CoLoRa: Enabling Multi-Packet Reception in LoRa

Shuai Tong, Zhenqiang Xu and Jiliang Wang
School of Software and BNRist,
Tsinghua University, P.R. China
{tl19,xu-zq17}@mails.tsinghua.edu.cn, jiliangwang@tsinghua.edu.cn

***Abstract*—LoRa, more generically Low-Power Wide Area Network (LPWAN), is a promising platform to connect Internet of Things. It enables low-cost low-power communication at a few kbps over upto tens of kilometers with a 10-year battery lifetime. However, practical LPWAN deployments suffer from collisions, given the dense deployment of devices and wide coverage area. We propose CoLoRa, a protocol to decompose large numbers of concurrent transmissions from one collision in LoRa networks. At the heart of CoLoRa, we utilize packet time offset to disentangle collided packets. CoLoRa incorporates several novel techniques to address practical challenges. (1) We translate time offset, which is difficult to measure, to frequency features that can be reliably measured. (2) We propose a method to cancel inter-packet interference and extract accurate feature from low SNR LoRa signal. (3) We address frequency shift incurred by CFO and time offset for LoRa decoding. We implement CoLoRa on USRP N210 and evaluate its performance in both indoor and outdoor networks. CoLoRa is implemented in software at the base station and it can work for COTS LoRa nodes. The evaluation results show that CoLoRa improves the network throughput by 3.4× compared with Choir and by 14× compared with LoRaWAN.**

I. Introduction

The success of the Internet of Things (IoTs) highly depends on connecting large scale IoT devices. As a promising communication platform, LPWAN can provide low-cost long-range communication with very low energy consumption for large scale IoT devices [1]. Long Range (LoRa) is a widely used and industry applied LPWAN technique, which works on the unlicensed sub-GHz ISM band, e.g., 475 MHz or 900 MHz bands [2]. Typically, end nodes of a LoRa network can achieve a communication range of kilometers or even tens of kilometers at few kbps. The energy consumption of LoRa end nodes is very low and a node can work for nearly 10 years powered by a button cell battery.

Deploying LoRa networks in practice, however, is very challenging. Each base station in LoRa covers a wide area and is expected to connect to a large number of devices [3]. This introduces severe signal collisions when multiple end nodes transmit packets to the base station concurrently, which decreases the network throughput and also ratchets up the energy consumption and network delay. Moreover, the limited energy budget and low-cost hardware make it difficult to apply sophisticated MAC protocol to resolve collisions [4], [5]. The relative long packet duration further accelerates the collision problem. This leads to a gap between LoRa's vision to provide low-power large-scale connections, and its practical capability [6].

There exist many concurrent decoding approaches in traditional wireless [7], [8]. As a representative one, ZigZag [7] decodes collided Wi-Fi packets but it requires m retransmissions to resolve an m-packet collision. mZig [8] decodes multiple ZigBee packets from a collision by leveraging ZigBee's coding characteristics. Recently, NetScatter [9] proposes a new encoding and decoding technique for multiple backscattered and synchronized chirp signals, while it cannot work for unsynchronized transmission in LoRa networks. Choir [10] shows that the hardware imperfection of low-cost LoRa end nodes will cause frequency offsets in LoRa symbols. It utilizes the frequency offsets to distinguish collided LoRa packets. However, it is difficult to extract accurate the tiny frequency offset especially for low SNR LoRa signal with inter-packet interference. Thus, the experiment results in [10] show a limited concurrency. Meanwhile, we also observe that the frequency offset is not stable for some low-cost LoRa nodes, which diminishes its benefit in practice.

Our Approach. To resolve collisions, we proposed CoLoRa, a protocol that enables Multi-Packet Reception (MPR) in LoRa. CoLoRa utilizes the packet time offsets to decompose multiple concurrent transmissions from one collision directly.

To see how CoLoRa works, consider the scenario in Fig. 1, where two packets collide. Both packet-A and packet-B consist of multiple chirp symbols. LoRa modulates signals with chirp spread spectrum (CSS) technique, where chirps with different frequency shifts encode different data bits. The LoRa decoding algorithm multiplies the received chirp with a standard down-chirp. When there is no collision, the result is a single tone which translates to a single peak in the frequency domain. Otherwise, there will be multiple frequency peaks, as shown in Fig. 1. The data bits of the two collided packets are mixed up and hence the decoding algorithm fails to decode any of them.

At the heart of CoLoRa is a physical layer algorithm that utilizes the packet time offset to disentangle collisions. Upon receiving a collision, CoLoRa first cut the received signal into a series of reception windows, each with length equal to a chirp. As shown in Fig. 1, when choosing a misaligned window, each chirp is divided into two segments by two consecutive windows. Then for signal in each window, we transform the incomplete chirp segments to frequency peaks by multiplying a down-chirp and applying a Fourier transform. We formally prove that the height of the peak is proportional to the length of the segment in Section III. We also show that the two peaks of the same chirp have the same frequency, e.g.,

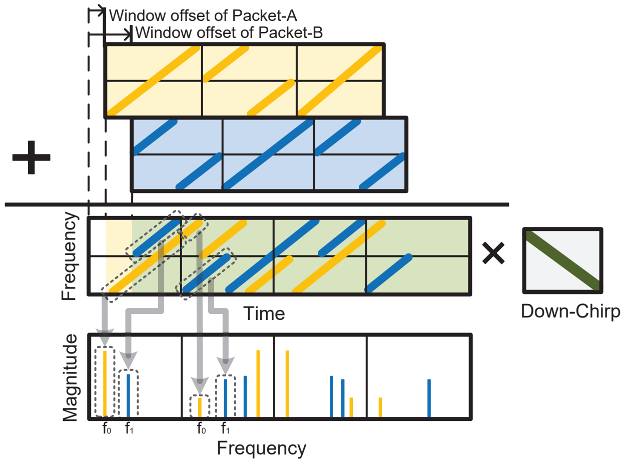


Fig. 1: An example of decomposing a two-packet collision with packet time offset. CoLoRa decomposes collided packets by transforming the packet time offset to frequency domain features.

f_0 for packet-A's first chirp. For two peaks belonging to the same chirp, define the peak ratio as the height of the latter one divided by that of the former one. We can see that peak ratio is identical for chirps of the same packet, while it is distinct for chirps of different packets. Thus, by grouping chirps with the same peak ratios, CoLoRa can finally disentangle the collided packets.

