1, y_2, y_3, \ldots, \ldots, y_{\frac{t_a}{\delta} - \gamma_s + 1}\right\}. \quad (15)$$

The main goal of this part is to select the time period and SF such that these nodes' uplink transmissions interrupt the downlink transmissions in RX1. We have selected

the nodes' *SF* and the minimal time moment, then we should define the maximum time moment. For each y_j, if $f(y_j-1,j)<0$, let $t_{\max}=t^{(2)}-l_{x_{\max}}\delta$, or else $t_{\max}=t_0+l_s\delta+t_1-l_{y-1}\delta+(j-1)\delta$.

4) If RX1 is occupied, there is at least one $SF=y$ node accessing the gateway in the time period $[t^{(3)}-l_y\delta, t_{\max}]$, where $y\in[y_j,12]$, as indicated by the shorter yellow rectangle in Fig. 10. The probability $p^s_{r2\_4}$ is

$$p^s_{r2\_4}=\prod_{j=1}^{\frac{t_a}{\delta}-\gamma_s+1}\left[1-\prod_{y=y_j}^{12}\left[1-P_y\left(t^{(3)}-l_y\delta\le t\le t_{\max}\right)\right]\right]. \tag{16}$$

Finally, the probability that node receives the gateway's reply in the second RX window is p^s_{r2}

$$p^s_{r2}=\left\{p^s_{r2\_1}*p^s_{r2\_2}*p^s_{r2\_4}|i=1\right\}+\sum_{i=2}^{\frac{t_a}{\delta}-\gamma_s+1}p^s_{r2\_1}*p^s_{r2\_2}*p^s_{r2\_3}*p^s_{r2\_4}. \tag{17}$$

Therefore, when a node with $SF=s$ accesses the network, the probability that the sender receives the reply from the gateway is

$$p^s_r=p^s_{r1}+p^s_{r2}. \tag{18}$$

C. Concurrent Transmission Regression Model

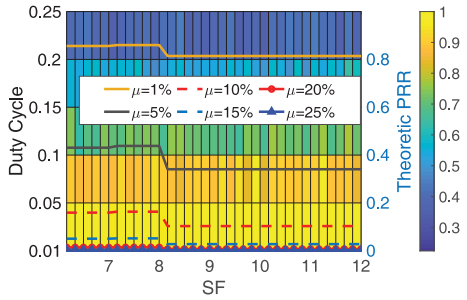
In this section, we compare the experimental results with that derived from our theoretical model in Section IV-B and analyze their relationship.

Fig. 11(a) depicts the comparison of experimental and theoretical downlink PRRs with different duty cycles and *SF*s. The color block graph shows the experimental results, and its duty cycles are marked in the left *y*-axis, different colors represent different PRRs. The theoretic results are plotted with lines, which represent different duty cycles, and values of the right *y*-axis reflect PRR results, and the *x*-axis is *SF* that is used for both graphs.

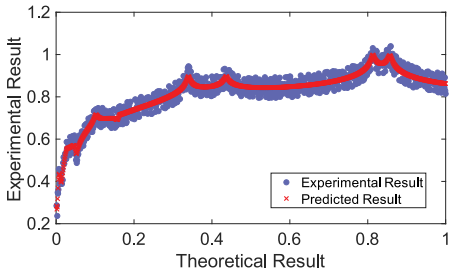
From Fig. 11(a), we observe that although the trends of the PRRs in both graphs with different duty cycles and *SF*s are similar, there are still differences between experimental and theoretical results. For example, the experimental PRR of $SF=7$ with 20% duty cycle is about 0.3, but the theoretic PRR is only about 0.04. This is partially due to the fact that the theoretical model calculates the PRR with strict and accurate time, and this is impossible in real experiments for the node's time offset may vary by a crystal oscillator. These time offsets can avoid collisions between uplink and downlink randomly, and resulting in the difference between experimental and theoretical results.

In order to obtain more realistic results, we use a regression learner to formally characterize the accuracy of the model, we try 19 models in five categories and select the best results of each category shown in Table III

From the comparison of these results, we observe that the medium Tree is the best one for the *R*-squared is 0.99 and other metrics are all minimum. Therefore, in this article, we use the medium Tree to fine tune our theoretical model, and Fig. 11(b) shows the predicted result using this regression and theoretical model.



(a)



(b)

Fig. 11. Experimental results versus theoretic results. (a) Comparison of experimental and theoretic results. (b) Results of regression.

TABLE III
RESULTS OF DIFFERENT REGRESSION MODEL

Model	RMSE	R-Squared	MSE	MAE
Interactions linear	0.13295	0.73	0.017677	0.11901
Medium Tree	0.030313	0.99	0.0009189	0.022814
Fine Gaussian SVM	0.047308	0.97	0.002238	0.038415
Rational quadratic GPR	0.031105	0.99	0.00096754	0.023447
Bagged Trees	0.30566	0.99	0.00093425	0.023147

D. Concurrent Transmission Parameters Optimization

We have analyzed the states of gateway and nodes in Section II, and the RX window size is considered the main reason that affects the reception of downlink transmissions, for the small length of the RX window makes the downlink PRR lower. Therefore, the simple way to improve the downlink PRR is to increase RX window size (i.e., t_a). If t_a is long enough, the ACK can be received sooner or later that do not consider the loss results from other interferences. However, the energy consumption would exceed the budget, and the transmission of new data also be affected. Fig. 12 shows that with an increase of t_a, all kinds of *SF* nodes' PRR and goodput are increased, and for a large *SF* node, the average energy is also decreased. But for a small *SF* node, the average energy is not decreased, even grows with an increase of t_a, because the energy consumption of retransmission is less than the energy caused by the increase of t_a.

