Combating Packet Collisions Using Non-Stationary Signal Scaling in LPWANs

Shuai Tong, *Student Member, IEEE*, Jiliang Wang, *Senior Member, IEEE*, and Yunhao Liu, *Fellow, IEEE, ACM*

***Abstract*— LoRa, a representative Low-Power Wide Area Network (LPWAN) technology, has been shown as a promising platform to connect Internet of Things. Practical LoRa deployments, however, suffer from collisions, especially in dense networks and wide coverage areas expected by LoRa applications. Existing collision resolving approaches do not exploit the modulation properties of LoRa and thus cannot work well for low-SNR LoRa signals. We propose *NScale* to decompose concurrent transmissions by leveraging subtle inter-packet time offsets for low SNR LoRa collisions. NScale (1) translates subtle time offsets, which are vulnerable to noise, to robust frequency features, and (2) further amplifies the time offsets by non-stationary signal scaling, i.e., scaling the amplitude of a symbol differently at different positions. In practical implementation, we propose a noise resistant iterative symbol recovery method to combat symbol distortion in low SNR, and address frequency shifts incurred by CFO and packet time offsets in decoding. We propose optimized designs for diminishing the time costs of computation-intensive tasks and meeting the real-time requirements of LoRa collision resolving. We theoretically show that NScale introduces < 1.7 dB SNR loss compared with the original LoRa. We implement NScale on USRP N210 and evaluate its performance in both indoor and outdoor networks. NScale is implemented in software at the gateway and can work for COTS LoRa nodes without any modification. The evaluation results show that NScale improves the network throughput by 3.3× for low SNR collided signals compared with other state-of-the-art methods.**

***Index Terms*— Internet of Things, LPWAN, LoRa, parallel transmission, collision resolving.**

I. Introduction

As A promising technology for Low-Power Wide Area Networks (LPWANs), LoRa draws extensive interests from both academia and industry. Different from high-power and high-bitrate Wi-Fi or cellular, LoRa focuses on the field of low-power, low-cost, and long-range communications for connecting millions of Internet of Things (IoT) devices [1]. As one of the key communication technologies for IoTs, LoRa is widely used in various IoT applications such as environment monitoring [2], wild animal tracking [3], disaster rescue [4], warehouse management [5], etc.

Manuscript received May 11, 2021; revised October 25, 2021; accepted November 23, 2021; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Ioannidis. Date of publication December 10, 2021; date of current version June 16, 2022. This work was supported in part by NSFC under Grant 62172250 and Grant 61932013, and in part by the Tsinghua University Initiative Scientific Research Program. *(Corresponding author: Jiliang Wang.)*

The authors are with the School of Software, Tsinghua University, Beijing 100084, China (e-mail: tl19@mails.tsinghua.edu.cn; jiliangwang@tsinghua.edu.cn; yunhao@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TNET.2021.3131704

However, LoRa networks in practice suffer from packet collisions, especially when connecting a large number of devices, which is expected for most LoRa applications [6], [7]. Collisions adversely cause packet loss and throughput degradation, which also drain battery life and waste precious air time and spectrum. Moreover, for design simplicity and energy conservation, LoRa uses the star-of-stars network topology and adopts a simple MAC layer design (e.g., ALOHA based MAC protocol in LoRaWAN), which further exacerbates packet collisions in LoRa networks [8], [9].

Existing Approaches. The collision problem should be carefully addressed before applying LoRa as the main technique for connecting millions of IoT devices. Although there exist a large collection of collision resolving approaches, they cannot work well for low-SNR LoRa as they do not exploit LoRa's modulation properties. For example, mLoRa [10] applies time domain successive interference cancellation (SIC) to LoRa collisions. It starts with a collision-free chunk and iteratively reconstructs and extracts each recovered chirp for packet decoding. According to the experiment, mLoRa mainly works for signals with $SNR > 5$ dB. Choir [11] exploits the hardware imperfections of low-cost LoRa nodes to separate collided packets. FTrack [12] decodes multiple LoRa packets from a collision by calculating the instantaneous frequency continuity by short time spectrum analysis.

Fundamental limitations: Existing collision decoding approaches [10], [12] have limitations in decoding low-SNR LoRa signals (e.g., $SNR < 0$). They focus more on the time domain signal analysis and interference cancellation. But they do not consider the modulation features of LoRa which can concentrate energy in the frequency domain. As a result, those methods have a high SNR loss compared with the original LoRa decoding and cannot work for low-SNR LoRa signals.

Our Approach. To resolve collisions in low SNR LoRa signals, we present *NScale* to decode packets from collided LoRa signals. The heart of NScale is to leverage the subtle packet time offsets to disentangle collided packets. To resolve collisions of low SNR, NScale (1) translates packet time offsets, which are vulnerable to noise, to more robust frequency features, and (2) amplifies the time offsets by non-stationary signal scaling, i.e., scaling a symbol differently at different positions. NScale then leverages the frequency features after non-stationary scaling to decompose concurrent transmissions.

To see how NScale works, consider a simplified collision scenario in Fig. 1, where two packets - each with three chirp symbols - collide. The PHY layer of LoRa uses the Chirp Spread Spectrum (CSS) to modulate data bits into

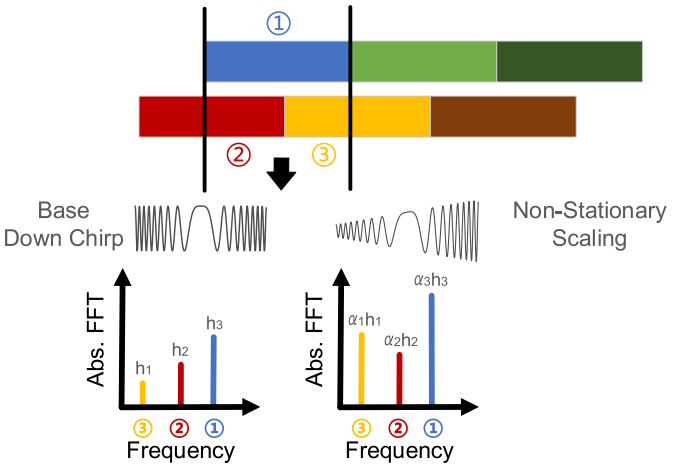


Fig. 1. NScale decoding example: for each collided symbol, NScale transforms and amplifies the time domain features to frequency features through non-stationary signal scaling. Chirp segments at different positions of the window are amplified with different peak scaling factors.

chirp symbols of linearly *increasing* frequencies. In demodulation, each received chirp symbol is multiplied with a base down-chirp of linearly *decreasing* frequency. When there is no collision, this dechirp operation results in a single frequency tone (a single peak after FFT) which represents the modulated data. When there are collisions, the dechirp results in multiple frequency tones in a demodulation window, making it difficult to distinguish symbols from different packets. To resolve the LoRa packet collisions, as shown in Fig. 1, we leverage the down-chirp with non-stationary amplitude to translate the time misalignment of packets into frequency features, which can be leveraged to correspond peaks to different transmitters. NScale first applies the standard demodulation with the base down-chirp to each window. The three chirp segments, as shown in Fig. 1, result in three FFT peaks (h_1, h_2 and h_3) with height proportional to the segment length and the signal amplitude. Those three peaks also show why traditional LoRa cannot decode the collision. Further, NScale dechirps signals in each window with a *non-stationary* scaled down-chirp, i.e., a down-chirp with varying amplification along time. This results in FFT peaks amplified with different factors ($\alpha_1 h_1$, $\alpha_2 h_2$ and $\alpha_3 h_3$), depending on which part of the non-stationary scaled down-chirp is multiplied with the corresponding chirp segment. Combining these two demodulation results, we can obtain the scaling factors α_i. By carefully designing the non-stationary scaled down-chirp, we can make α_i distinguishable for different chirp segments and derive the segment length and position related to the non-stationary scaled down-chirp. Based on this, we can group chirp segments for each packet with the same misalignment and then decode each packet.

Challenges. Turning the idea into reality, however, entails non-trivial challenges. First, NScale relies on accurate measurement of peaks for frequency tones after dechirping, which is difficult due to the low SNR of LoRa transmissions and the phase rotation property of the Fourier transform. We propose a noise resistant iterative peak recovery algorithm to combat the peak distortion, and achieve an accurate estimation of both the frequency and height for each peak. Second, it is non-trivial to design the non-stationary scaled down-chirp, which affects the performance of NScale. We make an in-depth analysis of the relationship between non-stationary scaled down-chirps and the decoding performance, and propose the designing strategy for non-stationary scaled down-chirps to optimize NScale's decoding performance in practice. Third, after grouping chirp segments to packets for decoding, we have to resolve the mixed impact of carrier frequency offsets (CFO) and symbol-to-window time offsets. We design a technique to calculate CFO and symbol-to-window time offset based on the combination of up-chirps and down-chirps in the preamble and SFD of LoRa packets. The last challenge arises from the practical requirements of decoding LoRa collisions in real-time. The computation overhead of NScale mainly stems from the in-window distribution detection and the packet identification. For detecting the segment distribution, the receiver estimates FFT peaks from both the standard demodulation and the non-stationary scaled demodulation, which is computation-intensive. To simplify the distribution detection, we propose an optimized peak scaling factor extraction method based on the fact that FFT peaks with and without non-stationary scaling locate at the same frequency. Besides, we optimize the packet identification by using the frequency-time relationship of chirps in LoRa preambles. We design the moving correlation windows to correlate received chirps with adjacent preamble symbols, and thus avoid the computation-intensive sample-step correlation with the standard preamble.

Main Results and Contributions.

- We propose NScale, a protocol leveraging non-stationary scaling to decompose concurrent transmissions for low SNR LoRa collisions, trying to bridge the gap between LoRa vision on providing low-power long-distance connection and its practical limitations. To address practical challenges in NScale design, we propose a noise-resistant iterative peak recovery algorithm to resolve peak distortion in low SNR LoRa signal, and remove the impact of the CFO and time offset to accurately decode packets. We optimize the time costs of computation-intensive tasks for meeting the real-time requirements of LoRa decoding.
- We theoretically analyze NScale performance and show that NScale incurs SNR loss $<$**1.7 dB** to original LoRa.
- We implement NScale on the SDR platform USRP N210. NScale is completely implemented in software at the LoRa gateway without any modification to LoRa end nodes. Thus, it can be easily applied to COTS LoRa nodes and existing LoRa networks.
- We thoroughly evaluate NScale's performance in both indoor and outdoor LoRa networks. The experiment results show that NScale can improve the network throughput in collisions by **3.3**$\times$ for low SNR LoRa signals compared with other state-of-the-art methods.