The benefits of using peak ratio as features to separate collided packets are three-fold. (1) It concentrates energy for time-domain features to signal peaks in the frequency domain and thus it works well for low SNR LoRa signal with inter-chirp interference. (2) It is resilient to signal dynamics and environment complications, as the peak ratio is determined by the packet time offset which is stable during the whole transmission. (3) There is no error accumulation as the peak ratio of each chirp is calculated independently.

Challenges. Using peak ratio to disentangle packets in CoLoRa also faces practical challenges: First, we find that the selection of reception windows affects the peak estimation. For example, an improper reception window selection may result in two peaks of unbalanced height, where the shorter peak is easily distorted or even be masked in noise. We propose an interleaved window selection strategy which can achieve a bounded division ratio in $[\frac{1}{3}, 3]$. Second, we find it is difficult to obtain an accurate peak estimation as the inter-peak interference leads to peak distortions. We propose an iterative peak recovery algorithm where the highest peak component is iteratively estimated, recovered and extracted. Thus, we can eliminate the inter-peak interference as well as solving the near-far problem. Third, after peak grouping, we find the packet decoding is severely impeded by the mixed impact of Central Frequency Offset (CFO) and reception window time offset. Based on the structure of LoRa preamble, we design a technique to estimate and compensate the CFO and window time offset for LoRa decoding.

Main results and contributions. The main results and contributions of this paper are as follows:

- We proposed CoLoRa, a protocol to decompose multiple concurrent transmissions from one collision in LoRa, to bridge the gap between LoRa's vision of providing low-power long-distance connections to large scale IoT devices, and its practical capability.
- We address practical challenges in CoLoRa design. The performance of CoLoRa highly relies on accurate information of peaks. We propose an efficient reception window selection strategy to generate balanced peaks. We design an iterative peak recovery algorithm to address inter-peak interference and recover peak information accurately. Finally, we remove the impact of CFO and time offset to accurately decode packets.
- We implement CoLoRa on USRP N210 and thoroughly evaluate its performance in different scenarios. CoLoRa is completely implemented in software at the base station without requiring any modifications at the end nodes. The experiments results show that CoLoRa can improve the network throughput by **3.4**× compared with Choir and **14**× compared with LoRaWAN.

II. A Primer on LoRa

LoRa physical layer employs the chirp spread spectrum (CSS) technique to modulate signals. CSS modulates signals into chirps of linearly increasing/decreasing frequency, i.e., up-chirps and down-chirps, making the signal occupying the entire spectral band. Chirp symbols are inherently robust against in-band interference and other channel degradations, and hence they can be detected and decoded even under extremely low SNR, which makes low-power and long-range communication be possible for LoRa end nodes.

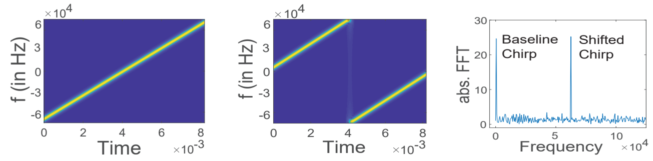
LoRa modulates data bits by cyclically shifting the baseline up-chirp. As shown in Fig. 2(a), the frequency of the baseline up-chirp increases linearly from $-\frac{BW}{2}$ to $\frac{BW}{2}$, and the length of the chirp is T. Thus, the frequency of baseline up-chirp can be represented as $kt - \frac{BW}{2}$, where $k = \frac{BW}{T}$ is the gradient of frequency sweeping. And the baseline up-chirp $C(t)$ can be represented as

$$C(t) = e^{j2\pi(-\frac{BW}{2}+\frac{kt}{2})t}$$

Given the frequency shift f of baseline up-chirp, the resulted symbol is $C(t)e^{j2\pi ft}$. Then given the spectrum bandwidth BW, after the shift, all the frequencies higher than $\frac{BW}{2}$ will be aligned down to $-\frac{BW}{2}$ as shown in Fig. 2(b). LoRa defines N different shifted frequencies, which results in N uniformly shaped up-chirps to encode $SF = log_2 N$ bits.

To demodulate, LoRa leverages the baseline down-chirp, i.e., the conjugate of baseline up-chirp $C^*(t)$. As shown in Fig. 2(c), by multiplying a baseline down-chirp, each shifted up-chirp is despreaded, and the result is calculated as

$$C^*(t) \times C(t)e^{j2\pi ft} = e^{j2\pi ft}$$



(a) (b) (c)

Fig. 2: LoRa Physical Layer: (a) Spectrogram of a baseline chirp symbol. (b) Spectrogram of a shifted chirp symbol. (c) The demodulation result of the base chirp symbol and the shifted chirp symbol.

i.e., a single tone at frequency f. The encoded data bits are then recovered by searching energy peaks from the result of the Fourier transformation.

III. Collision Separation Basics

In this section, we show the basics of how to separate packets from collisions in LoRa. Before explaining our collision separation strategy, first recall how the conventional LoRa receiver works. When a packet arrives without collision, the LoRa receiver first synchronizes itself with the received packet. As is shown in Fig. 3(a), after synchronization, each chirp symbol is aligned with the reception window. Then the receiver multiplies a down-chirp in each window and applies FFT to transform each chirp to a single peak, whose frequency represents the encoded data of the corresponding chirp.

When there is a collision, we also use the reception windows to cut the received signal. Note here we select the reception window not aligned to the packets. Thus, each chirp is divided into two segments in two consecutive reception windows, as shown in Fig. 1. Now, we analyze the result of the multiplication and show how to use this for packet separation.