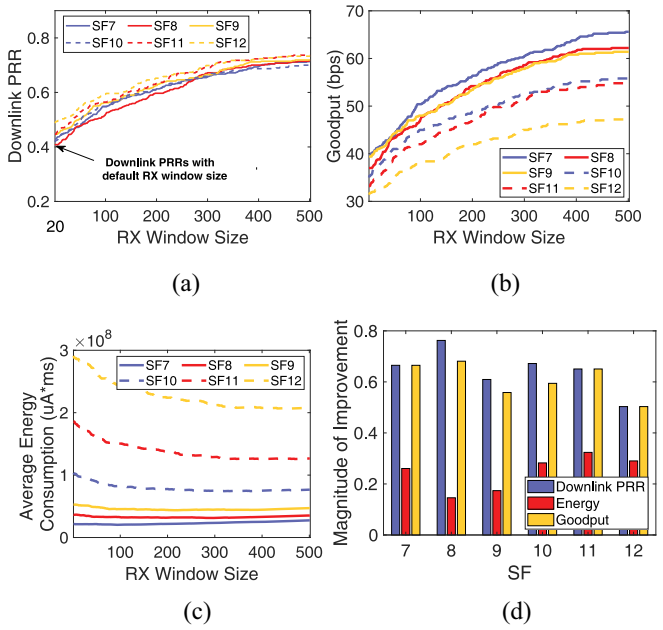


Fig. 12. Network performance with different RX window size. (a) Downlink PRR. (b) Goodput. (c) Average energy consumption. (d) Improvement.

Considering the constraints in objective function (2), we propose a simple method WW to select an optimized t_a and t_2. *WW* decreases t_2 and increase t_a at the same time (t_1 should be retained enough for the gateway to process data) to search a proper t_a that minimizes objective function (2) without increasing energy consumption.

So far, the three major components of Cantor have been proposed, and the pseudoalgorithm of Cantor is shown in Algorithm 1 in order to describe how Cantor works with these components to solve the problem in Section III.

Algorithm 1 displays a summary of the Cantor, which consists of three components. The gateway would first gather relevant parameters from nodes in input lines, and then, calculate the theoretical downlink PRRs (line 2). Afterward, the regression model has been established through the combination of theoretical and statistic downlink PRRs, and whenever a parameter changes, the gateway can calculate the updated downlink PRRs directly, instead of recounting from nodes.

In lines 4 and 5, the gateway would first calculate the initialized expected transmission times $\mathbb{E}(Rt)$ and consumed energy $ET_{sf}^{(t_1,t_2,t_a)}$. Then with *WW*, gateway iterates through each t_2 and t_a, and updates the current $ET_{sf}^{(t_1,t_2,t_a)}$ and $\mathbb{E}(Rt)$. More specifically, line 9 deals with the theoretical downlink PRRs updating, followed by the change of parameters in lines 7 and 8. Line 10 handles the realistic downlink PRRs updating process. Line 11 indicates the new $ET_{sf}^{(t_1,t_2,t_a)}$ and $\mathbb{E}(Rt)$, in which the optimized t_2 and t_a would be reached if the results no longer satisfy the constraints.

Besides, we discuss the computational complexity of Cantor in this part. In the initial stage, the gateway should collect parameters from nodes, whereas the most time-consuming operations reside in the calculation of optimized t_2 and t_a, we thus mainly focus on the analysis of this part. As noted in the updating process, each calculation will iterate through each changed parameters, and change times on these parameters could be approximated by $t_2/2i$, since the value of is a constant and thus can be ignored, the complexity of iteration is $O(t_2)$. Because the computational complexity of (10) and (16) is $O(t_a)$, 16 is $O(t_a^2)$, other equations within (6)–(17) are constant number, therefore, the computational complexity of (18) is $O(t_a^2)$. The computational complexity of Cantor is $O(t_2 \times t_a^2)$.

E. Analysis: Upper Bound of Goodput

The goodput of concurrent transmission is determined by the downlink PRR, which is influenced by whether nodes receive downlink transmission in RX windows. When the RX window size (i.e., t_a) is longer, the downlink PRR is higher as well. Since the limitation of energy and active time that should obey the duty cycle, t_a cannot be extended indefinitely, Cantor is likely to select a proper t_a that obtains the optimal goodput while satisfying the constraints. Note that in this article, the maximum length of t_a is $2t_a + t_2$, and t_2 becomes 0, t_1 should be left for the gateway to process data.