II. Background and Motivation

A. LoRa Background

LoRa physical layer adopts the Chirp Spreading Spectrum (CSS) technique for modulation [13]. CSS modulates

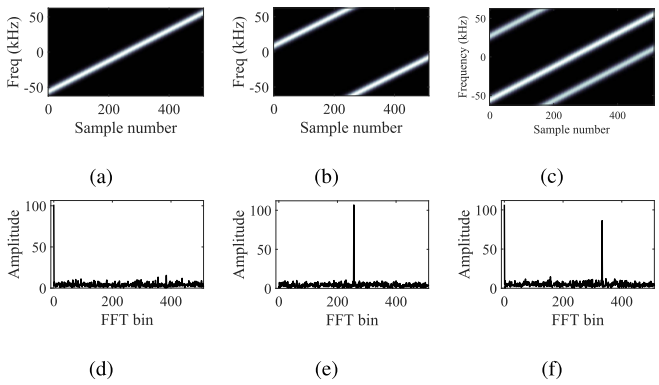


Fig. 2. Spectrogram and demodulation result (a)(d) a base chirp symbol, (b)(e) a shifted chirp symbol, and (c)(f) two LoRa chirp collision.

data into chirp symbols whose frequency change linearly over time. As shown in Fig. 2(a), given a predefined bandwidth BW, the frequency of the base up-chirp $C(t)$ linearly increases from $-\frac{BW}{2}$ to $\frac{BW}{2}$. Thus, the frequency of the base up-chirp can be represented as $kt - \frac{BW}{2}$, where k denotes the frequency increasing rate of the chirp. The phase of the up-chirp $\phi(t)$ will be

$$\phi(t) = 2\pi \int_0^t (k\tau - BW/2)d\tau \tag{1}$$

and thus base up-chirp $C(t)$ can be represented as

$$C(t) = e^{j\phi(t)} = e^{j2\pi(-\frac{BW}{2}t + \frac{k}{2}t^2)} \tag{2}$$

LoRa encodes data bits into symbols by shifting the initial frequency of the base up-chirp. A LoRa symbol with initial frequency f_{sym} is denoted as $C(t)e^{j2\pi f_{sym}t}$. Given the bandwidth BW, frequency for a symbol higher than $\frac{BW}{2}$ aligns down to $-\frac{BW}{2}$ as shown in Fig. 2(b). LoRa defines N different shifted initial frequencies, which results in N uniformly shaped up-chirps to encode $SF = log_2 N$ bits [14].

A typical LoRa receiver decodes a LoRa chirp by first multiplying it with a standard down-chirp, i.e., $C^{-1}(t)$ whose frequency decreases linearly over time. The multiplication leads to a single tone signal with the frequency of f_{sym}. Then the receiver applies the Fast Fourier Transform (FFT) on the dechirped signal, translating the signal into an energy peak in the frequency domain, as shown in Fig. 2(d) and (e). We finally extract the encoded data from the chirp symbol by identifying the index of the FFT peak in the frequency domain.

When multiple LoRa nodes transmit simultaneously, their signals collide at the receiver. Multiple overlapped chirps are transformed to multiple FFT peaks in the frequency domain through the LoRa decoding processes, as shown in Fig. 2(c) and (f). The LoRa demodulator cannot map FFT peaks to the correct transmitters, and thus it fails to decode the collided signals. The goal of this work is to decode LoRa collisions by corresponding multiple FFT peaks in each demodulation window to the correct transmitters.

B. Limitations & Challenges

The main advantage of LoRa design is that it can concentrate time domain energy into a single tone frequency peak by dechirping and FFT [15]. The modulated chirps are inherently robust against channel noise. Thus, LoRa signals can be detected and decoded even under extremely low SNR (e.g. SNR as low as -20 dB) [16], enabling low power and long range communications [17], [18]. Existing collision decoding approaches do not thoroughly exploit the LoRa encoding properties and cannot work well under low SNR LoRa signals. For example, SIC usually focuses on time-domain signal decoding and cancellation and does not leverage the LoRa properties. As a result, existing collision decoding approaches [10], [12] cannot work for low SNR LoRa signals. We argue this significantly removes the major advantages of LoRa, which is supposed to provide long range and low power communications with very low SNR of even -20 dB.

C. Motivation

We leverage the fact that collided LoRa packets are likely to be misaligned in time. As shown in Fig. 1, the collided signal is divided into consecutive demodulation windows of L, where L is the symbol length. When a packet is not aligned with the window, there will be two LoRa segments in each window. Assuming the length of the first symbol and the second symbol are $\gamma_1 L$ and $\gamma_2 L$, respectively, we have $\gamma_1 L + \gamma_2 L = L$. We have two observations: (1) Given a specific LoRa packet, the in-window segment distribution, i.e., γ_1 and γ_2, are the same across all consecutive windows. (2) For two unaligned LoRa packets, their in-window segment distributions should be different. The in-window segment distribution can be applied to disentangle different packets in a collision.

III. NScale Design

A. Design Overview

Design Goals. NScale has the following design goals:

- NScale should work for low SNR LoRa collisions and incurs very small SNR loss to LoRa decoding.
- NScale should incur no modification to LoRa node and thus can be applied to COTS nodes and existing LoRa network deployments.
- NScale should incur small computation overhead compared with typical LoRa decoding and thus can be processed in real time.

Fig. 3 illustrates the main flow of NScale design:

Packet identification. For a received signal sequence, NScale first detects the existence of LoRa packets and identifies whether there is a collision. NScale leverages a cross-correlation based packet identification approach, which can detect LoRa packets even when multiple LoRa preambles collide together. When there is no collision, the signal is sent directly to a standard LoRa decoder. Otherwise, NScale utilizes the preambles to identify the beginning of each collided packet. Then the collided signal is divided into multiple consecutive demodulation windows for demodulating.

In-window distribution detection. For signal in each demodulation window, NScale transforms the in-window distribution of each low-SNR symbol into robust FFT peak

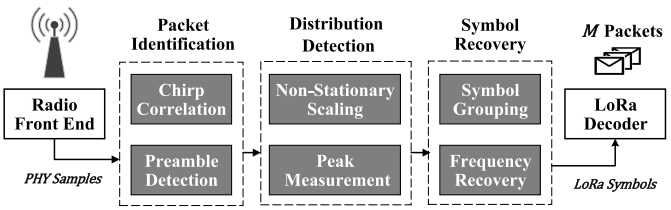


Fig. 3. Main workflow of NScale design.

features by non-stationary scaling. When two collided packets are not aligned, the in-window distributions of chirp segments for those two packets are different. Consequently, we can infer which packet the corresponding chirp segments belong to according to the segment distributions.

Symbol recovery. Based on the estimated in-window distribution information, NScale classifies symbols into multiple clusters, each corresponding to a collided LoRa packet. Before decoding, we estimate the CFO and the packet-window time offsets for each LoRa packet. Finally, NScale combines each pair of chirp segments into packet chirps, and the output chirps are fed to the standard LoRa decoder for packet decoding.

B. Packet Identification

Upon receiving a signal sequence, NScale first detects the existence of LoRa packets. NScale detects LoRa packets by identifying the preamble of each packet, where each preamble consists of N_c consecutive base up-chirps.

1) Preamble Detection: For detecting the incoming LoRa packet, NScale extracts each individual preamble chirp from the received signal by using a base up-chirp to correlate with the incoming signal. There will be a correlation peak when the base up-chirp is strictly aligned with a received preamble chirp. The intervals between each two adjacent correlation peaks of the same packet are identical, equal to the number of samples within a chirp, i.e., N. Denote $C[i]$ as the ith correlation output, our packet identification works as

$$find\ s, \quad s.t.\ |C[s+kN]| > \delta,\ \forall k \in [0, N_c - 1] \qquad (3)$$

where s is the start of packet preamble and δ represents the minimum correlation requirements, which can be determined by channel estimation. When multiple LoRa preambles collide together, after the cross-correlation, we can get multiple groups of equally spaced peaks, each corresponding to a LoRa packet, as shown in Fig. 4. We use Eq. 3 for grouping correlation peaks of the same preamble, and identifying the existence of collided packets based on the grouping result. We also get the start of each collided LoRa packet from the preamble identification.

The computational overhead of such a packet identification algorithm mainly stems from the sample-step moving correlation detection, whose time complexity is O(NM) where N denotes the number of samples in a chirp and M is the samples of the received signal.

2) Optimizing for Realtime Processing: In the packet identification, the sample-step moving correlation for detecting each individual preamble chirp dominate the computational overhead. In order to support real-time parallel decoding of LoRa transmissions, we propose an optimized method to avoid

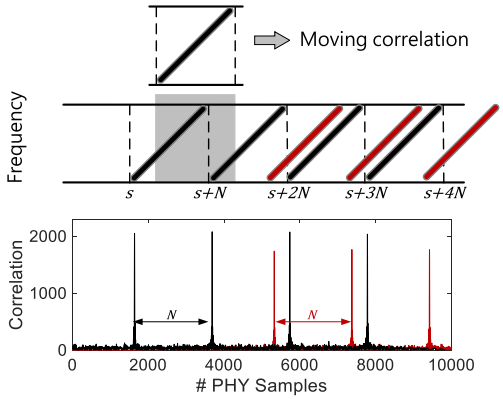


Fig. 4. Detecting LoRa packets by correlation with a single base up-chirp.

the costly sample-step correlation detection by leveraging the repeat pattern of LoRa preambles.