In LoRa, an encode chirp symbol is an up-chirp with a frequency shift, i.e., $x(t) = He^{j2\pi ft}C(t)$, where H is the signal amplitude and f is the shifted frequency to encode bits. Denote τ is the chirp-level offset between the packet start and the reception window in Fig. 3(b). For a chirp, the first chirp segment $x_1(t)$ can be written as

$$x_1(t) = x(t-\tau) = He^{j2\pi f(t-\tau)}C(t-\tau) \qquad \tau \le t < T \quad (1)$$

As time shift can be translated to frequency shift, we have $C(t-\tau) = e^{-j2\pi k\tau}C(t)$. Thus, Eq. (1) can be rewritten as

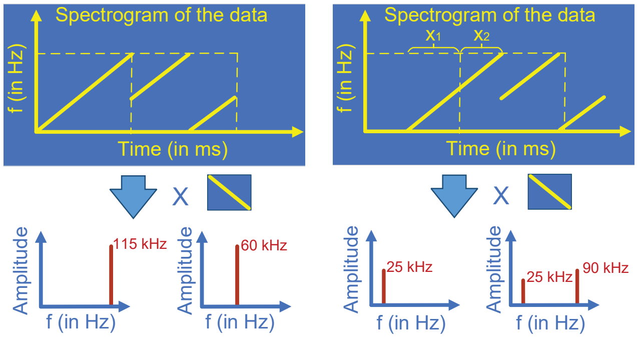
$$x_1(t) = He^{j2\pi(-f\tau+(f-k\tau)t)}C(t) \qquad \tau \le t < T \quad (2)$$

Similarly, the second chirp segment $x_2(t)$ can be written as

$$\begin{aligned} x_2(t) &= x(t-(T-\tau)) \\ &= He^{j2\pi(-f(T-\tau)+(f-k(T-\tau))t)}C(t) \quad 0 < t < \tau \end{aligned} \quad (3)$$

We multiply the signal in each reception window with a baseline down-chirp, despreading each chirp segment to a single tone. Mathematically, for $x_1(t)$ in the first reception window, after multiplying with the down-chirp, we have

$$\begin{aligned} \hat{x}_1(t) &= He^{j2\pi(-f\tau+(f-k\tau)t)}C(t)\cdot C^*(t) \\ &= He^{j2\pi(-f\tau+(f-k\tau)t)} \end{aligned} \quad (4)$$



(a) (b)

Fig. 3: (a) Conventional receiver uses aligned windows and demodulates chirp by chirp. (b) CoLoRa receiver uses unaligned windows and demodulates window by window.

which is a single tone at the frequency of $f_1 = f - k\tau$. $x_2(t)$ in the second window is also despreaded to a single tone, with the frequency of $f_2 = f - k(T-\tau)$. When using a sample rate equal to the bandwidth (i.e., BW), the two despreaded signal $\hat{x}_1(t)$ and $\hat{x}_2(t)$ share the same frequency in the spectrum, i.e., $f_1 = f_2 = f - k\tau$. Therefore, as shown in Fig. 3(b), two segments of the same chirp are transformed to two peaks located at the same FFT bin of $f - k\tau$.

Thus, for the first segment $x_1(t)$, the height of its peak is

$$h_1 = \sum_{n=0}^{N-1} \hat{x}_1[n]e^{-j2\pi(f-k\tau)\frac{nT}{N}} \quad (5)$$

where $\hat{x}_1[n]$ is nth sampling point of $\hat{x}_1(t)$ and N is the total sample points of a chirp. Substituting $\hat{x}_1[n]$ with the Eq. (4), we have

$$h_1 = Hf_s(T-\tau). \quad (6)$$

Similarly, for the second segment $x_2(t)$, the peak height is

$$h_2 = Hf_s\tau. \quad (7)$$

For each chirp, we define peak ratio P as

$$P = \frac{h_2}{h_1} = \frac{\tau}{T-\tau}. \quad (8)$$

It can be seen that the peak ratio is determined by the window offset τ. Thus, the peak ratio is identical for all chirps of the same packet. Meanwhile, we can also see from Eq. (6) and Eq. (7) that based on the peak height h_1 (or h_2), we can calculate the amplitude H of the chirp segment.

Summary. Through the analysis, we have the following results.

- The peaks of two segments of the same chirp are located at the same frequency, with height proportional to segment length.

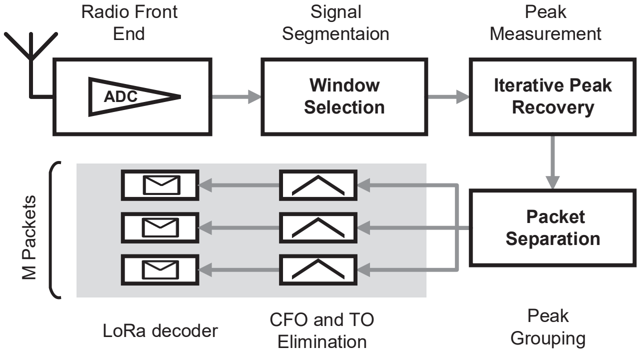


Fig. 4: The main workflow of CoLoRa.

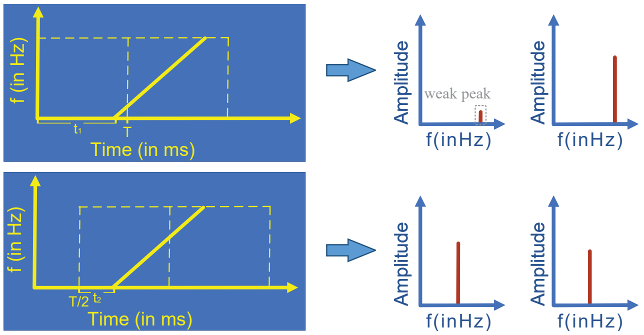


Fig. 5: Examples of interleaved windows for solving unbalanced window division

- The peak ratio is determined by the window offset, i.e., $P = \frac{\tau}{T-\tau}$, and thus is identical for all chirps of the same packet.
- The peak ratio is different for packets with different arrival time.
- Based on the peak height, we can calculate the amplitude of the chirp segment.

Briefly, CoLoRa leverages the peak ratio to distinguish packets and then use peak information to facilitate decoding. In the next section, we will detail CoLoRa's design to address practical challenges for separating packets and achieving multi-packet reception.

IV. CoLoRa Design

A. Overview

Fig. 4 shows the workflow of CoLoRa design. CoLoRa at the base station mainly consists of four components. (1) For a received signal, CoLoRa first selects reception windows to divide the received signal into segments with length equal to a chirp. (2) For each reception window, CoLoRa transforms the low SNR signals into robust FFT peaks and then accurately recover the features of peaks in the presence of noise and inter-peak interference. (3) Based on the estimation of FFT peaks, CoLoRa clusters the peaks into multiple groups where each group contains the peaks of the same packet. (4) Finally, CoLoRa decodes each group of peaks while addressing the challenge of Central Frequency Offset (CFO) and packet time offset. We show the details of each component and how to address practical challenges.

