Hitting the Sweet Spot: An SF-any Coding Paradigm for Empowering City-Wide LoRa Communications

Weiwei Chen*
Shanghai University
Shanghai, China
Southeast University
Nanjing, China
chen.ava.0012@gmail.com

Jiefeng Zhang*
Southeast University
Nanjing, China
jiefeng_zhang@seu.edu.cn

Xianjin Xia
The Hong Kong Polytechnic University
Hong Kong, China
xianjin.xia@polyu.edu.hk

Shuai Wang†
Southeast University
Nanjing, China
shuaiwang_iot@seu.edu.cn

Shuai Wang
Southeast University
Nanjing, China
shuaiwang@seu.edu.cn

Tian He
Southeast University
Nanjing, China
dr.tianhe2@gmail.com

ABSTRACT

LoRa technology has garnered significant attention for its exceptional performance in city-wide applications. LoRa encodes data across multiple samples to enable long-range communication, with the level of redundancy controlled by the Spreading Factor (SF). However, practical limitations restrict how high the SF can be set. To overcome communication challenges at the highest allowable SF settings, we introduce SF-any, a software-based coding paradigm that extends an SFk packet to a quasi-SF$(k + m)$ packet. SF-any encodes a quasi-SF$(k + m)$ symbol with 2^m SFk symbols. Hardware imperfections introduce time-varying frequency drifts and phase offsets, resulting in frequency leakage during quasi-SF$(k+m)$ packet decoding. To mitigate this, we strategically insert pilots into the packet for imperfection estimation and compensation. Additionally, to maintain and exploit the coding structure in LoRa PHY, we employ a grouped repetition code at the transmitter and a joint demodulation and decoding scheme at the receiver. Comprehensive evaluations demonstrate that SF-any's performance seamlessly scales with increasing SF, achieving up to a 14dB improvement over SF12 packets (the highest SF in LoRa PHY), and up to a 12dB improvement compared with state-of-the-art approaches.

KEYWORDS

LoRa, Reliability, Low SNR, Coherent Combining

1 INTRODUCTION

As a prominent Low Power Wide Area Networks (LPWANs) technology, LoRa [6] excels in long-range, low-power communication, enabling the connectivity of vast IoT deployments. Consequently, LoRa is widely used in various scenarios, especially in city-wide applications, such as Agricultural Monitoring [11], Wildlife Tracking [19], Marine and Oceanography [20], which require a communication range of up to several tens of kilometers.

LoRa achieves long-range communication by encoding data across a sequence of samples. The degree of redundancy and robustness in a LoRa link is contingent on the number of samples, which is determined by a parameter known as the Spreading Factor (SF). For example, an SFk symbol encompasses 2^k identical data copies, resulting in a processing gain of 2^k. By fine-tuning SF and the transmission rate, the LoRa Physical Layer (LoRa PHY) can adapt to varying link conditions. As will be explained in Sec. 2, simply increasing the value of SF will not improve, but instead compromises the overall processing gain. As a result, the maximum SF supported by LoRa PHY is presently 12 [21].

The constrained SF setting struggles to support communication over extremely weak links, resulting from not only long distances, but also common obstacles such as walls, vehicles, and pedestrians. This is especially critical for low-battery devices using the Class-A MAC protocol, which do not retransmit any uplink message until acknowledged by the gateway within a specified timeframe [1], leading to extremely long delays and unwarranted battery consumption. A fundamental question arises: how to enhance the signal quality for the extremely weak LoRa links?

Revisiting how LoRa's SF (Spread Factor) works, which groups together repeated physical samples (chips), inspires us to improve signal quality via incorporating more chips into a symbol to overcome noise. Based on this, we present SF-any, a lightweight software-based coding paradigm, compatible with COTS devices to support extremely low SNR links. More specifically, the coding paradigm modulates a quasi-SF$(k + m)$ symbol with 2^m SFk symbols. Here, m can be any integer. The coding paradigm breaks the SF limitations, adaptable to more severe channel states. Moreover, SF-any preserves the coding structure in LoRa PHY, paves the way to facilitate joint demodulation and decoding, thus fully leveraging the coding redundancy provided by LoRa PHY.

The primary challenges in implementing SF-any for COTS LoRa nodes are the following. First, to maximize the processing gain, the energy of a quasi-SF$(k+m)$ symbol should be efficiently aggregated. Unfortunately, as shown in [29], the hardware imperfections introduce phase and frequency offsets to the received chips. Even worse, such imperfections are time-varying and hard to estimate given extremely low SNRs, significantly hindering energy combining. To this end, we insert specially designed **PILOT** symbols in the packets. Joint processing of the preambles and PILOTS enables an accurate estimation and compensation for hardware imperfections.

*Both authors contributed equally to this paper.
†Shuai Wang is the corresponding author.

Second, to fully exploit the coding redundancy of LoRa PHY, the coding structure of the SFk packet is supposed to be preserved. Therefore, we devise a grouped repetition code to encode a quasi-SF$(k+m)$ symbol. In detail, refer to Fig. 6, instead of consecutively repeating a symbol, we group four payload symbols together, akin to the Hamming code operation in LoRa PHY, and repeat them 2^m times. Based on the code structure, we further introduce a joint demodulation and decoding scheme to recover symbols group by group, and restore corrupted symbols in each group.

SF-any enables versatile transmission rates. For instance, an SF13 packet can be emulated as an SF11 packet by repeating its symbol groups four times (quasi-SF$(11+2)$) or as an SF12 packet by repeating its symbol groups twice (quasi-SF$(12+1)$). Notably, this is the first work to support extremely high SF for COTS devices. Furthermore, SF-any operates independently and can be integrated with other methods that enhance LoRa communications in challenging channel conditions [3, 4, 7, 15–18, 27–29] for further improvement. In summary, the key contributions are as follows.