Our optimized packet identification approach leverages the fact that preamble chirps in the received signal have a high correlation with each other despite the existence of the CFO. Therefore, we can employ the self-correlation of the received signals to identify the existence of LoRa packets. When there is a LoRa preamble, the self-correlations of every two adjacent signal segments produce a significant correlation. Otherwise, the adjacent signal segments are completely uncorrelated, indicating the signal samples are noises or random payloads. Specifically, our optimized packet identification is mainly composed of three steps: (1) Upon the arriving of the signal samples, we first divide them into consecutive processing windows, each having the same length as a chirp, i.e., containing N samples. (2) Then, for samples in every two consecutive processing windows, we calculate their correlation through

$$SC_n = \sum_{k=1}^{N} s_n[k] s^*_{n+1}[k]$$

where $s_n[k]$ denotes the kth sample of the nth processing window, and $s^*_{n+1}[k]$ is the complex conjugate of the kth sample in the $n+1th$ processing window. As the instantaneous frequency of a preamble chirp increases linearly with time, the time offset between the processing window and the received LoRa packet introduces the same frequency shift for all preamble chirps. Thus, even the preamble chirps are not aligned with the processing windows, preamble segments in each reception window are still of high correlation. (3) Finally, we detect the existence of a LoRa preamble by searching $N_c - 1$ consecutive processing windows with significant correlations higher than a threshold, i.e., $SC_n > \delta,\ \forall n \in [K, K + N_c - 2]$. The cross-correlation between two unaligned preamble chirps of different LoRa packets is very low. Therefore, the correlation based approach can avoid inter-packet interference even when multiple LoRa preambles collide. After determining the existence of LoRa preambles, the receiver switches to individual-chirp correlation mode for getting the start of each collided packet. This optimized packet identification avoids computation-intensive sample-step moving correlation operation on preamble detection, and optimizes the time complexity from O(NM) to O(M).

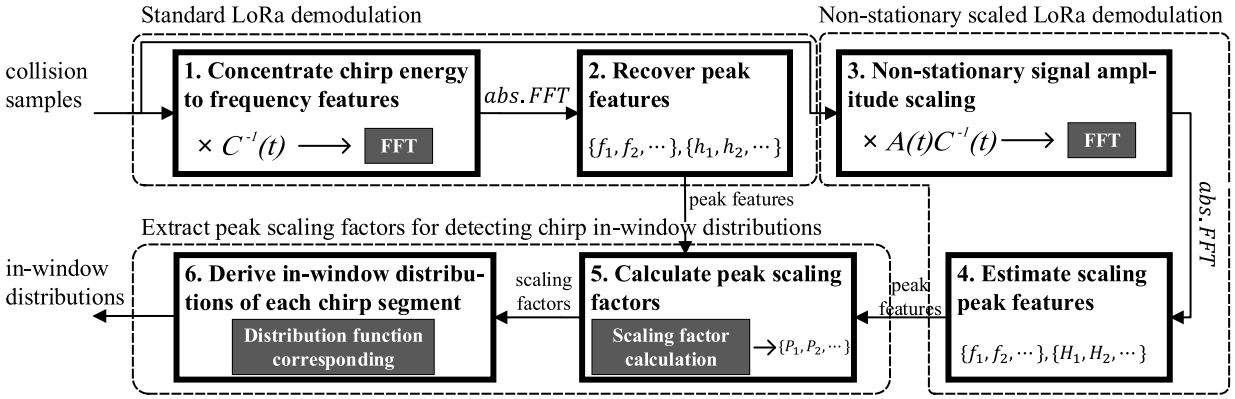


Fig. 5. Detecting in-window distributions for collided LoRa chirps with non-stationary signal scaling.

C. In-Window Distribution Detection

NScale separates LoRa collisions according to the in-window distribution of each chirp symbol. A practical challenge is precisely extracting the time domain segment distribution under low SNR. We address this from the following aspects: (1) translating the time-domain feature to robust FFT peak features in the frequency domain; (2) concentrating the energy in the time domain to frequency features by dechirping, which preserves the merits of LoRa decoding; (3) using the non-stationary signal scaling to further amplify the features. Fig. 5 summarizes the key steps of in-window distribution detection for collided LoRa chirps. (1) For signal in each window, we multiply it with a base down-chirp, and perform the Fourier transform on the multiplication result. This translates time-domain chirp segments to energy peaks in the frequency domain. (2) We further multiply the received signal with a non-stationary scaled down-chirp with varying amplitude over time. We transform the result of multiplication to energy peaks in the frequency domain, extracting each peak's frequency and height. (3) We pair the energy peaks from the above two steps according to peak frequencies, and calculate peak scaling factors as the height ratios of each pair of peaks. Given the scaling function of the non-stationary scaled down-chirp, we can derive the in-window distribution of each chirp segment from the peak scaling factors.

1) Modeling for Method Design: Suppose n LoRa packets collide at the receiver. We divide the collision signals into consecutive demodulation windows, each having the same length as a chirp. For separating collisions in each demodulation window, we have to extract the in-window distribution information for each chirp segment by first clustering chirp segments into two categories: the left segments adjacent to the start of the window (Fig. 6(a)) and the right segments adjacent to the end of the window (Fig. 6(d)). Fig. 6(a) shows an example of the left segment with a symbol-window time offset of Δt, i.e.,

$$C_L(t) = He^{j2\pi ft}C(t+\Delta t) \quad 0 \le t < T - \Delta t \tag{4}$$

where $C(t)$ is the base up-chirp, f and H denotes the initial frequency and amplitude of the received chirp. As the time offsets of chirps can be translated to frequency shifts, we have $C(t+\Delta t) = e^{j2\pi(k\Delta t)t}C(t)$, where k is the increasing rate of the chirp frequency. Similarly, the right segment in Fig. 6(d) can be written as

$$C_R(t) = He^{j2\pi ft}C(t-\Delta t) \quad \Delta t \le t < T$$

where Δt is the time offset between the chirp segment and the demodulation window. In the rest of this section, we will illustrate how to extract the in-window distribution of chirp segments for both $C_L(t)$ and $C_R(t)$.

2) Extracting In-Window Distributions: We illustrate our in-window distribution detection approach by taking $C_L(t)$ as an example. To initiate, we multiply the received signal by a base down-chirp (i.e., $C^{-1}(t)$) with stationary amplitude throughout the whole symbol duration. This multiplication dechirps the chirp segment into a single tone, where $C_L(t)$ is dechirped with the frequency of $f + k\Delta t$ and time range of $[0, T - \Delta t)$. After the multiplication, we perform FFT to aggregate the energy of the chirp segment to an energy peak in the frequency domain, as shown in Fig. 6(b). We perform zero-padding to the original signal before the Fourier transform to improve the frequency granularity. After LoRa demodulation, the energy of $C_L(t)$ is concentrated, leading to an energy peak in the frequency domain. Suppose the energy peak of $C_L(t)$ appears at the mth FFT bin, the height of that peak can be calculated as:

$$h_1 = |X_1[m]| = \Big| \sum_{n=0}^{N_0-1} C^{-1}[n] \times C_L[n] e^{-j2\pi \frac{nm}{N}} \Big|$$

where $X_1[m]$ is the FFT result at the mth FFT bin, $C_L[n]$ is the nth discrete sample of the chirp segment, N_0 and N represent the number of samples for the chirp segment and the whole demodulation window, respectively. Substituting $C_L[n]$ with Eq. 4, we have the peak height as $h_1 = H \times N_0$.

Beside the base down-chirp, we design a non-stationary scaled down-chirp $A(t)C^{-1}(t)$, whose amplitude changes over time with a known scaling function $A(t)$. We multiply the received signal with the non-stationary scaled down-chirp. After the Fourier transformation on the multiplication result, $C_L(t)$ is also translated to an energy peak at the mth FFT bin (as shown in Fig. 6(c)) with the height:

$$h_2 = |X_2[m]| = \Big| \sum_{n=0}^{N_0-1} A[n]C^{-1}[n] \times C_L[n] e^{-j2\pi \frac{nm}{N}} \Big|$$

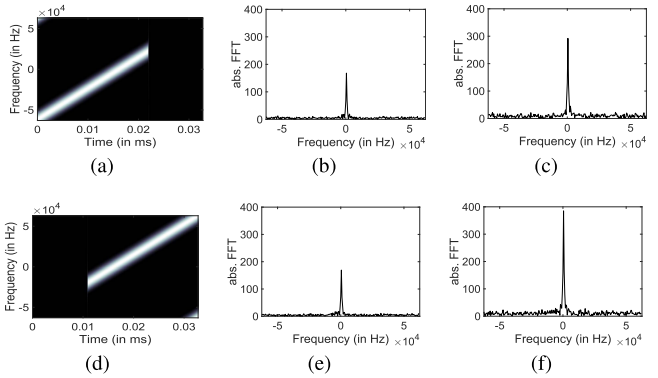


Fig. 6. In-window distribution detection: (a)(d) Symbol segments with different in-window distributions, (b)(e) demodulation with the base down-chirp, (c)(f) demodulation with the non-stationary scaled down-chirp.

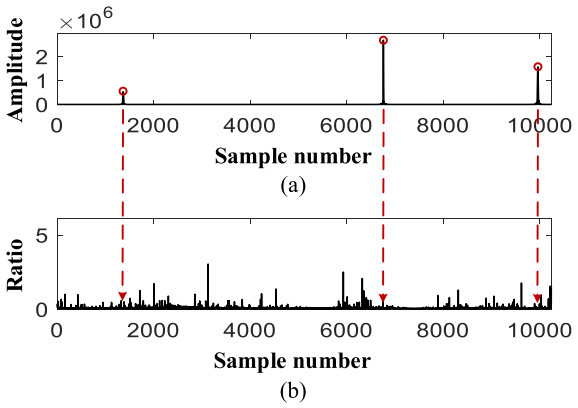


Fig. 7. Optimized peak scaling factor detection: (a) Multiplication and (b) division of the two FFT results in Step 1 and Step 3.

where X_2 is the FFT result with the non-stationary scaled down-chirp, $A[n]$ is the discrete sample of scaling function. Substituting $C_L[n]$ with Eq. 4, the height of the energy peak is simplified as $h_2 = H\sum_{n=0}^{N_0-1} A[n]$.

Note that the non-stationary scaling on the down-chirp does not affect the frequency of the FFT peaks. The energy peaks of both h_1 and h_2 locate at the same frequency, i.e., $f+k\Delta t$. The above procedures also translate the right segment in Fig. 6(d) into two FFT peaks with the same frequency, as shown in Fig. 6(e) and (f). In the presence of collisions, multiple overlapped chirp segments can be translated into FFT peaks simultaneously through a single dechirping. For each chirp segment, we pair the energy peaks from the two multiplications according to the peak frequencies. For peaks of the same chirp segment, we calculate the peak scaling factor P as the ratio of peak heights

$$P = \frac{h_2}{h_1} = \frac{\sum_{n=0}^{N_0-1} A[n]}{N_0} \tag{5}$$

As the scaling function $A[n]$ is already known, we can infer the in-window distribution for each chirp segment directly from the peak scaling factor P. We finally derive the in-window distributions for every chirp by corresponding the peak scaling factors to the known amplitude scaling function.