B. Reception Window Selection

Upon receiving a signal, CoLoRa needs to cut the signal into continuous reception windows each with length equal to a chirp. A practical challenge is that the selection of reception windows impacts the decoding performance. For example, for the collision cases, an improper selection of reception windows will result in an unbalanced division, where chirps are divided into very short segments corresponding to very low peaks, which are easily distorted or even masked in noise as shown at the top of Fig. 5. While for the collision-free cases, the misaligned windows will hinder the chirps to concentrate the whole symbol's energy and further diminish its ability to combat the noise. Thus, the selection of reception windows should satisfy the requirements for both the collision cases and the collision-free cases.

We propose an interleaved reception window selection strategy to achieve balanced division in collision cases and aligned windows in collision-free cases. To achieve this goal, we first detect the onset time of the received signal. We use an Akaike Information Criterion (AIC) [11] based algorithm to detect the onset time of the received LoRa packet, which is proved to work well even under the noise floor [12]. Based on the detection result, we select reception windows W_1 with window beginning aligned with the detected onset time. Then we use W_1 to decode the received signals and estimate the peaks for every reception window. If only one peak is detected in each window, it indicates that there is no collision and then the signal is decoded like in conventional LoRa. Otherwise, we select a new interleaved reception window W_2 by moving the start of W_1 by $\frac{T}{2}$, where T is the chirp length.

Here, we show that given any packet involved in the collision, at least one of those two interleaved reception windows (i.e., W_1 and W_2) can give a balanced division with a bounded division ratio in $[\frac{1}{3}, 3]$. As shown in Fig. 5, the time offset between the starts of the two reception windows W_1 and W_2 is $\frac{T}{2}$. Assume the chirp-level time offset between pkt and W_1 is t_1, and the offset between pkt and W_2 is t_2. Without loss of generality, assume $t_1 > \frac{T}{2}$. Thus, we have $t_2 = t_1 - \frac{T}{2}$. We show that either W_1 or W_2 should divide pkt into chirp segments with division ratio in $[\frac{1}{3}, 3]$.

We can calculate the shorter chirp segment by W_1 as $D_1 = min(t_1, T - t_1)) \leq \frac{T}{2}$. The shorter segment by W_2 is $D_2 = min(t_2, T - t_2))$. As $t_2 = t_1 - \frac{T}{2}$, we have $D_2 = \frac{T}{2} - D_1$. We choose the more balanced one, i.e.,

$$D = max(D_1, D_2) = max(D_1, \frac{T}{2} - D_1)$$

where D_1 is smaller than $\frac{T}{2}$. Thus, we can prove $D \geq \frac{T}{4}$. Similarly, we can prove $D \leq \frac{3T}{4}$. Thus, the division ratio, resulted from either W_1 or W_2, should be in the range of $[\frac{1}{3}, 3]$.

C. Iterative Peak Recovery

In each reception window, we transform the signal to peaks by multiplying baseline down-chirps and applying FFT. Then we need to estimate the accurate frequency and height for each peak in order to separate packets. While in practice, height estimation for peaks is prone to inter-chirp interference. Revisiting the progress of LoRa decoding, we multiply a down-chirp and apply FFT for signals in each reception window. We can observe that there are periodical sidelobes located around each main peak after FFT, a property that stems from the time limited input sequence. The sidelobes affect the peak estimation from two aspects. On one hand, sidelobes distort peaks and affect accurate measurements of the peak height and frequency. On the other hand, low height peaks tend to be masked by sidelobes of other strong peaks.

We propose an iterative peak recovery algorithm. The pseudocode for iterative peak recovery algorithm is shown in Algo. 1. We first find the highest peak from the reception window. The benefit of using the highest peak is two-fold. First, the relative distortion of the highest peak is smaller than the other peaks. Second, using the highest peak avoids incorrectly using sidelobes as peaks since the highest peak normally cannot be any sidelobe. By measuring the frequency and height of the highest peak, we can obtain a coarse estimation of the real peak. Assume the highest peak has a height of h_0, a center frequency f_0 and a phase of ϕ_0. To obtain a coarse estimation of chirp segment, we need further to obtain the position, amplitude and length of the chirp segment.

Position estimation. First we need to determine the position of the chirp segment, i.e., whether the segment is adjacent to the former window or the latter window. We use the signal from both the former and the latter windows to help determine the chirp position. Denote the signal of the current window is x_0, and the signal of the former and the latter windows are x_1 and x_2. If we combine the signal of each two adjacent windows together, we can get $y_1 = [x_1, x_0]$ and $y_2 = [x_0, x_2]$, each with the length of two chirps. Then, we multiply two continuous baseline down-chirp with y_1 and y_2, respectively. After FFT on the multiplication, we obtain two peaks h_1 and h_2 corresponding to y_1 and y_2, both located at f_0. If the chirp segment is adjacent to the latter window, the height of h_2 should be higher than that of h_1, and vice verse.

Amplitude and length estimation. We estimate the amplitude of the chirp segment based on y_1 and y_2. Assume h_2 is the height of the higher peak, as a chirp symbol can only span two windows, y_2 should contain the complete chirp symbol. As the peak height is proportional to the length of segment, we have $\frac{h_2}{h_0} = \frac{T}{L}$. Thus, we can estimate the length of chirp segment in x_0 as $L = \frac{h_0}{h_2}T$. According to Eq. (7), we can calculate the amplitude of the chirp segment as $H = \frac{h_0}{f_s L}$, where f_s is the sampling frequency.

Accurate peak recovery. Based on the frequency f_0, phase ϕ_0, length L and amplitude H, we can reconstruct the initial chirp segment as

$$\widetilde{s}(t) = He^{j2\pi ft+\phi_0}C(t) \tag{9}$$

where $t \in [0, L]$ if the segment is adjacent to the previous window, or $t \in [T - L, T]$ if the segment is adjacent to the following window. The chirp segment $\widetilde{s}(t)$ is a coarse estimation of the real chirp segment as it may be distorted by sidelobes of other chirps. For a more accurate estimation of the chirp segment, after canceling it from the signal, we will obtain less residual energy in the remaining signal. Thus, we search for different chirp segments in the near space of the initial chirp segment in terms of phase, amplitude, start frequency and length. For each chirp segment, we cancel it from the original signal and calculate residual signal in the frequency domain by summing up the energy of the FFT outputs. The goal of the search is to find the optimal chirp segment S that minimizes the residual energy.