Assuming there are n nodes accessing the gateway during time T. The packet length of node i is l_i, and the packet air time is $l_i\delta$. Therefore, the total time that the gateway in *RX* mode is: $T_{RX} \leq \sum_{i=1}^{n} l_i\delta$, and hence the total time of *TX* mode is: $T_{TX} \geq 1 - T_{RX}$. In order to acknowledge these nodes, at least $T_{\text{ack}} = n\gamma\delta$ should be left. The throughput is

$$G_{\text{throughput}} = \frac{\sum_{i=1}^{n} l_i\delta}{T}. \tag{19}$$

We obtain a percentage $p_{\max}$ that this proportion *RX* durations are less than $2t_a + t_2$ through the CDF of gateway's *RX* durations. Therefore, as noted in Section II-C, if $T_{\text{ack}}/T_{TX} < 1$, it implies that the gateway's *TX* time can afford the downlink transmissions, and combining $p_{\max}$, we can derive that $p_{\max}$ of the throughput is goodput.

But if $T_{\text{ack}}/T_{TX} > 1$, it means that only $T_{TX}/\gamma\delta$ nodes may receive the downlink transmissions. Considering the limitation of gateway's *RX* duration, the upper bound of the goodput without considering energy consumption is

$$G_{\text{goodput}}^{\text{upper}} = \begin{cases} p_{\max} G_{\text{throughput}} & T_{\text{ack}}/T_{TX} \leq 1 \\ \dfrac{p_{\max} \sum_{i=1}^{T_{TX}/\gamma\delta} l_i\delta}{T} & T_{\text{ack}}/T_{TX} > 1. \end{cases} \tag{20}$$

V. Evaluation

Previously, we have shown the performance of concurrent transmission and verified the effectiveness of our method. In this section, we summarize and analyze the performance of Cantor.

A. Experimental Setup

The experiments are conducted both indoor and outdoor environments: an indoor 7.5 m × 6.6 m lab, an indoor 18 m × 15 m office, an outdoor playground, and an outdoor open area near our department. Our network consists of one LoRa gateway and 48 LoRa nodes. These nodes are divided into six groups according to their SFs, and each group has eight nodes in different channels. All nodes send data to the gateway

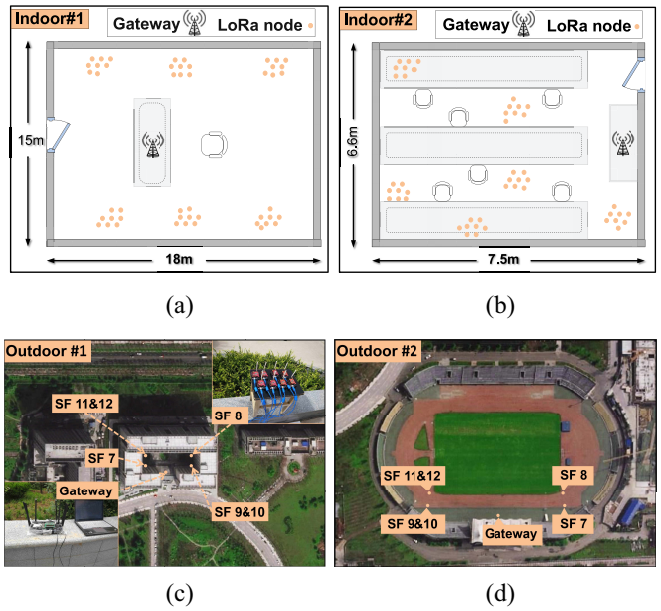


Fig. 13. Experiments's deployment in four scenarios. (a) Empty office. (b) Lab. (c) Open area. (d) Playground.

concurrently with a certain duty cycle, both the gateway and nodes operate at 915 MHz with a bandwidth of 125 kHz.

We benchmark Cantor's performance using three metrics: 1) goodput; 2) downlink PRR; and 3) average energy consumption.

B. Impact of Indoor and Outdoor Deployment

In our indoor deployment, the LoRa nodes are placed in the room while the gateway operates in the center of the room. Indoor #1 [shown in Fig. 13(a)] is an empty office and indoor #2 [shown in Fig. 13(b)] is a lab with about 30 students.

In outdoor scenarios, we place the gateway in a playground and an open area near our department, and nodes are placed around the gateway. Outdoor #1 [as shown in Fig. 13(c)] is the open area and outdoor #2 [as shown in Fig. 13(c)] is the playground.

In these four scenarios, nodes operate with a duty cycle of 20%, each experiment lasts at least 2 h. Fig. 14 shows the result of aggregated throughput and goodput in the four scenarios with different RX window sizes. At 500 ms, the goodput achieves around 350 bps, and the throughput is very stable with the increase of the RX window. We further evaluate the aggregated throughput in different scenarios, around 470 bps in the empty office and 425 bps in the playground, both the lab and the open area have a throughout of 450 bps.

Fig. 15 shows the comparison of the concurrent transmission performance of Cantor in the four scenarios and without Cantor. The gray bars indicate the performance without Cantor and the colorful bars indicate the performance of Cantor in different scenarios. Overall, for all kinds of nodes, the downlink PRRs are improved in all scenarios, and the magnitudes of improvement are about 60%, as shown in Fig. 16, and the goodput is improved correspond to the downlink PRR. But the average energy consumption is reduced slightly, especially for

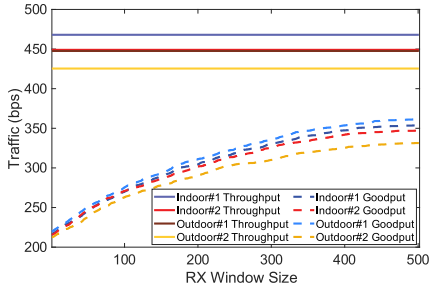


Fig. 14. Throughput versus goodput in different scenarios.