The computational overhead for in-window distribution detection mainly stems from the peak feature estimation, where peaks are iteratively extracted and canceled in each demodulation window as illustrated in Sec. III-D. For each chirp segment, the peak estimation is performed twice with the base down-chirp and the scaled down-chirp respectively, which incurs prominent computing overheads.

3) Optimizing In-Window Distribution Detection: To support real-time processing, we propose an optimized in-window distribution detection approach, avoiding the repetition of the costly peak feature estimation. Our optimized method leverages the fact that FFT peaks with and without non-stationary scaling locate at the same frequency. Therefore, we can combine these two FFT results to derive peak scaling factors directly without estimating peak features from the two FFTs. Specifically, this is done by multiplying the energy of the two FFT outputs on every frequency, and taking a square-root. As a result, the non-empty frequencies that correspond to FFT peaks are amplified. In contrast, the frequencies corresponding to noises will be attenuated as shown in Fig. 7(a). We also calculate the scaling factors on every frequency by dividing the energy of the two FFTs as shown in Fig. 7(b). Finally, we derive the in-window distribution for each chirp by searching energy peaks from the multiplication and obtaining scaling factors of the same frequency from the division. Compared with the non-optimized method, we estimate the peak feature from the multiplication only once, and thus we can save the computation-intensive repetition in the peak estimation to reduce the processing time.

At this point, we can calculate the in-window distribution of chirp segments. Then, two consecutive chirp segments for the same chirp are paired and merged by searching peaks with the same frequency in consecutive windows. Finally, we can obtain all chirp symbols extracted from the collision, each with the estimated in-window distribution information.

D. Peak Estimation in Practice

The in-window distribution calculation relies on accurately estimating FFT peaks from the LoRa collisions, including both the frequency and the height of peaks. However, accurate peak estimation is challenging due to the peak distortion caused by phase rotation property of the Fourier transform. In this subsection, we discuss the reason of peak distortion and further show how to improve the estimation accuracy.

In practice, the phase rotation of the Fourier transform distorts frequency peaks. A shift in the time domain can be translated into a phase rotation in the frequency domain [19]:

$$\begin{aligned} \mathcal{F}\{r(t)\} &= R(f) \\ \mathcal{F}\{r(t+\tau)\} &= R(f)\cdot e^{j2\pi f\tau} \end{aligned} \tag{6}$$

where $r(t)$ is the signal in the time domain, and $R(f)$ is the corresponding frequency-domain representation. LoRa conveys data by cyclically shifting the frequency of base up-chirps. After dechirping, a LoRa symbol generates two frequencies, i.e., f and $f-BW$, respectively. When the sampling rate is equal to the chirp bandwidth BW, the Fourier transform of the two frequencies will result in two peaks, denoted as $R_1(f)$ and $R_2(f)$, at the same location. If the LoRa chirp accurately aligns with the demodulation window, $R_1(f)$

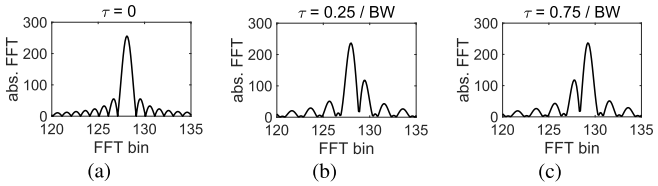


Fig. 8. Peak distortion example: (a) Ideal peak when a chirp is aligned with the demodulation window. (b)(c) Distorted peaks when a chirp and demodulation window are misaligned by a time offset τ.

and $R_2(f)$ add up constructively, resulting in an ideal peak as shown in Fig. 8(a). However, when the LoRa symbol and the demodulation window are misaligned (suppose the time offset is τ), the Fourier transform of the two frequencies will rotate by different phases. Recall the phase rotation property in Eq. 6, with the same time offset τ, peaks of different frequencies have different phase shifts, i.e., $R_1(f) \cdot e^{j2\pi f\tau}$ and $R_2(f) \cdot e^{j2\pi(f-BW)\tau}$. Those two peaks with different phase shifts add up destructively, resulting in peak distortions. As shown in Fig. 8(b) and (c), with a different time offset τ, peaks in the frequency domain are distorted differently, impacting the estimation of the frequency and height of peaks.

NScale recovers FFT peaks from distortion by compensating the phase rotation of the two frequencies, i.e., f and $f-BW$, which relies on the fact that the sampling rate of off-the-shelf ADCs is much higher than the chirp bandwidth BW. When sampling the received signal at a high rate (e.g., higher than $2BW$ according to the Nyquist-Shannon theorem [20]), the Fourier transform of the received signal will result in two separate peaks, $R(f)$ and $R(f-BW)$, respectively. To compensate the phase rotation, NScale searches the phase difference between $R(f)$ and $R(f-BW)$ via:

$$\phi = \arg \max_{0<\phi\leq 2\pi} R(f) \cdot e^{j\phi} + R(f-BW) \tag{7}$$

The maximum can be obtained only when the phase rotation effect is compensated. Then we can estimate the peak accurately. It is worth noting that we can apply stochastic gradient-descent algorithms on Eq. 7 with randomly chosen initial points that are likely to converge to the global maximum to speed up the searching process.

E. Symbol Recovery

To decode the collisions, we further need to group symbols into different packets and then recover the precise information of each symbol.

NScale utilizes a constrained k-means based approach to group the symbols into k clusters (i.e., the k collided packets). We determine the number of clusters, i.e., k, based on the packet identification result. NScale uses a cross-correlation based approach to detect LoRa preambles from the received signal, which gives the packet number k as well as the start of each collided packet. This information is used for initializing the k-means based symbol grouping approach. Then, we use the in-window distribution, which is identical for symbols of the same packet but distinct for symbols of different packets, as the characteristic value for symbol clustering. We apply several constraints to the clustering

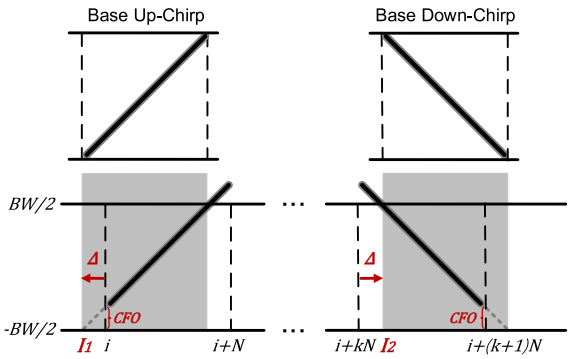


Fig. 9. Detecting accurate packet start by eliminating the impact of CFO.

method from the following aspects. (1) Symbols are grouped to the clusters in time order. (2) A new cluster can only emerge and start gathering symbols after detecting the start of a packet. (3) Each cluster has one and only one symbol in each demodulation window. Based on the constraints, we obtain k groups of symbols, each of which corresponds to a collided packet. Finally, we send the recovered symbols to a standard LoRa decoder for decoding.

Before packet decoding, we need to (1) eliminate the impact of CFO to recover the accurate frequency of each chirp, and (2) find the accurate start of each packet to compensate for the time offset of each chirp. A practical challenge is that the CFO and the packet time offset are cross dependent, impacting the estimation of each other. We leverage the unique structure of LoRa packets in accurately calculating CFO and packet time offsets. A LoRa packet consists of both base up-chirps (i.e., preambles) and base down-chirps (i.e., SFDs). One key finding is that for the same amount of CFO, the correlation peaks of up-chirps and down-chirps shift oppositely. As shown in Fig. 9, assuming a LoRa packet is received with a positive CFO, we correlate the received signal with a base up-chirp followed by a base down-chirp, respectively. The correlation peak of the up-chirp is shifted by $-N \cdot CFO/BW$, while the peak of the down-chirp is shifted by $N \cdot CFO/BW$. Denote $\Delta = N \cdot CFO/BW$. For the base up-chirp, the correlation peak appears at $I_1 = i - \Delta$, where i is the actual start of the packet. While for the base down-chirp, the correlation peak appears at $I_2 = (i + kN) + \Delta$. We calculate the shift of correlation peaks as

$$\Delta = MOD(I_2 - I_1, N)/2 \tag{8}$$

Then we can calculate CFO as

$$CFO = \Delta \times BW/N \tag{9}$$

Therefore, we can eliminate the impact of CFO and time offset for accurate frequency estimating in packet decoding.

IV. Designing Non-Stationary Scaling

NScale calculates the in-window distributions of collided symbols using non-stationary signal scaling. Therefore, the design of the scaling function impacts the decoding performance. We show the effect of the non-stationary scaling function on the decoding performance of NScale, and present strategies on designing an effective scaling function.

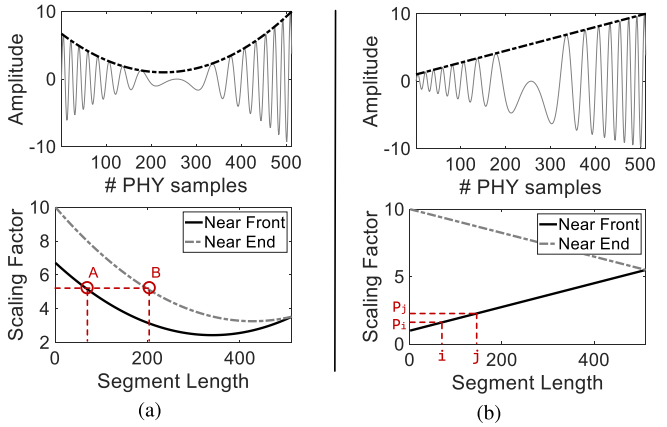


Fig. 10. Non-stationary scaling function design: (a) A non-monotonous function corresponds two different chirps to the same peak scaling factor, (b) A linear function maximize the mean inter-object distances.