Algorithm 1: Iterative Peak Recovery

```
Input: Signal in a reception window: $Sig$
Output: Height and frequency of peaks: $[H, F]$
$DemodSig$ = $Sig \otimes DownChirp$;
while SUM(FFT( $DemodSig$)) $> threshold$ do
    $[f_0, \phi_0, h_0]$ = HIGHESTPEAK(FFT($DemodSig$));
    $[Loc, h]$ = SEGMENTLOCATION();
    $L = \frac{Th_0}{h}$;
    $H = \frac{h_0}{f_s L}$;
    $\tilde{s}$ = INITIALCHIRPSEG($f_0, \phi_0, H, L, Loc$);
    $S$ = ITERATIVEREFINE($\tilde{s}$);
    $[H_i, F_i]$ = PEAKMEASURE(FFT($S \otimes DownChirp$));
    CANCEL $S$ FROM $Sig$;
    $DemodSig$ = $Sig \otimes DownChirp$;
end
return $[H, F]$;
```

Then we obtain an accurate peak by multiplying the optimal chirp segment with a baseline down-chirp. Moreover, by canceling the optimal chirp segment, we also cancel its interference with other peaks. We iteratively recover accurate peaks in the remaining signal until the residual energy is lower than a threshold. By using our iterative peak recovery algorithm, we can also address the near-far problem where a strong signal from a near source interferes with a weaker signal from a further source.

D. Packet Separation and Decoding

Till now, accurate peaks are recovered. We show how to use this information for separating and decoding packets. Recall that peaks of two segments from the same chirp are located at the same frequency. We can pair these two peaks by matching peaks of the same frequency in two consecutive windows. For each pair of peaks, we calculate the peak ratio, i.e., the height of the latter peak divided by the height of the former peak. Note that the peak ratio is identical for chirp symbols of the same packet but different from packet to packet.We use a k-means approach to group the peaks into k different clusters, each of which corresponds to a collided packet.

To decode for each group of peaks, we need to address the peak frequency bias introduced by CFO and window time offset. Both CFO and window time offset will result in chirp decoding errors as they introduce peak frequency

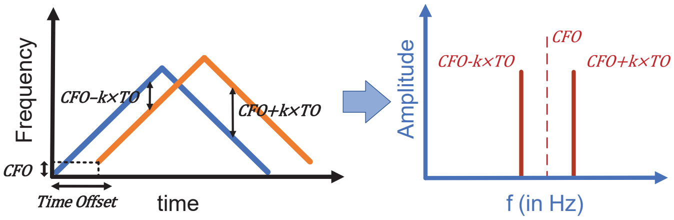


Fig. 6: Examples of CFO and Time Offset Estimation.

biases. Normally, we can measure the frequency shift of preambles to obtain the CFO. The challenge is that window time offset between the chirp and reception window also results in the frequency shift. We utilize the structure of the LoRa packet to resolve this challenge. In a LoRa packet, the preambles are baseline up-chirps and the SFD contains baseline down-chirps. As shown in Fig. 6, CFO will cause the same frequency shift to both the up-chirp and the down-chirp. When there is no window time offset, both baseline up-chirps and baseline down-chirps are transformed to peaks with zero frequency shift. Otherwise, each chirp is transformed to a peak with non-zero frequency shift proportional to the window time offset. Though both CFO and window time offset incur frequency shift, their impacts are different. CFO causes identical frequency shift for both the up-chirp and down-chirp while window time offset causes opposite frequency shifts for the up-chirp and down-chirp. Denote $TO = \tau$ is the window time offset, for the baseline up-chirp, the total frequency shift can be calculated as

$$\delta f_{up} = -\tau k + CFO \tag{10}$$

where $-\tau k$ is the frequency shift caused by the window time offset. Similarly, for the baseline down-chirp, the frequency shift is

$$\delta f_{down} = \tau k + CFO.$$

We calculate δf_{up} and δf_{down} by multiplying the preamble with down-chirps and the SFD with up-chirps. Then we can estimate the CFO as

$$CFO = \frac{\delta f_{up} + \delta f_{down}}{2}.$$

Meanwhile, by substituting CFO into Eq. (10), we can also obtain the estimation of the window time offset τ.

We compensate for each peak with the calculated CFO and window time offset. Finally, we can decode a packet with a group of peaks using a standard LoRa decoder.

V. Implementation

*a) **Hardware**:* We implement CoLoRa base station on USRP N210 software radios with a UBX daughterboard. The base station uses a single antenna with 2 dBi gain and can receive signals at 470 MHz and 900 MHz bands for LoRa. The implementation supports 125 kHz, 250 kHz and 500 kHz bandwidth for each channel. CoLoRa decoding algorithm is independent of the hardware platform, and it can also be implemented on other commercial LoRa base stations as long as the physical samplings can be obtained. CoLoRa does not have special requirement on end nodes and can work with any of existing LoRa nodes. In our implementation and experiment, we use the LoRa node with an SX1278 radio chip and a single antenna.

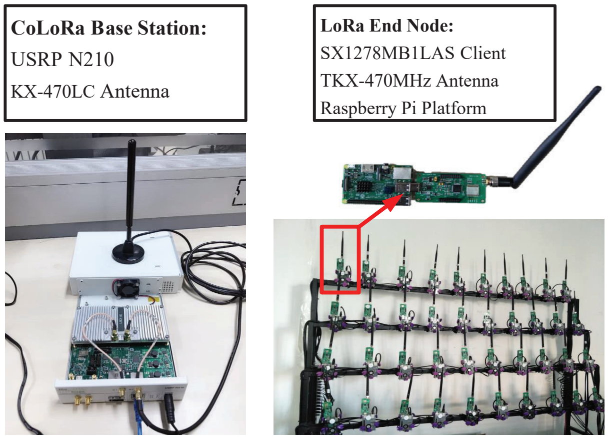


Fig. 7: CoLoRa base station on USRP N210 and LoRaNet testbed.

*b) **Software**:* We use the UHD+GNU-Radio library [13] for developing our own LoRa demodulator, and implement CoLoRa in MATLAB to process PHY samples. The functionality of CoLoRa is to decompose an m-packet collision into m sequences of collision-free symbols and then translate them into m packets.

VI. Evaluation

A. Methodology

*a) **Scenario**:* We evaluate the performance of CoLoRa in two different scenarios.