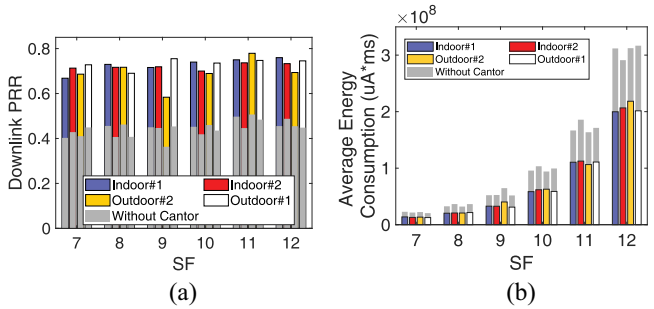


Fig. 15. Network performance in different scenarios. (a) Downlink PRR. (b) Average energy consumption.

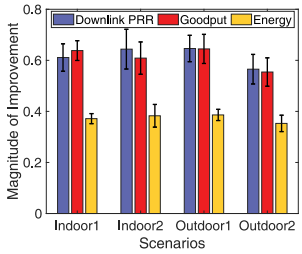


Fig. 16. Improvement in different scenarios.

small *SF* nodes. The reason is that increasing the RX window size will consume more energy, but on the other hand, it will decrease the energy consumption of retransmissions. Thus, the improvement in energy consumption is smaller than that in goodput. For small *SF* nodes, the energy is decreased slightly because their retransmission energy consumptions are smaller than large *SF* nodes'.

In conclusion, Cantor improves the performance of the concurrent transmission in both indoor and outdoor scenarios. The results show the goodput, downlink PRR, and energy consumption improvements of 0.56–0.65×, 0.57–0.65×, and 0.38×.

C. Impact of Duty Cycle

We now evaluate the network performance with different duty cycles (5%–25%) and show the results in Figs. 17 and 18. Overall, we can observe that all downlink PRR, goodput, and average energy consumption are improved with the duty cycle increased. The downlink PRR and average energy consumption without Cantor decrease along with the duty cycle increased, but the goodput increases first and declines then, as shown in Fig. 18(b). The major reason is that the higher

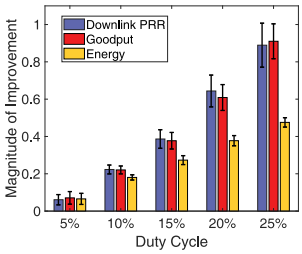


Fig. 17. Improvement with different duty cycles.

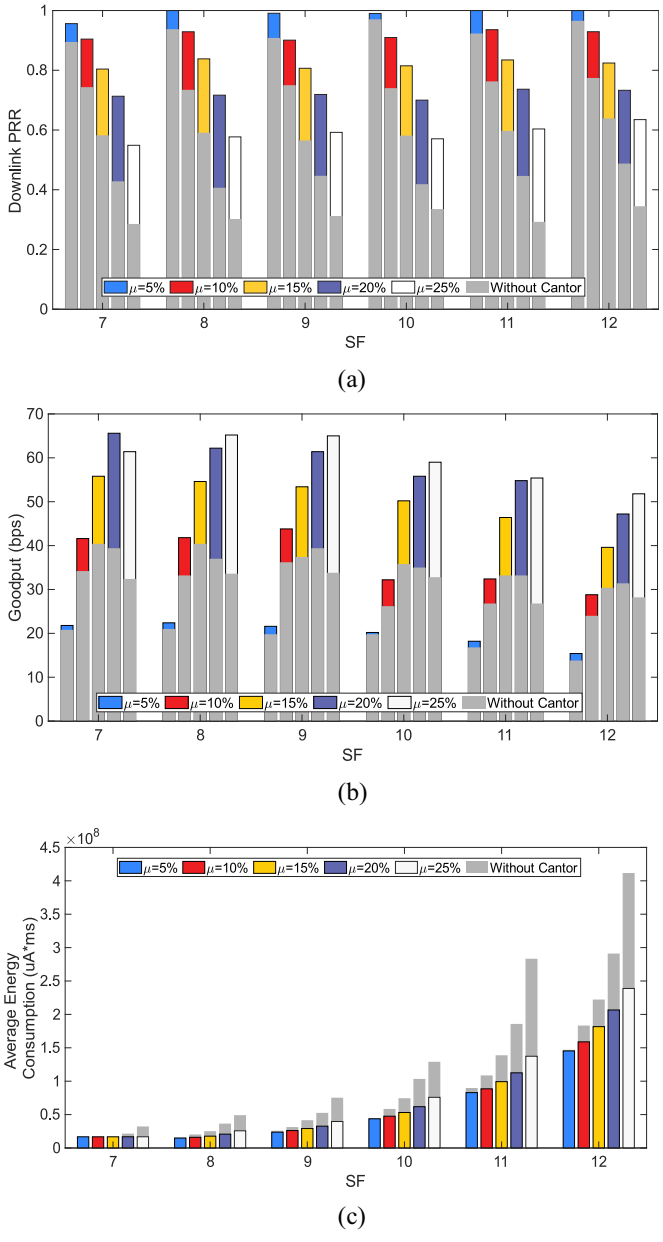


Fig. 18. Network performance with different duty cycles. (a) Downlink PRR. (b) Goodput. (c) Average energy consumption.