1) Universal Rules for Scaling Function Design: Given an arbitrary LoRa packet collision, there are two rules for non-stationary scaling function design. The first is being monotonous. Through non-stationary amplitude scaling, the in-window distributions of chirps are translated to peak scaling factors in the frequency domain (i.e., P_i in Eq.5). As shown in Fig. 10(b), for a monotonous scaling function, the chirp segments with different in-window distributions can generate different peak heights, leading to unique peak scaling factors. Otherwise, for a non-monotonous scaling function, symbols with different in-window time distributions may result in the same peak scaling factor, leading to ambiguity. We show an example in Fig. 10(a) with a down-chirp of non-monotonous scaling function. The bottom of this figure is the relationship between peak scaling factors and in-window distributions. As shown in Fig. 10(a), the points A and B in this figure represent two chirp segments of different in-window distributions. Due to the non-monotonicity of the scaling function, these two segments with distinct distributions generate the same peak scaling factor, making it difficult to distinguish them.

The scaling function should also be linear. As illustrated in Sec. III-E, NScale groups collided symbols into different packets according to the in-window distribution, which is reflected by the peak scaling factor in Eq. 5. For accurate distribution calculation, the peak scaling factors for symbols of different in-window distributions should be different. Thus, the goal of our non-stationary scaling design is to maximize the difference in peak scaling factors of different in-window distributions. For two different in-window distributions i and j, we define the distance between their peak scaling factors as

$$d(i,j) = \frac{|P_i - P_j|}{\max\{\mathbb{P}\} - \min\{\mathbb{P}\}}$$

where P_i and P_j are the scaling factors for i and j, as shown in Fig. 10(b); and $\mathbb{P}$ is a set of all possible peak scaling factors. For a demodulation window with N sample points, there are $2N-1$ possible in-window distributions corresponding to $2N-1$ different peak scaling factors, i.e., $|\mathbb{P}| = 2N-1$. All scaling factors are normalized to the range of $(0,1]$ by dividing $\max\{\mathbb{P}\} - \min\{\mathbb{P}\}$.

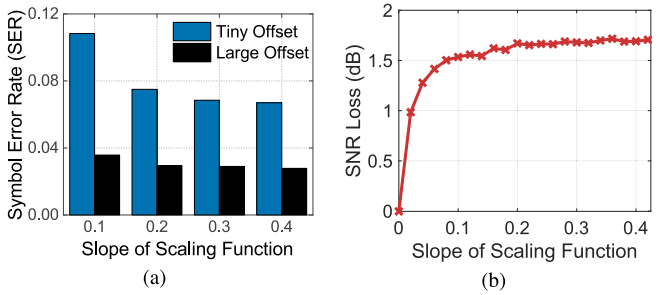


Fig. 11. Performance comparison of linear scaling functions with different slopes: (a) Averaged SER. (b) SNR Loss.

Given a specific scaling function, the set of all possible peak scaling factors, i.e., $\mathbb{P}$, can be derived according to Eq. 4. Each object in $\mathbb{P}$ corresponds to a different in-window distribution. For object $i \in \mathbb{P}$, let

$$a(i) = \frac{1}{|\mathbb{P}|-1} \sum_{j \in \mathbb{P}, j \neq i} |d(i,j)|^2$$

be the mean distance between i and all other objects. We can interpret $a(i)$ as a measure of how well i is separated from other in-window distributions (the bigger the value, the better the separation). Thus, the problem of designing non-stationary scaling is to finding a collection of $\mathbb{P}$ to maximize the mean inter-object distances, i.e.,

$$\mathbb{P}_{opt} = \arg\max_{\mathbb{P}} \frac{1}{|\mathbb{P}|} \sum_{i \in \mathbb{P}} a(i) \tag{10}$$

Given a monotonous scaling function and the number of j greater than i, as shown in Fig. 10(b), we have $d(i,j) = \sum_{k=i}^{j-1} d(k,k+1)$. The mean inter-object distances achieves maximum only when the distances between each pair of neighboring objects are identical, i.e., $d(k,k+1) = \frac{1}{|\mathbb{P}|}, \forall k \in \mathbb{P}$, indicating that the scaling function should be linear.

We further evaluate the NScale's decoding performance under different linear scaling functions. We use the slope of a linear scaling function to represent the change of its amplitude (starting at 1) over the whole chirp duration. We use linear scaling functions with different slopes to decode the same set of two-packet collision. As shown in Fig. 11(a), for collisions with small inter-packet offsets ($< 10\%$ symbol duration), the SER decreases as the slope increases. This is because scaling functions of large slope magnify the small time offsets. While for collisions with large time offsets ($> 35\%$ symbol duration), the slopes have less impact on the SERs, as the packets can already be distinguished. We also examine the SNR loss for scaling functions of different slopes. We vary the SNR of the received LoRa signal by manually adding white Gaussian noise, and evaluate the minimum decoding SNR requirement for both original LoRa and NScale with different linear scaling slope. The results of the experiment are shown in Fig. 11(b). We can see from the results that NScale introduces less than 1.7 dB SNR loss compared with the original LoRa.

2) Optimizing Scaling Function for Specific Circumstances: Linear function maximizes the average inter-object distances between any two potential in-window distributions, and can achieve an adequate performance for decoding LoRa collisions

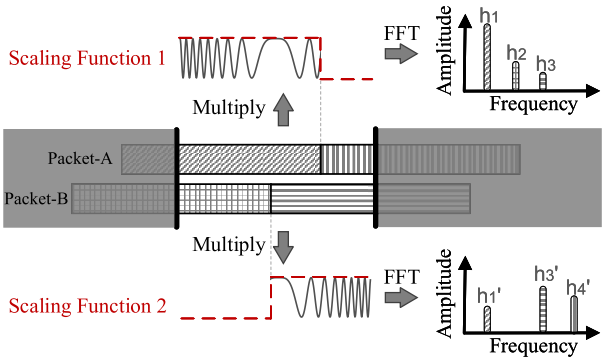


Fig. 12. Optimized scaling function design for collision circumstances with known packet arrival times.

with arbitrary signal arriving times. While, for some specific collision circumstances, a specially designed scaling function may achieve more prominent performance. For example, to resolve a particular LoRa collision, we can use a dedicated scaling function, designed according to the packet arrival time, to amplify time distribution difference of overlapping chirps. The linear function maximizes the average difference between any two potential in-window distributions. While the dedicated scaling function maximizes the difference between two particular distributions. Thus, the latter can achieve a better decoding performance for a specific collision.

We use a two-packet collision example to illustrate our design for the optimized scaling function. As shown in Fig. 12, two packets collide at the receiver. The receiver first estimates the arrival time of each LoRa packet [21]. Denote the time offsets between the demodulation window and the collision chirps are T_1 and T_2, respectively. We design the two scaling functions according to the edges of collision symbols. Assuming $T_1 \geq T_2$, the first scaling function can be written as

$$A_1(t) = \begin{cases} 1 & 0 \leq t < T_1 \\ 0 & T_1 \leq t < T \end{cases}$$

and the second scaling function is

$$A_2(t) = \begin{cases} 0 & 0 \leq t < T_2 \\ 1 & T_2 \leq t < T \end{cases}$$

When multiplying the collision signal $s(t)$ with a base down-chirp scaled by $A_1(t)$, right segments in $s(t)$ with in-window distribution over $[T_1, T)$ are suppressed. The FFT on the multiplication results on three energy peaks corresponding to two left segments and a partial of a right segment. Then, we multiply the collision signal with a base down-chirp of $A_2(t)$ and perform FFT, which retrains left segments distributed within T_2 and results on three FFT peaks corresponding to two right segments and a partial of a left segment. Comparing the demodulation results of $A_1(t)$ with that of $A_2(t)$, the peak height of h_1 decreases, h_3 increases, h_2 disappears, and h_4 is new emerging. Thus, the scaling factors for these four peaks are < 1, 0, > 1, $+\infty$, respectively. Therefore, by exploiting the frequency response of the two optimized scaling functions, we can identify the in-window distributions of every chirp segment with the maximum resolution. Besides, as the optimized scaling functions suppress some interfering segments before the FFT, it eliminates the inter symbol interference and facilitates peak measurements in the frequency domain.

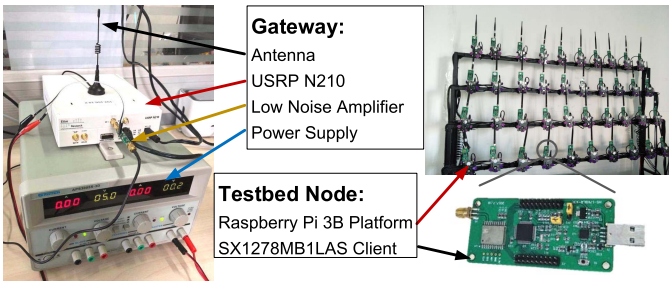


Fig. 13. LoRa Gateway and Testbed Setup: Depicts NScale's USRP N210 based gateway and commodity client based LoRa testbed.

3) Summary: Linear scaling function is a good design for universal LoRa collisions with random packet arrival times. The slope of the linear scaling function can be determined according to the inter-packet offset and SNR of the received collisions. For LoRa collisions with known packet arrival times, we can use dedicated designed scaling functions for further optimizing the collision decoding performance.

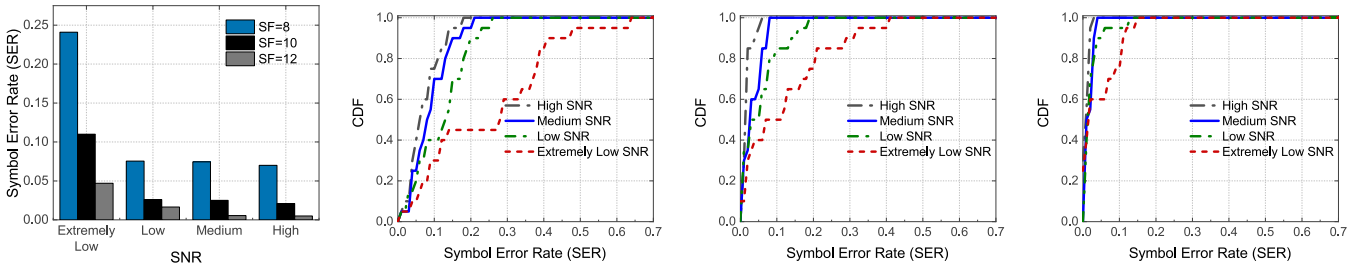
V. Evaluation

We implement NScale on the software defined radios (SDRs) and evaluate its performance with commercial LoRa devices. The prototype of NScale is shown in Fig. 13, which is composed of a USRP N210 along with a UBX daughter board, operating at the 470MHz bands. Decoding algorithms of NScale are hardware-independent, so it can be implemented on any other commercial LoRa gateways as long as the physical samples can be obtained. Note that LoRa gateways are usually deployed with tethered power supplies, and thus we do not consider energy consumption at the gateway. We use the UHD+GNU-Radio library [22] for developing our own LoRa demodulator, and implement NScale in MATLAB to process the PHY samples offline. By default, our experiment uses the spreading factor SF $= 10$, coding rate CR $=$ 4/5 and bandwidth BW $=$ 125 kHz. The sampling rate of NScale in our experiment is set to 1 MS/s.