- LoRa testbed (*LoRaNet*) which consists of 40 LoRa end nodes as shown in Fig. 7. Each end node consists of an SX1278 radio chip, working at the frequency of 470 MHz and placed at a fixed position of a shelf. All the LoRa nodes are connected to a backbone network through the Raspberry Pis and thus information from them can be efficiently collected.
- Outdoor real LoRa network, where 20 LoRa temperature and humidity sensors are placed at different locations of the campus such as buildings, roads and parking lots as shown in Fig. 8. Each sensor node can collect and transmit the data of temperature and humidity to the base station by LoRa packets. The nodes are distributed over a region of 0.3km by 0.5km.

*b) **Baseline**:* We compare the performance of CoLoRa with two different approaches.

- LoRaWAN [14]: The widely used standard LoRaWAN baseline using ALOHA.
- Choir [10]: A recent LoRa collision resolution approach that decouples collisions using hardware imperfection of

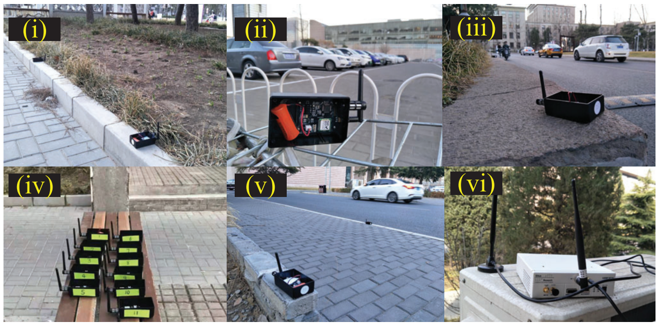


Fig. 8: The outdoor deployed LoRa network, each sensor node transmits temperature and humidity data by LoRa packets periodically.

end nodes. In practice, the hardware offsets of low-cost LoRa end nodes fluctuate according to the node type, packet length and SNR, as shown in Fig. 9. Thus, in our implementation, we use both the fractional FFT bin and the peak magnitude to match correct transmitters.

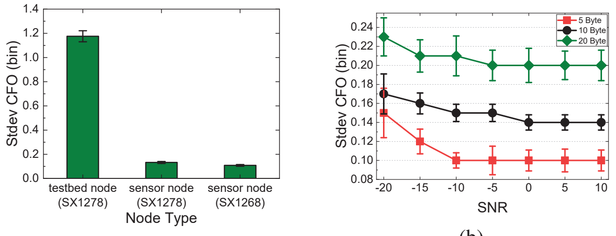
B. Experiment Result

*1) **Decoding Multi-Packet Collision:*** In this experiment, we examine CoLoRa's performance for separating multiple packets in collisions. As LoRaWAN cannot separate packets in collisions, their performance under collision is extremely low. Thus, we only show the performance of CoLoRa and Choir, which can separate multi-packet reception in LoRa.

We use the LoRaNet testbed to efficiently generate multi-packet collision in which we can control each collided nodes accurately. To produce a collision with M overlapped packets, we use a beacon to synchronize the transmission for M different end nodes.

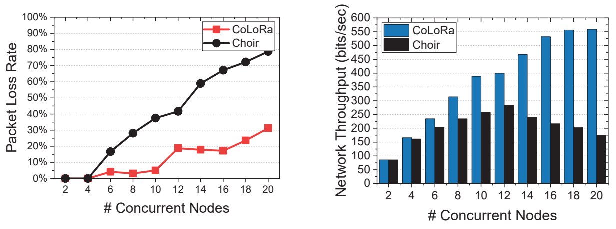
Upon receiving a beacon, all M end nodes wake up and transmit a LoRa packet. We allow a random processing delay for each end node. All packets are generated with a specific known sequence of bytes. At the base station, packets from the M end nodes are overlapped, leading to an M-packet collision. We produce collisions with a different number of overlapped packets by changing the number of involved end nodes M. At the base station, we use CoLoRa and Choir to decompose the collided packets. Then for each decomposed packet, we use a standard LoRa decoder to translate the chirp symbols into data bits. The packets sent by each end node is known in prior. Thus, we can verify the correctness of the decoded packets and calculate the Packet Loss Rate (PLR) and network throughput of all nodes in this experiment.

Fig. 10(a) shows the PLR for CoLoRa and Choir. As concurrent nodes increasing from 1 to 20, the PLRs of both the two approaches grow up. The PLR of CoLoRa increases much more slowly than that of Choir. This is because CoLoRa 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. This also coincides with the result in Choir [10] that it supports less than 6 concurrent transmissions and with more concurrent transmissions its performance degrades significantly. We further investigate the performance of CoLoRa and find that the packet loss usually happens when two packets have similar peak ratios causing peaks being clustered into a wrong group. We can also see that when the number of current nodes is less than 4, CoLoRa and Choir have similar performance. However, when the number of nodes increases, CoLoRa quickly outperforms Choir.



(a) (b)

Fig. 9: Characterizing Hardware Offsets: the root mean-squared error of the frequency offset within a packet for (a) different type of end nodes with the same configuration (SF12, packet length of 10 Byte); (b) different packet length and SNRs (sensor node with SX1278).



(a) (b)

Fig. 10: Decoding for different concurrent transmissions at a single-antenna USRP base station. (a) Packet Loss Rate (b) Network throughput.

We further send decomposed chirp symbols to a standard LoRa decoder for extracting the content of the packet. LoRa modulates the data bits into chirp symbols with an FEC code, hence some of the symbol errors can be corrected during the decoding. In our experiment, we initialize the testbed nodes with a coding rate of 4/8, where each four useful data bits are encoded eight bits along with FEC code. After decoding, we can get the overall network throughput as shown in Fig. 10(b). The network throughput of CoLoRa increases as the number of concurrent nodes increases from 2 to 20. This is due to the benefit of multi-packet reception in CoLoRa. Meanwhile, we can see that the network throughput of Choir is much lower than that of CoLoRa. The network throughput even starts to decrease when the number of concurrent nodes is larger than 12. This is because when the number of concurrent nodes is large, most packets are undecodable for Choir. When the number of nodes is 20, the network throughput of CoLoRa (552 bps) is about 3.4× of Choir (162 bps).

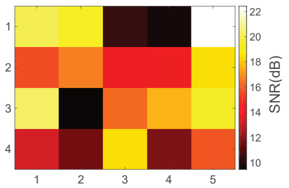


Fig. 11: SNR survey for LoRa nodes on the LoRaNet testbed.