- We propose SF-any, a software-based coding paradigm that is not just compatible with but inspires LoRa PHY's capability for receiving extremely low SNR packets without any hardware modification. SF-any is fine-tuned to adapt to various channel conditions, seamlessly scaling performance with higher SF. Moreover, SF-any can be incorporated into orthogonal methods to achieve better performance.
- We develop a platform to conveniently implement SF-any on COTS LoRa systems. In particular, we leverage reverse engineering to perform pilot insertion and grouped repetition in LoRa PHY by manipulating the payload. The pilots help estimate and compensate for various signal offsets, thus ensuring coherent combining of the received signals. At the receiver side, the platform jointly exploits the energy distribution of the received signals and the preserved Hamming code constraints to decode symbols and recover errors.
- We conduct extensive experiments on commodity LoRa nodes (i.e. Semtech SX1278) to evaluate SF-any, and compare it with existing studies. The results indicate that SF-any achieves up to a 14dB SNR gain compared with existing works.

The rest of the paper is organized as follows. Sec. 2 provides the background and motivations. Sec. 3 presents the SF-any system architecture. Encoder and decoder designs are elaborated in Sec. 4 and Sec. 5, respectively. Extensive evaluations are reported in Sec. 6. Related works are reviewed in Sec. 7. Discussion and future work is presented in Sec. 8, and the paper is concluded in Sec. 9.

2 BACKGROUND AND MOTIVATIONS

This section presents an overview of common challenges in LoRa communications and the motivations behind this work. For the sake of clarity and to maintain consistency, this paper uses the terms "LoRa symbol" and "LoRa chirp" interchangeably.

2.1 A Primer on LoRa

LoRa employs Chirp Spread Spectrum (CSS) for modulation, expanding each symbol into multiple chips, the number of which is determined by the SF. All chips within a chirp traverse the entire bandwidth, and the initial frequency of a chirp (or the frequency of the first chip within a chirp) modulates the transmitting symbol. In a LoRa packet, the sth chirp is denoted by z_s, and the ith chip of the sth chirp (represented as $x_{s,i}$) is as follows:

$$x_{s,i} = \begin{cases} e^{j2\pi(\frac{i}{2}+z_s-\frac{N}{2})\frac{i}{N}} & 0 \le i < N - z_s \\ e^{j2\pi(\frac{i}{2}+z_s-\frac{3N}{2})\frac{i}{N}} & N - z_s \le i < N. \end{cases} \tag{1}$$

As the frequency of a chirp reaches its maximum value, it switches to the lowest value, resulting in a frequency discontinuity at the $(N-z_s)$th chip. We refer to chips 0 through $N-z_s-1$ as the "**first chip set**" of a symbol, and chips $N-z_s$ through $N-1$ as the "**second chip set**". By de-chirping the signal using a standard down-chirp sequence, the chips within a chirp are transformed into a single-frequency tone. This frequency tone is subsequently extracted using an FFT operation.

2.2 The Impact of Frequency Leakage

In practice, there exist two types of frequency leakage that can drastically affect decoding efficiency.

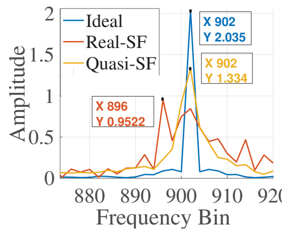
2.2.1 Frequency leakage brought by channel variations. In essence, SF (k) directly influences the number of chips in a symbol, resulting in increased redundancy and higher processing gain. LoRa employs different SF values to accommodate varying channel conditions. In particular, in LoRa PHY, the maximum SF value is 12. Consequently, links that cannot be sustained using SF12 packets may encounter communication restrictions.

A straightforward approach to improve link robustness is to increase the SF arbitrarily. However, raising the SF without proper consideration can introduce practical issues, potentially reducing the expected processing gain associated with the extended SF.

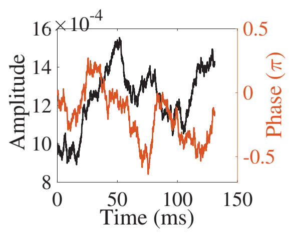
Here is one contributing factor. LoRa is characterized by its low bandwidth, supporting very low transmission rates for the chips. For example, with a 125kHz transmission bandwidth, the chip duration is 8μs. The symbol duration for an SF12 symbol extends beyond 32ms, and if an SF16 packet is used, the symbol duration exceeds 524ms. Furthermore, the typical coherence time of a wireless channel is several tens of milliseconds [24]. Beyond this duration, the complex real-world environment along the lengthy transmission path, spanning kilometers, is deemed to undergo significant variations. It implies that an SF16 symbol can undergo rapid channel variations, which prevents chips within a symbol from being coherently combined, ultimately resulting in a significant reduction in the processing gain associated with a higher SF.

Fig. 1(a) presents the spectrum distributions of ideally combined SF14 symbol with or without ideal combination, and that of a quasi-SF$(12+2)$ symbol. The ideally combined SF14 symbol results in a highest peak. The quasi-SF$(12+2)$ symbol generates a frequency peak located in the same frequency bin as the one generated from the ideal SF14 symbol with slightly lower amplitude. However, the SF14 symbol without compensation, suffering from frequency leakage, hardly concentrates the energy into any distinct peaks. Ideally, phase misalignment between chips due to channel variations can be effectively compensated for, leaving a distinguishable peak in frequency domain. However, it is impractical, since predicting the channel states for an extremely low SNR symbol is not feasible.