A. Evaluation Methodology

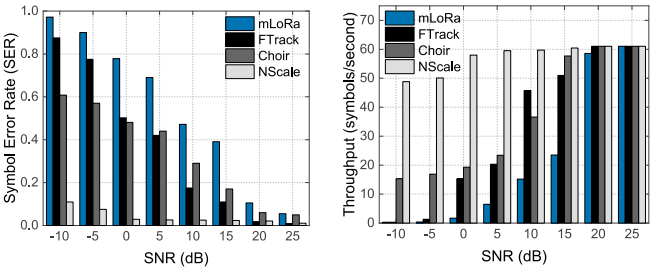
1) Experiment Environments: We evaluated NScale's performance in both indoor and outdoor environments.

- As shown in Fig. 13, the indoor testbed (*LoRaNet*) consists of 40 LoRa end nodes, each of which uses an SX1278 radio chip [23] and works at the frequency of 470 MHz. Each node is connected to a Raspberry Pi and placed at a fixed position on a shelf. All the Raspberry Pis are connected to a backbone network and thus all the LoRa nodes can be efficiently and accurately controlled to facilitate precise collision generation and measurement.
- The outdoor LoRa testbed is composed of enclosed nodes shown in Fig. 19, each of which can sense the temperature and humidity of the environment and transmits the sensed data to the gateway via an SX1268 LoRa radio chip.



(a) Overall SER performance (b) CDF of SERs (SF=8) (c) CDF of SERs (SF=10) (d) CDF of SERs (SF=12)

Fig. 14. In-depth study of SNR and SF on NScale's performance: (a) Overall performance of averaged SER. (b-d) CDF of the SER with different SNRs and SFs.



(a) (b)

Fig. 15. Performance comparison of four methods under different SNRs: (a) Averaged Symbol Error Rate (SER). (b) Network throughput.

The nodes of the outdoor testbed can harvest energy from solar power, making them easy to deploy on different locations such as roads, roofs, and parking lots.

2) Compared Methods: We compare NScale with three recent works for LoRa collision decoding.

- Choir [11]: A collision decoding method for LoRa using hardware imperfection.
- FTrack [12]: A collision decoding method for LoRa based on time domain signal analysis.
- mLoRa [10]: A collision decoding method for LoRa based on SIC.

B. Comparing With Existing Works

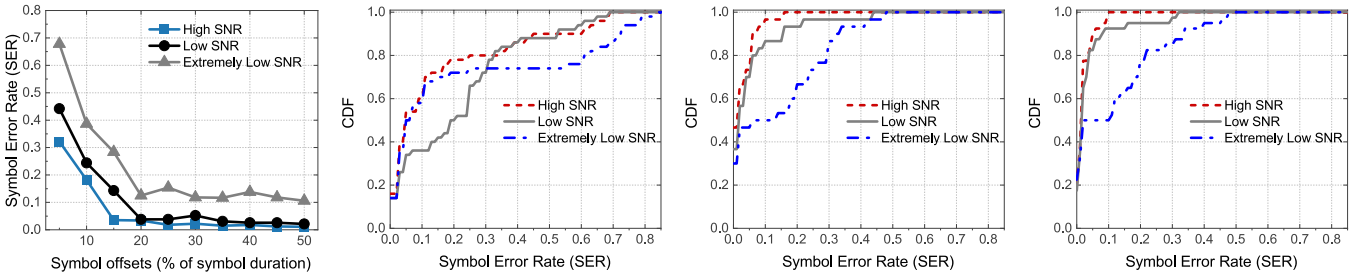
We first compare NScale's performance with three existing works. Three LoRa nodes are used for generating LoRa collisions. We configure one node to send beacons every 3 seconds. Upon receiving a beacon, the other two nodes each replies with a LoRa packet, to generate collisions. We configure an additional processing delay (smaller than a packet duration) for each transmitter to generate different misalignment. Thus, packets from transmitters collide at different parts with different time offsets (e.g., preambles, sync words, SFDs and payloads). We vary the transmitting power of the transmitters to generate collisions with different SNRs. For fine-grained SNR control, we add white Gaussian noise with controlled amplitudes to the collected I and Q traces.

We compare the performance of NScale with the other three LoRa demodulation schemes, in terms of SNR. Fig. 15 shows the results of the experiment. When packets collide with a relatively high SNR ($>$ 20 dB), both NScale and FTrack experience a low symbol error rate (SER $<$ 0.01) as well as a high network throughput. Choir and mLoRa fail to decode some of the concurrent transmissions even under such high SNR conditions. Choir uses the fraction of the FFT bin to distinguish collided symbols, which has errors due to the frequency offsets of low-cost LoRa nodes drift over time. Thus, collided symbols may be classified incorrectly, resulting in decoding errors. mLoRa uses an SIC based approach for decoding packet collisions, which suffers from error propagation. This leads to symbol recovering errors for mLoRa, especially when the packet length is long or the concurrency increases. As the SNR decreases, the SER of mLoRa and FTrack increase rapidly, because both of these two methods have fundamental limitations in decoding low SNR LoRa signals. For collisions under extremely low SNR (< -5 dB), both mLoRa and FTrack even turn to be invalid, resulting in a network throughput close to zero. The performance of Choir also decreases for low SNR situations, as the tiny hardware offsets are vulnerable to noise interference. NScale performs much better than the other three methods. When packets collide under extremely low SNR (-10 dB), the network throughput of NScale (49 symbols per second, sps) is about $3.3\times$ of Choir (15 sps).

C. Basic Performance

In this subsection, we examine NScale's basic performance for separating LoRa collisions regarding spreading factors (SFs), SNRs, and inter-packet offsets.

First, we evaluate the impact of SNR on the performance of NScale. The decoding performance is evaluate under four SNR regimes: high ($>$20 dB), medium (5$\sim$20 dB), low ($-5\sim$5 dB) and extremely low (< -5 dB). High channel noise cause peaks of chirp segments suffering from distortion, which further disturbs the detection of symbol in-window distribution. Fig. 14 shows the SER for NScale under different levels of SNRs with different SFs. NScale can decode collisions even with extremely low SNR as it concentrates the energy of a chirp and translate the time domain information to robust peak features in the frequency domain. As shown in Fig. 14(a), the SER is low under all four different SNR levels. The performance slightly degrades for extremely low SNR scenarios. However, those errors can mostly be recovered by the Forward Error Correction strategy of LoRa. Fig. 14(b-d) shows the relationships between SERs and SNR levels regarding three different SFs. We observe that in high, medium and low SNR



(a) Overall SER performance (b) Small offsets (c) Medium offsets (d) Large offsets

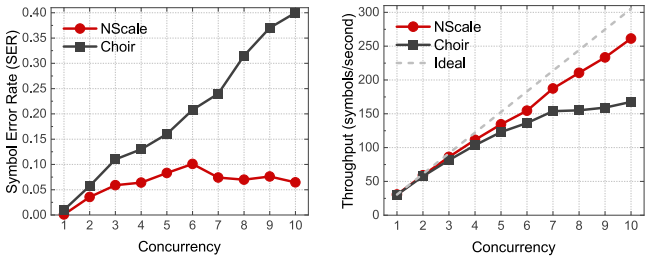
Fig. 16. Relationship between the SER and symbol offsets of collisions: (a) Overall performance of averaged SER. (b-d) CDF of the SERs when symbol offset is small ($< 20\%$), medium ($20\% \sim 35\%$) and large ($> 35\%$), respectively.

conditions, the SERs of 100% with high SF (SF12 and SF10) and 90% with small SF (SF8) are lower than 20%. This is because NScale reduces some of the noise interference, making its SER performance robust against channel noise. In the situation of extremely low SNR, the median SERs for SF8, SF10, and SF12 are 0.28, 0.07, and 0.02, respectively. In practice, small SFs in LoRa are used for near-range high data rate transmission. Therefore, we can increase SF for the scenario of low SNRs to improve the decoding performance to a very low SER of 0.02.

We also explore the impact of inter-symbol time offsets, i.e., packet misalignment. It has been shown that the NScale leverages the time offset information to separate collided packets. We examine how the inter-symbol time offsets affect NScale's performance. Fig. 16(a) shows the averaged SER of NScale in terms of the inter-symbol time offsets and SNR levels. The SER performance of NScale decreases when packets collide with smaller inter-symbol time offsets. This is because a small offset leads to a small difference between in-window distributions, which further makes it difficult to distinguish collided packets. While in practice, nodes in LoRa transmit packets in random time, where the inter-symbol offset follows a uniform distribution within a symbol duration. Thus, the inter-symbol offsets vary in practice, and NScale can successfully separate collisions in most cases. We can further solve the decoding failures by using the retransmission mechanism of LoRaWAN protocol. Fig. 16(b-d) present the detailed SER performance regarding SNRs and inter-symbol time offsets. For collisions with small offsets, decoding errors are mainly due to the ambiguity of in-window distributions. And the influence of the SNR is not that obvious, as high SNR collisions may also fail to decode due to ambiguous in-window distribution clustered incorrectly. The SER in Fig. 16(b) is higher than that of medium offsets (Fig. 16(c)) and the large offsets (Fig. 16(d)) under all three SNR levels. While for collisions with median and large offsets, the median SERs of NScale for both high SNR and low SNR scenarios are below 0.02, indicating that most collided packets are correctly decoded.