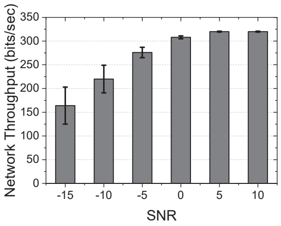
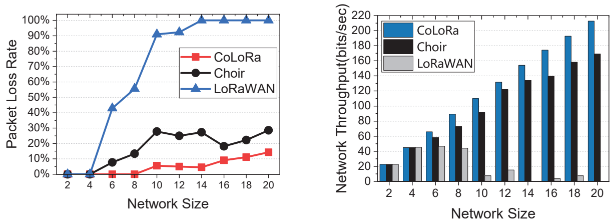


Fig. 12: CoLoRa's network throughput under different levels of SNR.



(a)

(b)

Fig. 13: Performance in real deployed low duty cycled networks: (a) Packet Loss Rate; (b) Network Throughput under different network size.

2) ***Impact of SNR:*** In this experiment, we show the impact of SNR on the performance of CoLoRa. High channel noise will cause peaks of chirp segments suffering from distortion, which further disturbs the calculation of peak ratios. To characterize the impact of channel noise, we use the testbed to produce packet collisions where each end node transmit a randomly chosen sequence of bits concurrently

We define the SNR of a collision signal as the SNR of its strongest signal component. Fig. 11 plots the SNR survey for signals from each testbed node. The SNRs of different nodes are diverse, but all above the noise floor. Thus, for precise SNR control and emulating low SNR scenario, we artificially add noise traces to the received collision signals. By controlling the magnitude of the added noise traces, we can achieve a certain SNR in dB defined as $10log_{10}\frac{A^2}{\mathbb{E}[Z_Q^2(t)+Z_I^2(t)]}$, where $z_Q(t)$ and $z_I(t)$ are the Q and I traces of the added Gaussian noise and A is the signal's amplitude.

Fig. 12 shows the Network Throughput for CoLoRa under different levels of SNR. CoLoRa can also work for the low SNR signal as it concentrates the energy of a chirp to a signal peak in the frequency domain. As shown in Fig. 12, the total throughput is stable when SNR is higher than -10dB and the performance slightly degrades for lower SNR scenario.

3) ***Addressing Collision in Real LoRa Networks:*** In this experiment, we verify the performance of CoLoRa in a real deployed LoRa network consisting of 20 LoRa end nodes in the campus. Fig. 8 shows the deployment environment, which has several multi-story buildings, trees and hills. Each end node is equipped with a temperature sensor and a humidity sensor, and the sensed data is transmitted to the base station periodically in a regular interval (duty cycle of 0.1 in our experiment). The transmitted data is encoded with a spreading factor of 12 and a coding rate of 4/8 and the length of the packet is no more than 20 Byte. On the MAC layer, we adopt LoRaWAN MAC based on pure ALOHA for all the end nodes.

Fig. 13 shows the performance of three different LoRa receivers under a low duty-cycled network of size less than 20. Considering the LoRaWAN receiver without any collision resolution scheme, the PLR of LoRaWAN increase rapidly when 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 channel utilization. However, when the network scales, collisions happen frequently, which significantly degrades the performance of the LoRaWAN receiver. Overall, we consider the case for 10 nodes with 10% duty cycle ratio. In such a case, there are enough data in the channel for transmission and LoRaWAN+Oracle should achieve its highest throughput, i.e., 100%. We can see that LoRaWAN only achieves 7.2% of the Oracle. The throughput of CoLoRa is about 14× that of LoRaWAN. Choir can separate overlapped packets from a collision based on the hardware offset. For the low duty cycle network with a small number of nodes, Choir also performs well as the number of concurrent transmissions is expected to be low (e.g., ≤ 4). This coincides the results in Fig. 10(a) where CoLoRa and Choir have similar performance for low concurrency. The performance of Choir also degrades when the size of network increases.

As shown in Fig. 13(a), PLR of Choir is around 30% when there are 20 concurrent nodes in the network. CoLoRa outperforms the other two approaches when the network scales and it keeps a relative low PLR. The resulted network throughput grows nearly linearly as the size of network increases, indicating CoLoRa can decompose most of the collided packets.

VII. Related Work

a) ***Collision resolution in Wireless:*** Extensive works focus on collision resolution and parallel decoding in various wireless systems [15]–[18] (e.g., Wi-Fi, RFIDs and cellular networks). Some advocate Multiple-Input Multiple-Output (MIMO) to exploit spatial diversity across multiple paths [19], [20]. MIMO based approaches, which significantly improves the throughput, cannot be used in LoRa with a single antenna. Successive interference cancellation (SIC) based approaches resolve collisions by iteratively canceling interference from collided signals [21], [22]. These schemes work only when the colliding senders transmit under strict power control, and thus they are usually used in cellular networks. ZigZag [7] combats inter-packet collisions in 802.11. It utilizes different collision-free parts of different collisions to iteratively decode the overlapped packets. In this way, ZigZag decodes an m-packet collision based on m repeated collisions. mZig [8] decompose m concurrent ZigBee packets from one collision directly. It starts with a collision-free chunk and then iteratively reconstructs and extracts each decoded symbol.

*b) **Concurrent transmissions in LoRa:*** Recently, NetScatter [9] proposes a multi-packet reception strategy which enables hundreds of concurrent transmissions in LoRa backscatter systems [23]. The key innovation of NetScatter is a distributed coding mechanism where each node is assigned a shifted chirp symbol and uses on-off keying to modulate data. NetScatter requires that all the transmitters are synchronized, and hence it cannot work for unsynchronized transmission in existing LoRa communications. DeepSense [24] enables random access and coexistence for different LoRa configurations by exploring machine learning algorithms on-board. It identifies the presence of LoRa collisions using the neural networks. However, in the emergence of collisions with the same LoRa configuration, DeepSense cannot recover any of the collided data bits. Choir [10] proposes a collision resolution method for LoRa. It depends on the fact that the hardware imperfection of a low-cost LoRa end node causes a frequency offset of the corresponding generated chirp signal. Choir utilizes this frequency offset to decompose collided packets of different end nodes. However, accurately extracting the tiny frequency offset is very difficult especially for low SNR LoRa signals. Meanwhile, the frequency offsets of low-cost end nodes are changeable over time, which impacts its performance in practice. mLoRa [25] applies SIC to LoRa collisions. It starts with a collision-free chunk and then iteratively reconstructs and extracts each decoded chirp symbol. FTrack [26] decodes multiple LoRa packets from a collision by calculating the instantaneous frequency continuity. Both mLoRa and FTrack have fundamental limitations in decoding low SNR LoRa signals, as they focus on the time domain signal analysis and do not consider the modulation features of LoRa.