The channel state of the SF14 symbol is depicted in Fig. 1(b). Here, the bandwidth is set to 125kHz, and the SF14 symbol lasts 131 ms. In

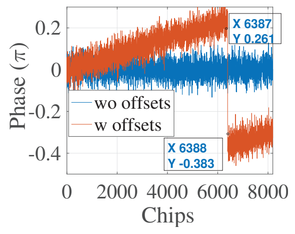


(a) Spectrum distribution

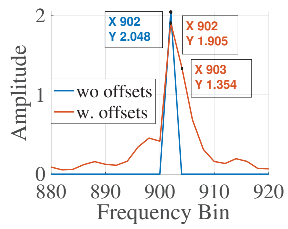


(b) Channel state

Figure 1: (a). The spectrum distribution of a real SF14 symbol, a Quasi-SF$(12+2)$ symbol, and the ideal SF14 symbol with ideal compensation for channel state and hardware imperfections. (b). The channel state of an SF14 symbol.



(a) Phases of each chip



(b) Spectrum distribution

Figure 2: (a) The phases of each chip in a symbol after removing the starting frequency $z_s = 902$. (b) The spectrum of an SF12 symbol with and without intra-chirp offsets.

the non-ideal case, the chips in an SF14 symbol are combined using the standard approach implemented in LoRa PHY. Since channel states vary within a symbol, the chips are not coherently aligned, resulting in substantial frequency leakage. Therefore, rather than improving the SNR, elevating the SF leads to unexpected demodulation errors. The frequency leakage becomes more pronounced when the length or SF of a symbol further increases.

Fortunately, we have observed that replacing an extremely long SF symbol with a set of shorter SF symbols can effectively mitigate the frequency leakage caused by channel variations. Take SF14 as an example, a quasi-SF$(12+2)$ symbol simulating an SF14 symbol is composed of 2^2 SF12 symbols. To demodulate such a symbol, we first leverage the standard LoRa PHY approach to combine chips within each SF12 symbol. We aggregate the absolute values of the four symbols to obtain the final demodulation. Since all chips in the SF12 symbols are well-aligned (given the relatively stable channel state within an SF12 symbol), the frequency leakage within a symbol is greatly reduced, facilitating accurate decoding.

2.2.2 Frequency leakage brought by hardware imperfections. As demonstrated in [29], hardware imperfections significantly distort the received signal. Typically, these imperfections fall into three categories: Carrier Frequency Offset (CFO), Sampling Time Offset (STO), and Sampling Frequency Offset (SFO). Alongside the effects of channel variations, hardware imperfections inevitably introduce frequency leakages when receiving high SF symbols.

Phase offsets within a symbol: Fig. 2(a) displays the phases of all the chips within a symbol received by both an ideal receiver and an actual receiver. In this scenario, an SF12 symbol is transmitted

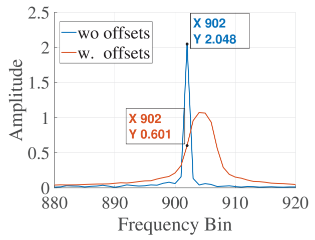


Figure 3: An illustration of how frequency drifts degrade the signal combining efficiency.

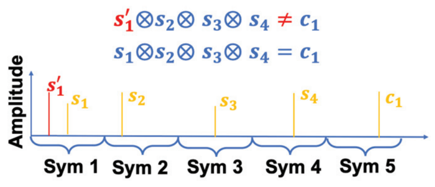
with a bandwidth of 125kHz, and the sampling rate is twice that of the bandwidth. The transmitted symbol is $z_s = 902$. Notably, a phase shift occurs between the first chip set (chip 0 to 6387) and the second chip set (the remaining chips). Fig. 2(b) highlights frequency leakage in FFT results due to hardware imperfections, revealing other peaks (e.g., 900, 901, 903, 904) with amplitudes close to the main peak at 902, indicating significant signal distortion. This leakage, especially in low SNR packets, can be exacerbated by environmental noise, potentially drowning out the intended signal.

Continuous frequency drift across symbols: As shown in Fig. 2(a), the slope of the red lines represents the frequency drift within a symbol, a joint consequence of both STO and CFO as discussed in [29]. Due to the SFO, the STO experiences changes over time, resulting in a time-varying frequency drift for each symbol. Although frequency drift is negligible within one symbol, it accumulates over the course of receiving a complete packet. According to the datasheet [21], the allowable frequency drift for long SF packets (in which LDRO mode is recommended) is up to $\frac{16}{3}$ chips.

Employing repetition codes further magnifies this challenge. The higher the repetition rate, the longer the packet length, and the more pronounced the frequency drifts. Therefore, repeating symbols in a packet, without compensating for offsets as is done in Ostinato [26], limits the highest possible repetition rate due to ever-growing frequency drifts. Once the frequency drifts exceed one frequency bin, the subsequent symbols cannot be demodulated correctly, leading to constant bit errors regardless of SNR levels.

When combining SFk symbols to emulate a quasi-SF$(k+m)$ symbol, it signifies that the frequency peaks of each SFk symbol will not align, but will drift over time. Consequently, instead of consolidating the energy of the frequency peaks, it spreads the energy, introducing substantial frequency leakage. Fig. 3 depicts the impact of frequency drifts when combining 32 symbols. In this case, the ever-growing frequency drift not only shifts target peak (bin 902), but causes severe frequency leakage which spreads the peak energy into a wide range of frequency bins (bin 900 to 910).