duty cycle indicates more packets are transmitted to the gateway at the same time, and hence, the goodput is increased. When the duty cycle is increased to 20%, the goodput without Cantor reaches the peak, and the downlink PRR without

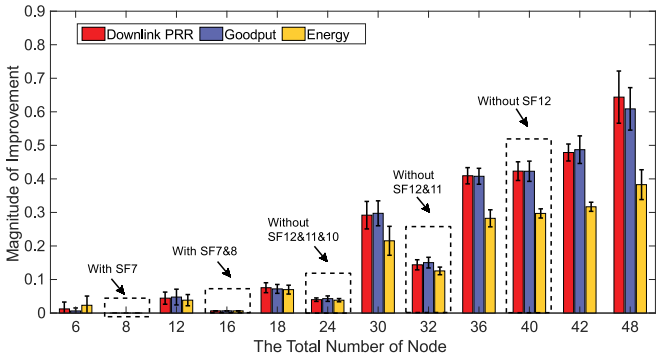


Fig. 19. Improvement with different total number of nodes.

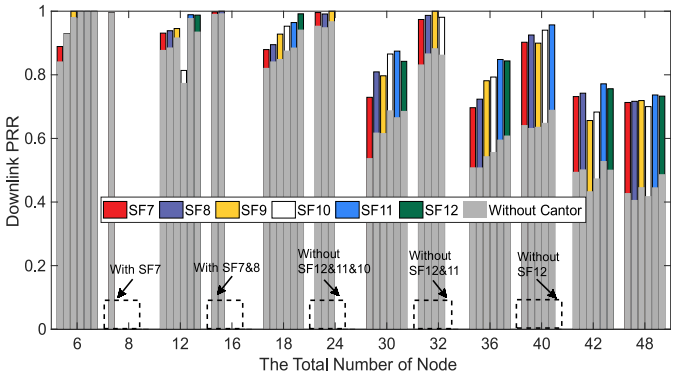


Fig. 20. Downlink PRR with different total number of nodes.

Cantor is decreasing as shown in Fig. 18(a), it means that more packets need to be retransmitted. When the duty cycle increases to 25%, the retransmitted packets are more than received, therefore, the goodput decreases.

Fig. 18 also reveals that different SFs nodes' improvements of downlink PRR, goodput, and energy in a certain duty cycle are similar, although the energy of different nodes as shown in Fig. 18(c) is increasing, the improvement is also stable, and the error bar of different SFs nodes in Fig. 17 supports this result. These results show that Cantor is effective on nodes with different SFs and different duty cycles.

D. Impact of the Network Size

In this section, we test how network size affects the performance of Cantor. We conduct 12 experiments with a network size of 6–48 nodes to measure the performance, and Fig. 19 shows the magnitude of improvement and standard deviation for each kind of *SF* nodes. The experiments with a network size of 8, 16, 24, 32, and 40 are conducted with the absence of at least one kind of *SF* nodes, e.g., 40 nodes transmit data scanning eight channels but without nodes of $SF = 12$. Therefore, it is obvious that the overall improvement of these experiments is lower than that with all SFs, and the standard deviation is also smaller. Cantor performs poorly in a network without large *SF* node is not the major reason, it is because without large *SF* nodes, the downlink PRR and goodput are better than that with large *SF* nodes, as shown in Figs. 20 and 21. This results first reveal that large *SF* nodes may cause the uplink and downlink blocking problem, and

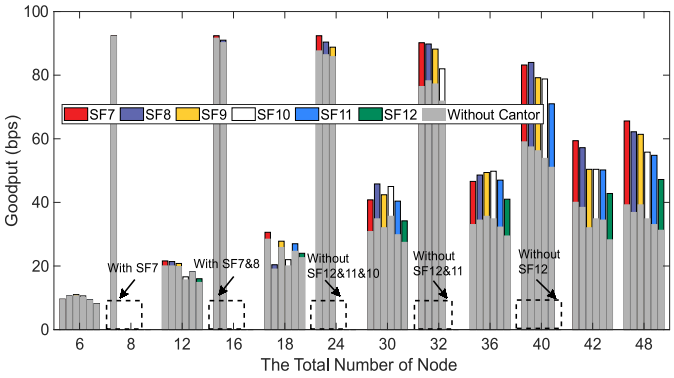


Fig. 21. Goodput with different total number of nodes.

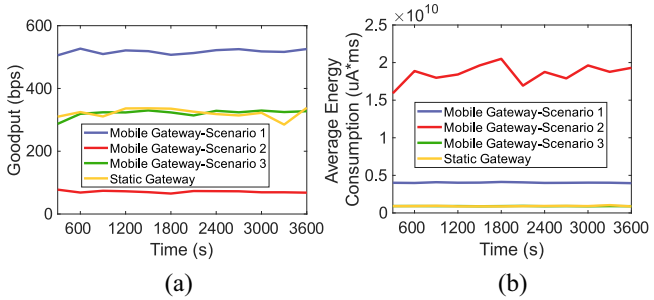


Fig. 22. Comparison of different mobile scenarios with Cantor. (a) Goodput. (b) Average energy consumption.

hence decreasing the goodput. Second, Cantor has a significant effect on large *SF* nodes, for the standard deviations are larger than that without large *SF* nodes.