D. Impact of Concurrency

In this experiment, we examine the scalability of NScale by decoding LoRa collisions with different number of concurrent transmissions. As mLoRa and FTrack cannot work for $SNR < 0$, we only show the performance of NScale and Choir. We use the indoor *LoRaNet* testbed to efficiently generate multi-packet collisions. To produce a collision with m overlapped packets, we use a beacon to synchronize transmissions for m different end nodes. At the gateway, we use NScale and Choir to decompose the collided packets. The packets sent by each end node is known in prior. Thus, we can calculate the SER and network throughput in this experiment.



(a) (b)

Fig. 17. Decoding collided transmissions with different concurrency. (a) Averaged SER. (b) Network throughput.

Fig. 17(a) shows the SER for NScale and Choir. As the number of concurrent nodes increases from 1 to 10, the SERs of both NScale and Choir grow up. We can see that the SER of NScale increases much more slowly than that of Choir, because NScale extracts more efficient features to separate packets while Choir uses hardware imperfection, which is less stable and difficult to detect, especially under inter-chirp interference and channel noise. We further investigated the performance of NScale and found that a symbol error happens when the in-window distribution is incorrectly detected or a symbol is incorrectly clustered to a packet.

We also show the overall network throughput in Fig. 17(b). The network throughput of both NScale and Choir increase as the number of concurrent transmitters increases, due to the benefit of multi-packet reception from collision resolution. Meanwhile, the network throughput of Choir is much lower than that of NScale, especially for the large concurrency situations. That is because the hardware offsets in Choir inevitably resemble each other as the number of LoRa nodes increases, which leads to symbol clustering errors. We will further show the performance of NScale in the outdoor real deployed LoRa networks in Sec.V-F.

E. Real Time Performance

We evaluate the performance of NScale in computation time for decoding LoRa collisions. We implement LoRa receivers

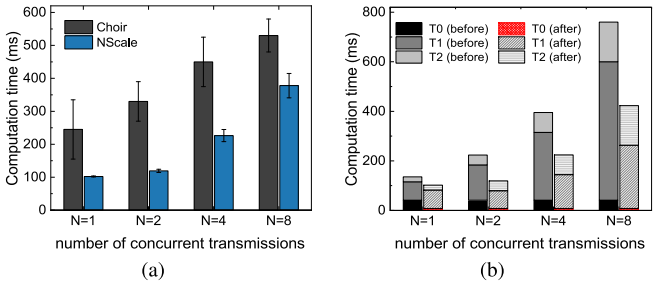


Fig. 18. Performance comparison of computation time: (a) The overall computation time for choir and NScale, (b) Anatomy of the time costs of NScale: T_0 is time of packet identification, T_1 is the time of in-window distribution detection, and T_2 is the time of symbol clustering and demodulation.

of Choir and NScale, and run the receivers on a PC with the CPU of Intel Core i7. We use the receivers to process the received LoRa collisions with various number of concurrent transmissions. Fig. 18(a) shows the overall computation time of Choir and NScale receivers. Generally, the computation time of Choir is longer than that of NScale. The Choir receiver decomposes LoRa collisions by exploiting the fraction of a FFT bin for each energy peak. Therefore, it performs Fourier transforms over a wider window ($10\times$ larger) by zero-padding the signal, which introduces a much higher computational overhead than NScale. The computation time of both Choir and NScale increases as there are more concurrent transmissions involved in the collision. This is because the increase of concurrent transmissions leads to more FFT peaks in each demodulation window, which requires much more processing time for identifying and estimating each single FFT peak. However, as devices in LoRa transmit packets under a extremely low duty cycle, there is sufficient time for LoRa receivers to decode collisions.

We further examine the effectiveness of our proposed real-time optimizations for NScale, i.e., packet identification in Sec III-B and in-window distribution detection in Sec III-C. Fig. 18(b) plots the anatomy time costs of NScale with and without real-time optimizations. The computation time of NScale for resolving a LoRa collision is mainly divided into three parts, i.e., packet identification (T_0), in-window distribution detection (T_1), and symbol clustering and demodulation (T_2). The in-window distribution detection dominates the computation time of collision resolving, where the LoRa receiver estimates and recovers the signal of each overlapped chirp. The optimized method significantly reduces the time costs of this step, as it avoids repeatedly estimating FFT peak features from the two demodulations (i.e., demodulations with the standard down-chirp and the non-stationary scaled down-chirp). Besides, the optimized method reduces the packet identification time from hundreds of milliseconds to a few tens of milliseconds, by avoiding the operation of computation-intensive sample-step moving correlations. For the future work, we can adopt more efficient hardwares, such as FPGA, multi-processor and ASIC, to further accelerate NScale so as to meet the real-time processing requirement.

F. Performance in a Real Network

We evaluate the performance of NScale in a real deployed LoRa network. As illustrated in Fig. 19, the outdoor LoRa

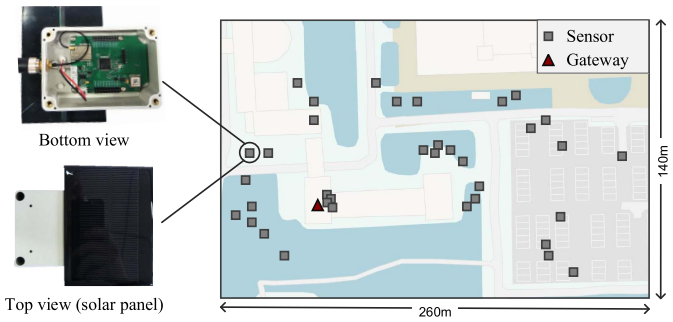


Fig. 19. Outdoor Real LoRa Network: LoRa nodes with temperature and humidity sensors are placed around the campus, which consist of SX1268 radio chip and operate by harvesting solar power.

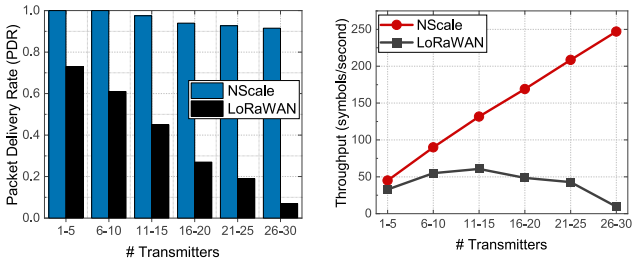


Fig. 20. Performance in real outdoor deployed LoRa networks: (a) Packet Delivery Rate. (b) Network throughput.

network consists of 30 LoRa sensor nodes, each of which can sense the temperature and humidity information from the surrounding environment. We deploy the sensor nodes across various environments with different distance from the gateway, and the SNR of the received signal varies from -15dB to 10dB. With an integrated solar panel, each sensor node collects and transmits sensed data by harvesting solar power. The LoRa sensors transmit to the LoRa gateway in a duty-cycled manner, where we set the duty cycle ratio of each node to 10%. In the experiment, we change the number of active nodes in the network, and evaluate NScale's performance regarding to different network sizes.

Fig. 20 shows the performance of NScale and original LoRaWAN in the outdoor LoRa network. For the LoRaWAN receiver without collision resolution, the Packet Delivery Rate (PDR) decreases rapidly as the network scales. The network throughput of the LoRaWAN receiver first grows up and then rapidly drops down as the size of the network increases. When the network size is small, the increase of concurrent nodes improves the channel utilization. However, when the network scales, frequent collisions occur, which significantly degrades the performance of the LoRaWAN receiver.

The performance of NScale is much better than the original LoRaWAN due to NScale's decoding advantage. As the network scales, NScale shows a relatively high PDR, where more than 90% packets are successfully delivered even when the size of the network reaches 30. The overall network throughput of NScale increases as the number of transmitters goes up. As shown in Fig. 20(b), when 30 sensor nodes operate simultaneously, the network throughput of NScale (247 sps) is about $27\times$ than that of the original LoRaWAN (9 sps).

VI. Related Work

Resolve collisions in traditional wireless: Combating signal collision is a traditional problem in wireless. There are many excellent works in this area. Some works aim to avoid collisions by using MIMO/MU-MIMO [24]–[27] or fallback strategy [28] on wireless devices. They synchronize signal phases from different transmitters, enabling concurrent transmissions without inter-packet interference. However, such solutions have high demands on hardware overheads, and thus is not appropriate for LPWAN devices. Successive Interference Cancellation (SIC) eliminates signal collisions by estimating and extracting decoded symbols iteratively [29]–[31]. ZigZag [32] combats signal collisions in 802.11, where the collision free chunk, due to the misalignment between collided packets, are leveraged for separating overlapped signals. For decoding an m-packet collision, ZigZag requires each end node retransmitting m times to generate m repeated collisions. Similarly, mZig [33] also leverages the packet misalignment to decode collisions in ZigBee networks. It can decompose m concurrent ZigBee packets from one collision directly. Both the two approaches decode collisions based on the signal in the time domain, and they cannot work well for low SNR LoRa signal.

Parallel transmissions in LoRa: This work is inspired by some recent advances for concurrent transmission and collision resolution in LoRa. Netscatter [34] migrates LoRa encoding mechanism to backscatter devices and enables hundreds of concurrent transmissions for backscatter systems. Transmissions in Netscatter are strictly synchronized. Thus, we cannot apply Netscatter in current LoRa networks, where end nodes transmit to the gateway in the manner of ALOHA. DeepSense [35] enables random access and coexistence for LPWANs by identifying collided frames using neural networks. It can support carrier sense across different LPWAN protocols. However, in the emergence of packet collisions, DeepSense only identifies the existence of each frame, without recovering the encoded data bits.

The most related works to ours are Choir [11], mLoRa [10], and FTrack [12]. Choir [11] exploits the hardware imperfection of low-cost LoRa devices to decompose overlapped signals. However, as demonstrated in [34], this approach does not scale well as the tiny frequency offset is difficult to extract under low SNR. More recently, mLoRa [10] and FTrack [12] exploit the misaligned edges of LoRa symbols to separate collisions. mLoRa [10] uses a collision-free chunk to boot up the decoding, and iteratively reproduces and extracts collided symbols based on the known pattern of LoRa chirps. FTrack [12] recovers collisions by detecting the edge of each LoRa symbol and then removing interference based on the continuity of each symbol's frequency. Both of the two approaches have fundamental limitations in processing low SNR signals.