Therefore, to achieve the processing gain when emulating SF-any symbols, hardware imperfections need to be meticulously compensated for. However, accurately estimating frequency and phase offsets becomes challenging given extremely low SNR levels. Traditional methods, such as those used in P^{2}LoRa [12] and SCLoRa [10], relying on up-chirps and down-chirps in the preamble, struggle due to the severe distortion of frequency information caused by strong noise. Additionally, as the length of the packet increases, the frequency drift and STO resulting from SFO tend to change over time, making it difficult to accurately estimate based solely on the



$s_1' \otimes s_2 \otimes s_3 \otimes s_4 \neq c_1$
$s_1 \otimes s_2 \otimes s_3 \otimes s_4 = c_1$
Amplitude
s_1'
s_1
s_2
s_3
s_4
c_1
Sym 1
Sym 2
Sym 3
Sym 4
Sym 5

Figure 4: An illustration of joint demodulation and decoding.

preamble information. In contrast, SF-any resolves this issue via pilot insertion to track these offsets throughout the packet.

2.3 Converting Hamming Code in LoRa PHY into a Robust Error Correction Code

In the conventional LoRa PHY approach, the system traditionally selects the highest frequency peak in the FFT result as the demodulated symbol. Subsequently, the demodulated symbols undergo verification in the Hamming decoding module. However, independent demodulation and decoding processes underutilize the potential of the energy distribution of received IQ signals. In theory, with four correctly demodulated symbols, it is possible to recover the remaining symbols, subject to known coding constraints. The question then arises: How do we determine the corrected symbols? Fortunately, this decision can be informed by considering both the energy distribution of the IQ signals and the coding constraints.

In Fig. 4, instead of straightforwardly demodulating the first symbol as s_1', a more informed approach is adopted. We determine that s_1' is an erroneously demodulated symbol due to two key reasons: i) Its amplitude, although slightly higher than that of s_1, remains within a similar scale; and ii) Demodulating the first symbol as s_1' would cause the group of symbols to fail the parity check. Based on these insights, the first symbol is determined to be s_1. This observation motivates the exploration of coding redundancy in Hamming code to devise a resilient error correction decoding scheme.

2.4 The Need of SF-any

The state-of-the-art design Ostinato [26] decodes low-SNR LoRa packets via a symbol-by-symbol repetition coding, without exploiting LoRa's built-in Hamming coding mechanism. This is because the Hamming code structure, especially the mandatory parity symbols, violates the symbol repetition principle of Ostinato. Moreover, the lack of compensation for the signal offsets further limits the performance of Ostinato. In contrast, SF-any i) proposes a grouped repetition approach, which complies with the built-in Hamming coding mechanism, thereby achieving additional performance gain compared to Ostinato; ii) allows a flexible repetition rate, as opposed to Ostinato, which only supports $\leq$ 4 repetitions; and iii) carefully investigates the hardware imperfections to accomplish the signal offset compensation, providing further performance gain. In addition, besides the repetition code and Hamming code, SF-any can be combined with other orthogonal approaches, such as coding schemes (e.g., the Reed-Solomon code [28]) and multiple gateway approaches (e.g., CONST [29]), to obtain additional gains.

3 SYSTEM OVERVIEW

As shown in Fig. 5, the architecture of SF-any primarily consists of the encoder at the transmitter and the decoder at the receiver.

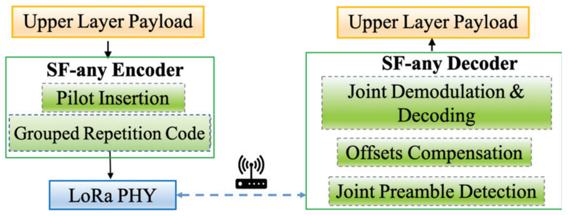


Figure 5: The overall design architecture of SF-any.

3.1 SF-any Encoder Design

The encoding module in the transmitter incorporates PILOTs into the packet and applies a grouped repetition method to encode the upper-layer payload. An SF-any header is introduced to streamline the transmission of SF-any packets. Notably, this encoding module operates by manipulating payload bits before transmitting them via LoRa PHY, eliminating the need for any hardware modifications to implement SF-any in a LoRa transmitter. The SF-any encoder is typically utilized in battery-constrained remote LoRa nodes, as gateway nodes, being power-sufficient, can utilize higher power levels to reach remote nodes.

3.2 SF-any Decoder Design

The preamble detection sub-module serves to detect new packet arrivals and synchronize with them. Meanwhile, the hardware offset compensation module extracts hardware offsets from preambles and pilots, compensating for frequency drift and intra-symbol phase drift caused by CFO and STO. This module also combines the repeated 2^m SFk symbols into quasi-SF$(k+m)$ symbols. Lastly, the error correction-based decoding module performs joint demodulation and decoding of each group of quasi-SF$(k+m)$ symbols. It checks parity constraints, identifies error symbols within the group, and corrects the wrongly decoded symbols based on their spectrum energy distributions.

4 THE DETAILED DESIGN OF THE ENCODER

This section describes the SF-any packet format and how to encode the upper layer payload into an SF-any packet.

4.1 Grouped Repetition Code

In LoRa PHY, four symbols form a group. The group of symbols will be encoded by Hamming code to produce a larger group. For example, with CR4/5 or CR4/7 encoding, four symbols are encoded into a group of five or seven symbols, respectively, with the first four symbols being the payload and the rest being parity symbols.

To preserve the Hamming coding structure while enhancing symbol reception energy, SF-any employs a grouped repetition code, bundling four symbols together and repeating them in groups. This approach also leverages time diversity by distributing chips comprising an SF-$(k+m)$ symbol across different SFk symbols, which are likely to experience varying channel conditions, leading to increased channel hardening with higher repetition rates, ensuring stable performance for different symbols in a packet.