E. Impact of Mobility

In this section, we test how node mobility affects the performance of Cantor. This experiment is conducted in an outdoor yard with 48 nodes and a gateway, the basic experimental setup is the same as in Section V-A. The only difference is that the gateway moves at the speed of 0.5 m/s in this experiment. According to the distances between gateway and nodes, the experimental scenarios can be divided into three categories: all nodes' *SF* are 7 is mobile gateway-scenario 1, for the gateway is close to all nodes, and nodes are able to send data to the gateway with $SF = 7$; another is mobile gateway-scenario 2 while all nodes' *SF* are 12, for the gateway is far away from all nodes, and nodes have to increase the *SF* in order to communicate with gateway; in the last scenario mobile gateway-scenario 3, the gateway moves randomly around nodes so that the number of nodes with different *SF*s remains unchanged in the network. The results of these three scenarios and a static gateway are shown in Fig. 22. It is obvious that the performance of mobile gateway-scenario 3 is similar to the static gateway in both goodput and average energy consumption. Besides, mobile gateway-scenario 1 outperforms other scenarios in goodput and static gateway performs better than mobile gateway-scenario 1 in energy.

The main reason is *SF*. As specified in LoRaWAN, node adjusts *SF* through adaptive data rate (ADR) to ensure the transmission quality at different communication distances, the smaller *SF*, the smaller packet size, and the faster transmission rate. In mobile gateway-scenario 1, the time that each node occupies the channel is shorter than that in mobile gateway-scenarios 2 and 3, and the amount of data per unit time is also more than others, therefore the goodput is best. But on the other hand, all nodes transmit data with the same *SF* increase the probability of collision, and thus, some nodes have to back off and retransmit to avoid collision, resulting in the increase of average energy consumption.

Moreover, this experiment only includes the case where the gateway moves at a low speed. The main reason we do not consider the high speed is that the high-speed movement leads to unstable channel quality between node and gateway, and according to LoRaWAN: ADR control may not be possible when the radio channel attenuation changes fast and constantly. It means that the uplink and downlink transmission qualities between nodes and gateway will be affected. At this time, it is difficult to ensure the concurrent transmission, without concurrent transmission, Cantor is meaningless.

F. Comparison of Cantor and NAK

In this section, we compare Cantor with benchmark scheme NAK, the benchmark is a representative mechanism used to ensure data transmission in the wireless network. The experimental setup is the same as in Section V-E except for the mobile gateway. Fig. 23 depicts the results of the comparison, from Fig. 23(a) and (b), it is observed that NAK achieves better performance than Cantor in goodput and average energy consumption. This is because the gateway does not acknowledge all uplink transmissions in NAK, and nodes send new packets without receiving the response from the gateway. As a result, NAK's performance of goodput and energy consumption is better. Nonetheless, the goodput of NAK decreases with the increase of packet loss rate because the downlink transmissions from the gateway for requesting nodes to retransmit the lost packets also increase, and the uplink and downlink blocking problem occurs.

Although NAK's goodput and average energy consumption are better, it is unsuitable for LoRa concurrent transmission network, Fig. 23(c) depicts the main reason: the uplink PRR of NAK is lower than Cantor. This is because the retransmission request information sent by the gateway will be blocked by other uplink transmissions, resulting in the loss of uplink packets. Besides, NAK does not support the implementation of ADR in LoRa because ADR is based on the downlink response.

In summary, the proposed method Cantor is effective and is robust to different environments (indoor and outdoor), duty cycle variation (the largest is 25%), and network size variation (the network contains different number of SFs nodes). In addition, a mobile gateway in the network has little effect on Cantor. Although the benchmark method NAK is slightly better than cantor in terms of goodput and energy consumption, the uplink PRR is too large, and this mechanism is not suitable for adaptive rate adjustment of LoRa nodes.

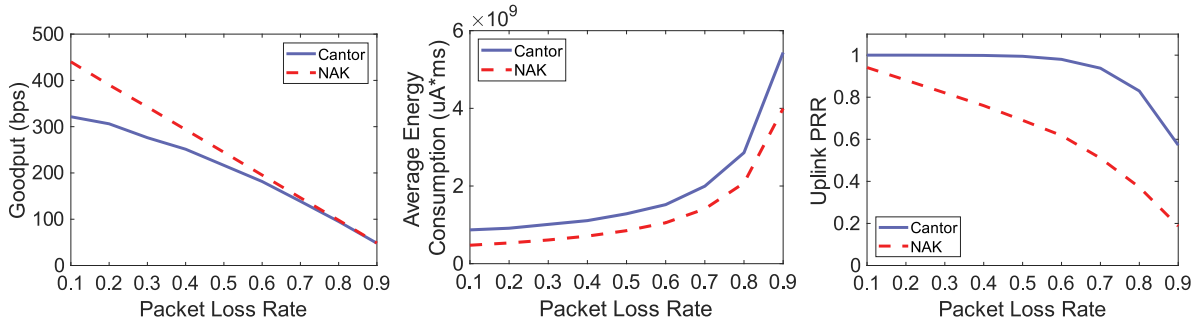


Fig. 23. Performance comparison of Cantor and benchmark NAK. (a) Goodput. (b) Average energy consumption. (c) Uplink PRR.