VII. Conclusion

We present NScale, a novel protocol to resolve the low SNR LoRa packet collisions, whereby NScale utilizes the subtle packet time offset to decompose multiple collided packets. To deal with collisions with extremely low SNR, NScale translates the packet time offset, which is vulnerable to noise, to more robust frequency features through non-stationary signal scaling. We propose several novel techniques to address practical challenges in NScale design. To accurately measure frequency peaks, we propose a noise resistant iterative peak recovery algorithm to combat peak distortion in low SNR LoRa signal. Further, we remove the impact of the CFO and symbol-window time offset to decode each separated packet. We propose optimized design for diminishing the time costs of computation-intensive tasks to meet the real-time requirements of LoRa collision resolving. We implement NScale on USRP N210 and thoroughly evaluate its performance in both indoor and outdoor networks. NScale is completely implemented in software at the gateway, without requiring any modifications to the end nodes; thus we can apply NScale to current LoRa networks with little overhead. The evaluation results show that NScale improves the network throughput by **3.3**$\times$ for low SNR collided signals compared with other state-of-the-art methods.

References

[1] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 855–873, 2nd Quart., 2017.

[2] N. A. A. Ali and N. A. A. Latiff, "Environmental monitoring system based on LoRa technology in island," in *Proc. IEEE Int. Conf. Signals Syst. (ICSigSys)*, Bandung, Indonesia, Jul. 2019, pp. 160–166.

[3] J. G. Panicker, M. Azman, and R. Kashyap, "A LoRa wireless mesh network for wide-area animal tracking," in *Proc. IEEE Int. Conf. Electr., Comput. Commun. Technol. (ICECCT)*, Coimbatore, India, Feb. 2019, pp. 1–5.

[4] L. Chen *et al.*, "WideSee: Towards wide-area contactless wireless sensing," in *Proc. 17th Conf. Embedded Netw. Sens. Syst.*, New York, NY, USA, Nov. 2019, pp. 258–270.

[5] R. Jedermann, M. Borysov, N. Hartgenbusch, S. Jaeger, M. Sellwig, and W. Lang, "Testing LoRa for food applications—Example application for airflow measurements inside cooled warehouses with apples," *Proc. Manuf.*, vol. 24, no. 5, pp. 284–289, Feb. 2018.

[6] B. Ghena, J. Adkins, L. Shangguan, K. Jamieson, P. Levis, and P. Dutta, "Challenge: Unlicensed LPWANs are not yet the path to ubiquitous connectivity," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw.*, Los Cabos, Mexico, Oct. 2019, pp. 1–12.

[7] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa low-power wide-area networks scale?" in *Proc. ACM MSWiM*, Malta, U.K., Nov. 2016, pp. 59–67.

[8] 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. Sen. Netw.*, vol. 15, no. 2, pp. 1–35, Feb. 2019.

[9] A. Gamage, J. C. Liando, C. Gu, R. Tan, and M. Li, "LMAC: Efficient carrier-sense multiple access for LoRa," in *Proc. 26th Annu. Int. Conf. Mobile Comput. Netw.*, London, U.K., Sep. 2020, pp. 1–13.

[10] X. Wang, L. Kong, L. He, and G. Chen, "MLoRa: A multi-packet reception protocol in LoRa networks," in *Proc. IEEE 27th Int. Conf. Netw. Protocols (ICNP)*, Chicago, IL, USA, Oct. 2019, pp. 1–11.

[11] R. Eletreby, D. Zhang, S. Kumar, and O. Yağan, "Empowering low-power wide area networks in urban settings," in *Proc. Conf. ACM Special Interest Group Data Commun.*, Los Angeles, CA, USA, Aug. 2017, pp. 309–321.

[12] X. Xia, Y. Zheng, and T. Gu, "FTrack: Parallel decoding for LoRa transmissions," in *Proc. 17th Conf. Embedded Netw. Sensor Syst.*, New York, NY, USA, Nov. 2019, pp. 2573–2586.

[13] A. Berni and W. Gregg, "On the utility of chirp modulation for digital signaling," *IEEE Trans. Commun.*, vol. COM-21, no. 6, pp. 748–751, Jun. 1973.

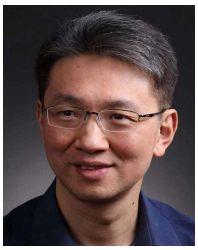
[14] V. Talla, M. Hessar, B. Kellogg, A. Najafi, J. R. Smith, and A. Gollakota, "LoRa backscatter: Enabling the vision of ubiquitous connectivity," in *Proc. ACM Ubicomp*, Honolulu, HI, USA, Sep. 2017, pp. 1–24.

[15] A. Dongare *et al.*, "Charm: Exploiting geographical diversity through coherent combining in low-power wide-area networks," in *Proc. 17th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, Porto, Portugal, Apr. 2018, pp. 60–71.

[16] 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. 1–18, Sep. 2016.
[17] B. Reynders and S. Pollin, "Chirp spread spectrum as a modulation technique for long range communication," in *Proc. Symp. Commun. Veh. Technol. (SCVT)*, Mons, Belgium, Nov. 2016, pp. 1–5.
[18] S. Demetri, M. Zúñiga, G. P. Picco, F. Kuipers, L. Bruzzone, and T. Telkamp, "Automated estimation of link quality for LoRa: A remote sensing approach," in *Proc. 18th Int. Conf. Inf. Process. Sensor Netw.*, Montreal, QC, Canada, Apr. 2019, pp. 145–156.
[19] O. Abari, D. Vasisht, D. Katabi, and A. Chandrakasan, "Caraoke: An e-toll transponder network for smart cities," in *Proc. ACM Conf. Special Interest Group Data Commun.*, London, U.K., Aug. 2015, pp. 297–310.
[20] H. Nyquist, "Certain topics in telegraph transmission theory," *Trans. Amer. Inst. Elect. Eng.*, vol. 47, no. 2, pp. 617–644, Apr. 1928.
[21] C. Gu, L. Jiang, R. Tan, M. Li, and J. Huang, "Attack-aware synchronization-free data timestamping in LoRaWAN," *ACM Trans. Sensor Netw.*, vol. 18, no. 1, pp. 1–31, Feb. 2022.
[22] S. Saruwatari, "GNU radio," *J. Inst. Image Inf. Telev. Eng.*, vol. 65, no. 8, pp. 1186–1189, 2011. [Online]. Available: http://gnuradio.org
[23] SEMTCH. *Sx1276/77/78/79 Datasheet*. Accessed: Dec. 3, 2021. [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-core/sx1278#download-resources
[24] S. Gollakota, F. Adib, D. Katabi, and S. Seshan, "Clearing the RF smog: Making 802.11 robust to cross-technology interference," in *Proc. ACM SIGCOMM*, Toronto, IL, Canada, Aug. 2011, pp. 170–181.
[25] E. Hamed, H. Rahul, M. A. Abdelghany, and D. Katabi, "Real-time distributed MIMO systems," in *Proc. ACM SIGCOMM Conf.*, Florianopolis, Brazil, Aug. 2016, pp. 412–425.
[26] E. Hamed, H. Rahul, and B. Partov, "Chorus: Truly distributed distributed-MIMO," in *Proc. Conf. ACM Special Interest Group Data Commun.*, Budapest, Hungary, Aug. 2018, pp. 461–475.
[27] H. Rahul, S. Kumar, and D. Katabi, "JMB: Scaling wireless capacity with user demands," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, Helsinki, Finland, Oct. 2012, pp. 235–246.
[28] J. Li, Y. Zhang, M. Shi, Q. Liu, and Y. Chen, "Collision avoidance strategy supported by LTE-V-based vehicle automation and communication systems for car following," *Tsinghua Sci. Technol.*, vol. 25, no. 1, pp. 127–139, Feb. 2020.
[29] S. Sen, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi, "Successive interference cancellation: Carving out MAC layer opportunities," *IEEE Trans. Mobile Comput.*, vol. 12, no. 2, pp. 346–357, Feb. 2013.
[30] D. Halperin, T. Anderson, and D. Wetherall, "Taking the sting out of carrier sense: Interference cancellation for wireless LANs," in *Proc. 14th ACM Int. Conf. Mobile Comput. Netw. (MobiCom)*, San Francisco, CA, USA, Sep. 2008, pp. 339–350.
[31] P. Patel and J. Holtzman, "Analysis of a simple successive interference cancellation scheme in a DS/CDMA system," *IEEE J. Sel. Areas Commun.*, vol. 12, no. 5, pp. 796–807, Jun. 1994.
[32] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," in *Proc. ACM SIGCOMM Conf. Data Commun. (SIGCOMM)*, Seattle, WA, USA, Aug. 2008, pp. 159–170.
[33] L. Kong and X. Liu, "mZig: Enabling multi-packet reception in ZigBee," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, Paris, France, Sep. 2015, pp. 552–565.
[34] M. Hessar, A. Najafi, and S. Gollakota, "NetScatter: Enabling large-scale backscatter networks," in *Proc. USENIX NSDI*, Boston, MA, USA, Feb. 2019, pp. 271–284.
[35] J. Chan, A. Wang, A. Krishnamurthy, and S. Gollakota, "DeepSense: Enabling carrier sense in low-power wide area networks using deep learning," 2019, *arXiv:1904.10607*.

Shuai Tong (Student Member, IEEE) received the B.E. degree from the College of Computer Science, Nankai University, in 2019. He is currently pursuing the Ph.D. degree with the School of Software, Tsinghua University. His research areas mainly include low-power wide-area networks, the Internet of Things, and mobile computing.

Jiliang Wang (Senior Member, IEEE) received the B.E. degree in computer science and technology from the University of Science and Technology of China and the Ph.D. degree in computer science and engineering from The Hong Kong University of Science and Technology. He is currently an Associate Professor with the School of Software, Tsinghua University. His research interests include wireless and sensor networks, the Internet of Things, and mobile computing.



Yunhao Liu (Fellow, IEEE) received the B.S. degree from the Department of Automation, Tsinghua University, and the M.S. and Ph.D. degrees in computer science and engineering from Michigan State University. He is currently a Professor with Tsinghua University. His research interests include sensor networks and the IoT, localization, RFID, distributed systems, and cloud computing. He is a Fellow of ACM.