For example, when modulating quasi-SF$(k+m)$ symbols, a symbol group is repeated 2^m times, forming a supergroup. E.g., as

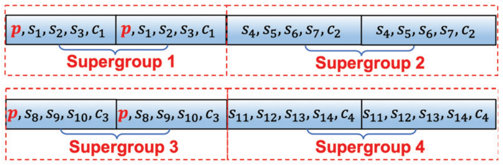

p, s_1, s_2, s_3, c_1
p, s_1, s_2, s_3, c_1
s_4, s_5, s_6, s_7, c_2
s_4, s_5, s_6, s_7, c_2
Supergroup 1
Supergroup 2
p, s_8, s_9, s_{10}, c_3
p, s_8, s_9, s_{10}, c_3
$s_{11}, s_{12}, s_{13}, s_{14}, c_4$
$s_{11}, s_{12}, s_{13}, s_{14}, c_4$
Supergroup 3
Supergroup 4

Figure 6: An illustration of how to form supergroups.

depicted in Fig. 6, a packet encoded with CR4/5 and $m = 1$ would comprise two groups of symbols forming a supergroup.

4.2 Pilot Insertion

SF-any incorporates pilot symbols to enhance the tracking of hardware imperfections. These pilot symbols can take any value, but in this paper, we use $z_s = 2^{k-1}$ as the pilot value. As shown in the following section, setting the pilot to $z_s = 2^{k-1}$ helps to achieve a balanced distribution of the number of chips in the first chip set (chip 0 to chip $N - z_s - 1$) and the second chip set (chip $N - z_s$ to chip $N - 1$) for the STO estimation.

The next question is how to insert pilots into the packet. Generally, pilots can be inserted into any supergroup. However, once a pilot is inserted into a supergroup, it repeats 2^m times to maintain the Hamming coding structure. For example, in Fig. 6, a quasi-SF($k + 1$) symbol is emulated with SFk symbols, payload symbols along with pilot symbol within the supergroup repeat twice.

Since the preambles effectively estimate the initial states of CFO and STO, the pilots primarily extract SFO information. To achieve a balanced performance and complexity tradeoff, we set 8 pilots each six supergroups (sacrificing one sixth data rate) for double repetitions, ensuring that the data rate for SF-any remains equivalent to that used for Low Data Rate Optimization (LDRO) mode in Ostinato with the same repeating rate. The LDRO mode ignores the last two bits of each symbol (i.e., one sixth of the data rate in SF12) for symbol protection against frequency leakage. We insert 16 pilots for quadruple repetitions and 32 pilots for higher repetitions.

4.3 Packet Format for SF-any Implementation

SF-any is implemented on Commercial Off-The-Shelf (COTS) LoRa transmitters, and it adheres to the packet format specifications of LoRa PHY. Fig. 7 illustrates the packet format for an SF-any packet. Specifically, the number of preambles in a packet can be customized to an arbitrary value. When implementing a quasi-SF($k+m$) packet with SFk symbols, the number of preambles is set to $\max\{6, 2^m\}$ SFk symbols to ensure accurate preamble detection.

Following the Start of Frame Delimiter (SFD), SF-any constructs a Header to embed essential information for packet decoding. The SF-any Header consists of two symbols: one pilot symbol for synchronization and another symbol indicating the parameters used in an SF-any packet. In detail, for a quasi-SF($k + m$) packet, the first three bits encode the value of m, while the next five bits represent the number of supergroups (denoted by s_g). Denote k bits of SF-any header by $\{b_0, b_1, ..., b_{k-1}\}$, where m and s_g are calculated as $m = \sum_{i=0}^{2} b_i 2^i$, $s_g = \sum_{i=3}^{7} b_i 2^{i-3}$, and $b_i = 0$ for any $7 < i < k$.

In total, there are s_g supergroups, each repeated by 2^m groups of symbols. It is important to note that coding rate information is embedded in the grouping pattern (the number of symbols in a supergroup), and it can be decoded once the pilots are detected.

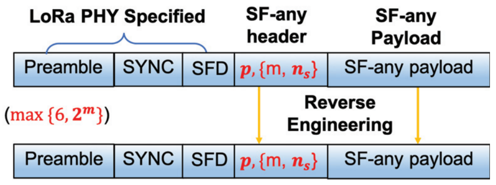


Figure 7: The structure of an SF-any packet.

Upon receiving the upper layer payload, SF-any computes the number of supergroups s_g, assembles the SF-any header, and allocates pilots to their respective supergroups. SF-any subsequently employs a reverse engineering technique [14, 26] to generate the final payload, which is then transmitted through LoRa PHY.

5 THE DETAILED DESIGN OF THE DECODER

This section describes how to detect and decode an SF-any packet.

5.1 Preamble Detection

The preamble detection mechanism identifies the arrival of a packet with sliding windows. Since the number of preambles in a quasi-SF($k+m$) packet is set to $\max\{6, 2^m\}$, the sliding window is initially configured with a size of 2^5 symbols. This window continuously scans the received signal, with a step size equal to one fourth the length of an SFk symbol. In this process, all the chips within each symbol are correlated with downchirps and subjected to a standard 2^k-point FFT. The absolute values of the spectrum for all symbols within the window are then summed. As the receiver constantly monitors environmental noise, it can detect the arrival of a new packet when the cumulative SNR of a frequency peak surpasses a predefined margin. The choice of this SNR margin is extended in the evaluations.

To determine the initial preamble position, the sliding window is left-shifted until the average cumulative SNR of the preambles within the current window decreases. From this starting position, the window either extends to the right or reduces in size from the right, following the same criteria.