VI. Related Work

The existing concurrent transmission in LoRa can be broadly divided into the following three categories.

Multiple Channels and Orthogonal Concurrency: Comparing with other wireless networks, LoRa has the capability of concurrent transmission. To be more specific, this concurrency consists of two aspects. First, a standard LoRa gateway [16] demodulates simultaneous transmissions from multiple nodes in different channels. Second, LoRa can efficiently demodulate concurrent orthogonal transmissions even on the same channel, because LoRa modulation supports different spreading factors, which are orthogonal. Recent work [17] conducts a series of experiments to verify the performance of concurrent transmission, and the results are similar to the claims made by Semtech.

Single Channel and Same SF Concurrency: Besides, there have been efforts made to enable a concurrent transmission in the same channel with a certain *SF*. Choir [18] exploits hardware imperfection that produces offsets in time, frequency, and phase. These properties are leveraged to separate and decode collisions from 5 to 10 LoRa nodes at a software radio. This concurrent transmission system improves network throughput and expands communications range significantly.

LoRa Backscatter Concurrency: With the recent innovations in low-power backscatter communication [19], [20], LoRa backscatter [21], [22] becomes attractive because it achieves long transmission distance while keeps batteryfree. Long-range backscatter systems primarily work at the link layer and cannot deal with the number of tags, therefore, NetScatter [23] introduces a distributed chirp spread spectrum (CSS) coding, which combines the CSS and ON–OFF keying modulation. NetScatter supports hundreds to thousands of concurrent transmissions in the LoRa backscatter network.

We note that while prior works support concurrent transmission, little effort has been made to investigate the goodput in LoRa transmission [14], [24]. The study in [17] discovers that downlink traffics, especially ACK packets can block a significant amount of uplink traffics. The work in [14] suggests that LoRa supports uplink concurrency but no downlink concurrency, this imbalance blocks acknowledging each uplink packet. This article uses the NAK scheme to avoid acknowledging all concurrent uplink transmissions, i.e., the gateway only responds to a node if its packets are lost. However, the downlink responses may also be blocked by the uplink transmissions and the lost packets may have been flushed from the node's memory.

In this article, we conduct a deep analysis on the concurrent uplink and downlink transmissions, discover the window mismatch problem, and propose a congestion control scheme that improves the goodput and reduces energy consumption.

VII. Conclusion

In this article, we presented a novel control scheme to improve the goodput of LoRa concurrent transmission. Cantor constructs a concurrent transmission to explore the correlation between downlink PRR and transmission parameters (such as duty cycle, receive delay, and RX window size), and then leverages a regression model to derive the realistic downlink PRR with different network settings. Finally, Cantor selects the proper parameters that optimize the goodput for nodes. The experimental results show that Cantor is robust in various scenarios and able to improve the goodput and decrease energy consumption. In our future work, we would like to explore the availability of Cantor's theoretical model in the designing of other strategies.

References

[1] R. Sanchez-Iborra and M. D. Cano, "State of the art in LP-WAN solutions for industrial IoT services," *Sensors*, vol. 16, no. 5, pp. 708–722, 2016.

[2] T. E. T. K. N. Sornin, M. Luis, and O. Hersent, *LoRaWAN Specifications*, LoRa Alliance, San Ramon, CA, USA, 2015.

[3] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of LoRa: Long range low power networks for the Internet of Things," *Sensors*, vol. 16, no. 9, pp. 1466–1484, 2016.

[4] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, and T. Watteyne, "Understanding the limits of LoRaWAN," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 34–40, Sep. 2017.

[5] E. D. Ayele, "Performance analysis of LoRa radio for an indoor IoT applications," in *Proc. Int. Conf. Internet Things Global Commun.*, Funchal, Portugal, 2017, pp. 1–8.

[6] O. Khutsoane, B. Isong, and A. M. Abu-Mahfouz, "IoT devices and applications based on LoRa/LoRaWAN," in *Proc. IEEE 43rd Annu. Conf. Ind. Electron. Soc. (IECON)*, Beijing, China, Nov. 2017, pp. 6107–6112.

[7] A. Lavric and A. I. Petrariu, "LoRaWAN communication protocol: The new era of IoT," in *Proc. Int. Conf. Develop. Appl. Syst. (DAS)*, Suceava, Romania, 2018, pp. 74–77.

[8] L. Li, J. Ren, and Q. Zhu, "On the application of LoRa LPWAN technology in sailing monitoring system," in *Proc. 13th Annu. Conf. Wireless On-Demand Netw. Syst. Serv. (WONS)*, Jackson, WY, USA, Feb. 2017, pp. 77–80.