VIII. Conclusion

We present CoLoRa, a multi-packet reception protocol in LoRa to address the practical collision problem of LPWAN deployments. CoLoRa utilizes, perhaps counter-intuitively, packet time offset to decompose multiple packets from a single collision. We propose several novel techniques to address practical challenges in CoLoRa design. We translate time offset, which is difficult to measure for symbols in collisions, to robust frequency features, i.e., peak ratios, for low SNR LoRa signal. We design a method to extract accurate peak ratios by canceling inter-packet interference. Finally, we address the frequency shift incurred by CFO and time offset to decode LoRa packets. CoLoRa is completely implemented in software at the base station, without requiring any modifications to end nodes. The evaluation results show that CoLoRa improves the network throughput by **3.4**× compared with Choir and **14**× compared with LoRaWAN. We believe CoLoRa can be easily applied to today's LoRa networks with a very small overhead.

Acknowledgements

This work is in part supported by National Key R&D Program of China 2018YFB1004800 , National Natural Science Fund for Excellent Young Scholars (No. 61722210), National Natural Science Foundation of China (No. 61932013, 61532012).

References

[1] Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. Low power wide area networks: An overview. *IEEE Communications Surveys & Tutorials*, 2016.

[2] Jothi Prasanna Shanmuga Sundaram, Wan Du, and Zhiwei Zhao. A survey on lora networking: Research problems, current solutions and open issues. *IEEE Communications Surveys & Tutorials*, 2019.

[3] Silvia Demetri, Marco Zúñiga, Gian Pietro Picco, Fernando Kuipers, Lorenzo Bruzzone, and Thomas Telkamp. Automated estimation of link quality for lora: A remote sensing approach. In *Proceedings of ACM/IEEE IPSN*, 2019.

[4] Weifeng Gao, Wan Du, Zhiwei Zhao, Geyong Min, and Mukesh Singhal. Towards energy-fairness in lora networks. *Proceedings of IEEE ICDCS*, 2019.

[5] Akshay Gadre, Revathy Narayanan, Anh Luong, Anthony Rowe, Bob Iannucci, and Swarun Kumar. Frequency configuration for low-power wide-area networks in a heartbeat. In *Proceedings of USENIX NSDI*, 2020.

[6] Ferran Adelantado, Xavier Vilajosana, Pere Tuset, Borja Martínez, Joan Melià-Seguí, and Thomas Watteyne. Understanding the limits of lorawan. *IEEE Communications Magazine*, 2016.

[7] Shyamnath Gollakota and Dina Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. In *Proceedings of ACM SIGCOMM*, 2008.

[8] Linghe Kong and Xue Liu. mzig: Enabling multi-packet reception in zigbee linghe. In *Proceedings of ACM MobiCom*, 2015.

[9] Mehrdad Hessar, Ali Najafi, and Shyamnath Gollakota. Netscatter: Enabling large-scale backscatter networks. In *Proceedings of USENIX NSDI*, 2019.

[10] Rashad Eletreby, Diana Zhang, Swarun Kumar, and Osman Yağan. Empowering low-power wide area networks in urban settings. In *Proceedings of ACM SIGCOMM*, 2017.

[11] Reinoud Sleeman and Torild van Eck. Robust automatic p-phase picking: an on-line implementation in the analysis of broadband seismogram recordings. In *Physics of the earth and planetary interiors*, 1999.

[12] Chaojie Gu, Rui Tan, and Jun Huang. Secure data timestamping in synchronization-free lorawan. In *ArXiv*, 2019.

[13] G. FSF. Gnu radio - gnu fsf project.

[14] N Sornin, M Luis, T Eirich, T Kramp, and O Hersent. Lorawan specification. *LoRa alliance*, 2015.

[15] Omid Salehi-Abari, Deepak Vasisht, Dina Katabi, and Anantha Chandrakasan. Caraoke: An e-toll transponder network for smart cities. In *Proceedings of ACM SIGCOMM*, 2015.

[16] Jiajue Ou, Mo Li, and Yuanqing Zheng. Come and be served: Parallel decoding for cots rfid tags. In *Proceedings of ACM MobiCom*, 2015.

[17] Yuanqing Zheng and Mo Li. Read bulk data from computational rfids. *Proceedings of IEEE INFOCOM*, 2014.

[18] Lei Yang, Jinsong Han, Yong Qi, Cheng Wang, and Yunhao Liu. Revisiting tag collision problem in rfid systems. In *Proceedings of IEEE ICPP*, 2010.

[19] Swarun Kumar, Diego Cifuentes, Shyamnath Gollakota, and Dina Katabi. Bringing cross-layer mimo to today's wireless lans. In *Proceedings of ACM SIGCOMM*, 2013.

[20] Yaxiong Xie, Yanbo Zhang, Jansen Christian Liando, and Mo Li. Swan: Stitched wi-fi antennas. In *Proceedings of ACM MobiCom*, 2018.

[21] Daniel Halperin, Thomas Anderson, and David Wetherall. Taking the sting out of carrier sense: interference cancellation for wireless lans. In *Proceedings of ACM MobiCom*, 2008.

[22] Shyamnath Gollakota, Samuel David Perli, and Dina Katabi. Interference alignment and cancellation. In *Proceedings of the ACM SIGCOMM*, 2009.

[23] Vamsi Talla, Mehrdad Hessar, Bryce Kellogg, Ali Najafi, Joshua R Smith, and Shyamnath Gollakota. Lora backscatter: Enabling the vision of ubiquitous connectivity. In *Proceedings of ACM Ubicomp*, 2017.

[24] Justin Chan, Anran Wang, Arvind Krishnamurthy, and Shyamnath Gollakota. Deepsense: Enabling carrier sense in low-power wide area networks using deep learning. *ArXiv*, 2019.

[25] Xiong Wang, Linghe Kong, Liang He, and Guihai Chen. mlora: A multi-packet reception protocol for lora communications. In *Proceedings of IEEE ICNP*, 2019.

[26] Xia Xianjin, Zheng Yuanqing, and Gu Tao. Ftrack: Parallel decoding for lora transmissions. In *Proceedings of ACM SenSys*, 2019.