5.2 Packet Synchronization

While the sliding window aims to capture all preambles, there may be instances where some preambles are missing due to extremely low SNR. In such cases, the SF-any header plays a crucial role in accurately synchronizing to the payload of a packet.

The initial step involves locating the positions of the pilot symbols, which is determined by coding rates. For example, when using CR4/5 or CR4/8 encoding, the pilot symbol repeats every five or eight symbols, respectively. Therefore, various pilot intervals are tested to identify the one that maximizes the cumulative pilot energy. More specifically, let $F_{s,p}$ represent the pth frequency tone of the sth symbol, with p being the intended pilot symbol and s as the estimated symbol position after SFD. In cases of extremely low SNR, s may deviate from its actual position by one or two symbols. The search is conducted to locate the pilot's position subject to:

$$\{c, s, \bar{m}\} = \arg\max_{c,s,\bar{m}} \frac{1}{2^{\bar{m}}} \sum_{k=0}^{2^{\bar{m}}-1} |F_{s+ck,p}|. \qquad (2)$$

Here c and s refer to the coding rate and the starting position of the SF-any header. E.g., when $c = 5$, the coding rate is CR4/5. $\bar{m}$ is the estimated quasi-spreading factor. We use the absolute values of the first and the second chip sets to measure the amplitude of the pilot as it is a known value.

Theoretically, once the positions of the pilot symbols are determined, it's possible to ascertain the quasi-spreading factor m. Furthermore, to improve the accuracy of the spreading factor estimation and the synchronization process, we proceed to decode the SF-any header, which explicitly encodes the value of m. Let $F_{s,h}$ represent the hth frequency tone of the sth symbol, where h encodes the SF-any header. We define B_h as $\{b_0, b_1, b_2, \cdots, b_{k-1}\}$, where $h = \sum_{i=0}^{k-1} b_i 2^i$. For each h, designate m_h as $\sum_{i=0}^{2} b_i 2^i$, and H_m as $\{h | m_h = m, \text{and } b_i = 0 \text{ for all } 7 < i \leq k-1\}$. Here, m_h represents the quasi-SF parameter m, and H_m is the set of frequency bins that could potentially encode the SF-any header with quasi-SF parameter m. The target quasi-SF parameter m^* and the corresponding SF-any header h^* is determined by:

$$\{m^*, h^*\} = \arg\max_{h \in H_m} \frac{1}{2^m} \sum_{k=0}^{2^m - 1} |F_{s+ck,h}|. \tag{3}$$

Upon determining the optimal m^* and h^*, the number of super-groups is also obtained. It is worth noting that, as the last part of the header is set to 0 (four bits for SF12 packet), these bits are resilient to spectrum leakages caused by hardware imperfections.

5.3 Signal Combining in SF-any

This section details how hardware imperfections affect received signals and introduces the corresponding combining scheme.

5.3.1 The impact of hardware imperfections on the received symbols. As discussed in [29], hardware imperfections can introduce significant distortions in the received signal. Of particular concern are three key offsets: CFO (Carrier Frequency Offset), STO (Sampling Time Offset), and SFO (Sampling Frequency Offset). CFO and SFO are denoted by δ and η, respectively. While δ and η may vary gradually across different transmissions, they are relatively stable within a single packet, assuming the oscillators in both the transmitter and receiver function normally. Additionally, let τ_s represents the STO of the sth symbol. Due to SFO, τ_s can vary over time. Designate Φ_s as the phase offset resulting from hardware imperfections.

Since an SFk symbol duration is shorter than the coherent time of the channel, we use a fixed channel state h_s to approximate the channel state of each chip in a chirp. It is worth noting that in cases where the channel exhibits extremely rapid variations, it is possible to reduce the value of k and the symbol duration to mitigate the extent of channel variations within a symbol. Furthermore, given that LoRa nodes are typically deployed in relatively stable locations, the frequency drift resulting from channel fading is considerably lower than that caused by hardware imperfections.

Given the above analysis, the received and de-chirped signals are characterized as follows, as outlined in [29].

$$y_{s,i} = \begin{cases} |h_s| e^{j2\pi\Phi_s} e^{j2\pi(z_s + f_s)\frac{i}{N}}, & 0 \leq i < N - z_s, \\ |h_s| e^{j2\pi\hat{\Phi}_s} e^{j2\pi(z_s + f_s)\frac{i}{N}}, & N - z_s \leq i < N. \end{cases} \tag{4}$$

Here, $N = 2^k$ is the number of chips in an SFk symbol, $|h_s|$ and $\angle h_s$ represent the amplitude and the angle of h_s. With this, we have

$$\Phi_s = \angle h_s + \Phi_{s,CFO} + \Phi_{s,STO}, \tag{5a}$$

$$\Phi_{s,CFO} = (s-1)\delta, \tag{5b}$$

$$\Phi_{s,STO} = -\frac{z_s \tau_s}{N} + \frac{\tau_s}{2} + \frac{{\tau_s}^2}{2N}, \tag{5c}$$

$$\hat{\Phi}_s = \Phi_s + \tau_s, \tag{5d}$$

$$f_s = \delta - \tau_s. \tag{5e}$$

According to (4), there are three types of drifts caused by hardware imperfections and channel state:

i) Inter-symbol phase offsets: the phase offset Φ_s is a joint effect of the channel state h_s, the CFO and the STO;

ii) Intra-symbol phase offsets: given the assumption that channel state remains stable in a symbol, the phase offsets between the first and the second chip set within a symbol is solely caused by STO (refer to (5d), $\tau_s = \hat{\Phi}_s - \Phi_s$); and

iii) Inter-symbol frequency drifts: while CFO causes a constant frequency drift (δ), STO causes a time-varying frequency drift ($-\tau_s$). The two factors make the frequency tones ($\delta - \tau_s$) drift further and further away from the actual transmitted frequency tone.