[9] P. Neumann, J. Montavont, and T. Noël, "Indoor deployment of low-power wide area networks (LPWAN): A LoRaWAN case study," in *Proc. 12th IEEE Int. Conf. Wireless Mobile Comput. Netw. Commun. (WiMob)*, New York, NY, USA, Oct. 2016, pp. 1–8.
[10] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," 2016. [Online]. Available: https://arxiv.org/abs/1606.07360.
[11] A. A. Boulogeorgos, P. D. Diamantoulakis, and G. K. Karagiannidis, "Low power wide area networks (LPWANs) for Internet of Things (IoT) applications: Research challenges and future trends," 2016. [Online]. Available: https://arxiv.org/abs/1611.07449.
[12] W. Michal, K. Pawel, M. Katarzyna, and K. Bogdan, "Time-aware monitoring of overhead transmission line sag and temperature with LoRa communication," *Energies*, vol. 12, no. 3, pp. 1–23, 2019.
[13] *LoRa: DNA of IoT*, SMETECH, Camarillo, CA, USA, 2020. [Online]. Available: https://www.semtech.com/lora
[14] C. Gu, R. Tan, X. Lou, and D. Niyato, "One-hop out-of-band control planes for low-power multi-hop wireless networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Honolulu, HI, USA, Apr. 2018, pp. 1187–1195.
[15] *SX1276-7-8-9 Datasheet*, SMETECH, Camarillo, CA, USA, May 2013. [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1276
[16] *SX1301 Datasheet*, SMETECH, Camarillo, CA, USA, Jun. 2014. [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-gateways/sx1301
[17] J. C. Liando, A. Gamage, A. W. Tengourtius, and M. Li, "Known and unknown facts of LoRa: Experiences from a large-scale measurement study," *ACM Trans. Sens. Netw.*, vol. 15, no. 2, pp. 1–35, 2019.
[18] R. Eletreby, D. Zhang, S. Kumar, and O. Yagan, "Empowering low-power wide area networks in urban settings," in *Proc. Conf. ACM Spec. Interest Group Data Commun. (SIGCOMM)*, Los Angeles, CA, USA, Aug. 2017, pp. 309–321.
[19] P. Hu, P. Zhang, and D. Ganesan, "Laissez-faire: Fully asymmetric backscatter communication," in *Proc. ACM Conf. Spec. Interest Group Data Commun. (SIGCOMM)*, London, U.K., Aug. 2015, pp. 255–267.
[20] M. Jin, Y. He, X. Meng, Y. Zheng, D. Fang, and X. Chen, "FlipTracer: Practical parallel decoding for backscatter communication," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 330–343, Feb. 2019.
[21] Y. Peng *et al.*, "PLoRa: A passive long-range data network from ambient LoRa transmissions," in *Proc. Conf. ACM Spec. Interest Group Data Commun. (SIGCOMM)*, Budapest, Hungary, Aug. 2018, pp. 147–160.
[22] V. Talla, M. Hessar, B. Kellogg, A. Najafi, J. R. Smith, and S. Gollakota, "LoRa backscatter: Enabling the vision of ubiquitous connectivity," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 1, no. 3, pp. 1–24, 2017.
[23] M. Hessar, A. Najafi, and S. Gollakota, "NetScatter: Enabling large-scale backscatter networks," in *Proc. 16th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Boston, MA, USA, Feb. 2019, pp. 271–284.
[24] J. Robert and A. Heuberger, "LPWAN downlink using broadcast transmitters," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Cagliari, Italy, Jun. 2017, pp. 1–5.

Dan Xu received the B.S. degree in software engineering and the M.S. degree from Northwest University, Xi'an, China, in 2011 and 2014, respectively, where she is currently pursuing the Ph.D. degree with the School of Information Science and Technology.

Her current research interests include data transmission in wireless networks, LoRa signal processing, and neural networks.

Xiaojiang Chen (Member, IEEE) received the Ph.D. degree in computer software and theory from Northwest University, Xi'an, China, in 2010.

He is currently a Professor with the School of Information Science and Technology, Northwest University. His current research interests include localization and performance issues in wireless *ad hoc*, mesh, and sensor networks and named data networks.

Nannan Zhang was born in 1996. He is currently pursuing the M.S. degree with Northwest University, Xi'an, China.

His current research interests include time synchronization in LoRa network.

Nana Ding was born in 1995. She is currently pursuing the M.S. degree with Northwest University, Xi'an, China.

Her current research interests include wireless signal processing, LoRa, and neural networks.

Jing Zhang was born in 1996. She received the bachelor's degree in software engineering from Northwest University, Xi'an, China, in 2017, where she is currently pursuing the master's degree in computer application technology.

Her current research interests include wireless signal analysis, wireless communication with LoRa signal, optimization of wireless networks, and backscatter technology.

Dingyi Fang (Member, IEEE) received the Ph.D. degree in computer application technology from Northwestern Polytechnical University, Xi'an, China, in 2001.

He is currently a Professor with the School of Information Science and Technology, Northwest University, Xi'an. His current research interests include mobile computing and distributed computing systems, network and information security, and wireless sensor networks.

Tao Gu (Senior Member, IEEE)received the bachelor's degree from the Huazhong University of Science and Technology, Wuhan, China, in 1990, the M.S. degree from Nanyang Technological University, Singapore, in 2001, and the Ph.D. degree in computer science from the National University of Singapore, Singapore, in 2005.

He is currently an Associate Professor in computer science with RMIT University, Melbourne, VIC, Australia. His current research interests include Internet of Things, ubiquitous computing, mobile computing, embedded AI, wireless sensor networks, and big data analytics.