5.3.2 Hardware imperfection estimation for weak links. The decoder leverages both preambles and pilots for hardware imperfection estimation. Specifically, refer to (5d), the STO τ_s is calculated as:

$$\tau_s = \frac{1}{2\pi} \angle\left(\frac{g_{s,2}}{g_{s,1}}\right), \tag{6a}$$

$$g_{s,1} = \sum_{i=0}^{N-z_s-1} y_{s,i} e^{-j2\pi(z_s+\delta-\tau_s)\frac{i}{N}}, \tag{6b}$$

$$g_{s,2} = \sum_{i=N-z_s}^{N-1} y_{s,i} e^{-j2\pi(z_s+\delta-\tau_s)\frac{i}{N}}. \tag{6c}$$

When $z_s = \frac{N}{2} = 2^{k-1}$, the number of chips in $g_{s,1}$ and $g_{s,2}$ are balanced well. This explains why we choose $\frac{N}{2}$ as the pilot.

Consider the set of preambles and pilots denoted by S_p. For each preamble, following a similar approach as used in Magnifier [5], we shift the window to the middle of two preambles. This transformation effectively converts the preambles into pilot signals carried by the frequency tone $z_s = \frac{N}{2}$. To estimate the hardware imperfections, we employ the optimization approach outlined below.

$$\max_{\tau_0,\delta,\eta} \sum_{s \in S_p} |g_{s,1} + g_{s,2} e^{-j2\pi\tau_s}| \tag{7a}$$

$$\text{s.t.} \quad \begin{cases} \tau_s = \tau_0 + (s-1)\eta, \\ g_{s,1} = \sum_{i=0}^{N-z_s-1} y_{s,i} e^{-j2\pi(z_s+\delta-\tau_s)\frac{i}{N}}, \\ g_{s,2} = \sum_{i=N-z_s}^{N-1} y_{s,i} e^{-j2\pi(z_s+\delta-\tau_s)\frac{i}{N}}. \end{cases} \tag{7b}$$

According to [29], the STO of a normally functioned transmitter grows almost linearly with time. Therefore, the STO is approximated with a linear function of SFO (7b).

5.3.3 *Hardware imperfection compensation and symbol combining.*

In theory, the maximum processing gain is achieved when all chips within an SF-any symbol are coherently combined. However, practical challenges arise as the acquisition of channel state information $h_{s,i}$ is not achievable. Consequently, we do not apply compensation for the phase offsets between symbols. Nevertheless, refer to (5), the assumption that the channel state remains relatively stable within a symbol allows direct compensation for intra-symbol phase offsets.

Moreover, the frequency drift gradually increases over time due to SFO, leading to substantial frequency leakage if left unaddressed. Therefore, besides the intra-symbol phase offset, inter-symbol frequency drifts should also be compensated for as follows:

Signal combining within a chirp: As the channel state in a symbol is stable, chips within a symbol can be aligned to concentrate the energy of the target frequency tone while mitigating the frequency leakage arising from hardware imperfections.

Specifically, we will address the intra-symbol phase offsets caused by STO, and the frequency drift caused by both CFO and STO. Once the offset information has been estimated from (7), denote the estimated STO, CFO, and SFO by $\tau_s^*, \delta^*, \eta^*$, respectively, and the corresponding aggregation of first chip set and second chip set by $g_{s,1}^*$ and $g_{s,2}^*$ respectively. Let $y_{s_z,i} = y_{s,i}$, $Z_{s_z} = z$, and $\tau_{s_z}^* = \tau_s^*$ (refer to (6b) and (6c)), the signal of the zth frequency tone denoted by $Y_{s,z}$ can be compensated with:

$$Y_{s,z} = g_{s_z,1}^* + g_{s_z,2}^* e^{-j2\pi\tau_s^*}. \tag{8}$$

The compensation helps align chips in a symbol. After compensating for the intra-symbol phase offsets, and the inter-symbol frequency drift, frequency leakage can be significantly alleviated.

Signal combining between different chirps: As the channel state changes from symbol to symbol, and hard to estimate due to links' extremely low SNR, compensating the phase offsets resulted from channel variations is not viable. We therefore only compensate for the frequency drifts. Specifically, different SFk symbols belonging to the same quasi-SF$(k+m)$ symbol are combined by summing their absolute values. Denote the cumulative amplitude of a frequency tone z of the same quasi-SF symbol by $\tilde{Y}_z$, given coding rate c, $\tilde{Y}_z$ is expressed as:

$$\tilde{Y}_z = \sum_{k=0}^{2^m-1} |Y_{s+ck,z}|. \tag{9}$$

This helps to average out the environmental noise while enhancing the target frequency peak.

5.4 Joint Demodulation and Decoding

SF-any capitalizes on the energy distribution of frequency peaks and the constraints imposed by the Hamming code to perform joint demodulation and decoding of symbols within a group. The underlying insights are as follows:

Observation-1: The majority of symbol frequency peaks are higher than the noise level, which allows for successful decoding. In cases where most symbols cannot be decoded successfully, increasing the quasi-spreading factor is necessary to generate symbols that are more resilient to noise.

Observation-2: Even for symbols with energy lower than some noise peaks, their energy levels should still be significantly higher than the average noise level. If this criterion is not met, it is advisable to raise the quasi-spreading factor for the same reasons.

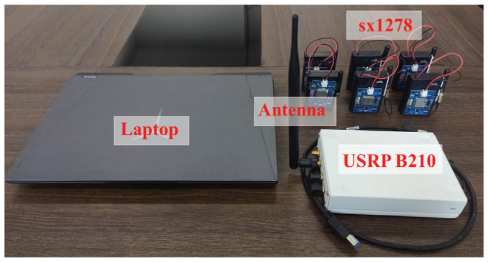


Figure 8: Devices used in experiments.

These observations inspire SF-any to optimize symbol decoding. As a result, we introduce an error-tolerant joint demodulation and decoding algorithm to further enhance packet reception quality. The detailed procedure is as follows: First, two thresholds, denoted by Γ_1 and Γ_2, are initialized subject to $\Gamma_2 < \Gamma_1$. Γ_1 is the energy threshold a peak to be selected directly, and Γ_2 is the energy threshold a peak restored from Hamming constraints to be considered valid. The thresholds keep decreasing until at least four symbols are selected directly. Based on these symbols, all valid symbol combinations which: i) contain four directly selected symbols; ii) comply with the Hamming constraints; and iii) have all peak energy over Γ_2; are examined. The symbol combination with the largest logarithmic sum of peak energy is chosen as the final result. Γ_1 and Γ_2 keep decreasing until a result is found, ensuring a valid output.

6 EVALUATIONS

This sections first describes the experimental setting, and then presents the performance of SF-any.

6.1 Experimental Setup

The transmitter and gateway implementation. We implement SF-any on commercial off-the-shelf (COTS) LoRa transmitters equipped with the SX1278 LoRa chip. The transmitter is controlled by MCU STM32F030F4 and powered by three 1.5V dry batteries. The communication frequency is set at 433MHz, with a bandwidth of 250kHz. The payload size for SF$(12+1)$ to SF$(12+3)$ is set to 30 bytes. The payload size for SF$(12+4)$ and SF$(12+5)$ packets, due to the 255 bytes limit, is set to 12 bytes and 6 bytes, respectively.

To collect the IQ signals, we use a USRP B210 as a gateway. It is connected to a laptop with an R7-5800H CPU, 32GB of RAM, and running Ubuntu 20.04. The received signals are processed in Matlab for further analysis and evaluation.

Schemes to be compared with. We conducted a comparative analysis involving SF-any, the standard SF12 provided by LoRa PHY, Ostinato [26] that introduces the concept of accumulating multiple symbols for SNR enhancement, and CONST [29] that compensates offsets and accumulates signals from multiple gateways.

In Ostinato, a parameter denoted as K is used to represent the repetition rate. Strictly speaking, the highest repetition rate Ostinato can support is 4. When $K = 8$, due to coding constraints, only symbol 0 can be transmitted. In addition, Ostinato is implemented here with the Low Data Rate Optimization (LDRO) mode, as it lacks the capability to track frequency drifts within a packet. In the

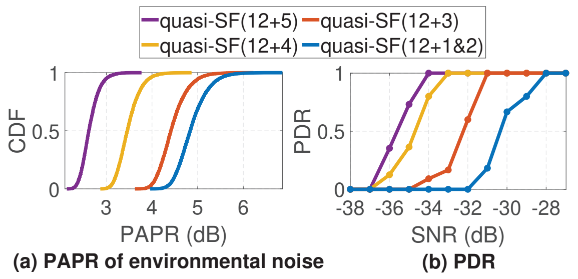


Figure 9: Threshold for packet detection. (a). Peak-to-Average Power Ratio (PAPR) of environmental noise. (b). Packet Detection Rate (PDR) versus SNR for different SF-any packets.

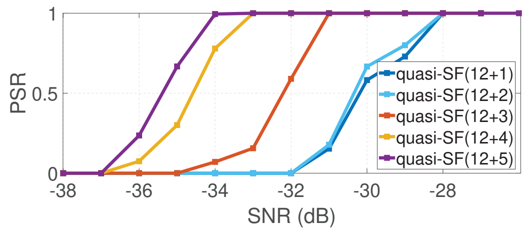


Figure 10: Packet Synchronization Rate (PSR) versus SNR for SF-any packets with various repetition rates.

LDRO mode, the last two bits are disabled to mitigate the effects of frequency drifts. As discussed in Sec. 2.4, the overhead for SF-any is set to be the same as that of Ostinato with SF12.

Regarding CONST, the demodulation procedure consists of three parts. First, the peak energy of each symbol in different gateways is combined coarsely to select a candidate symbol list that belongs to the packet. Second, this symbol list, similar to PILOT symbols in SF-any but with higher uncertainty, is used to estimate and compensate signal offsets in the packet received by each gateway. Finally, the compensated symbols are recombined to determine the final results. To make a fair comparison, the number of gateways involved is equal to the repetition rate in Ostinato and SF-any. Similarly, the positions of the gateways are carefully selected so that the SNR of the packets received from different gateways is close.

In addition, to demonstrate SF-any's compatibility with orthogonal methods, we integrate SF-any with CONST and evaluate the performance of this combined method. Here, the offset estimation and compensation parts of CONST are replaced by SF-any.

Evaluation Scenarios. We evaluate both indoor and outdoor performance for various schemes. The indoor experiments explore the supportable SNR limits of different schemes. The outdoor experiments evaluate SF-any under various channel conditions.

Performance Metrics. We evaluate the Packet Detection Rate (PDR), the Packet Synchronization Rate (PSR), the SFO Estimation Accuracy (SEA) of SF-any. We then evaluate the average Bit Error Rate (BER), the Packet Reception Rate (PRR) and the effective throughput for different schemes under various situations.

6.2 Evaluation Results

We first evaluate SF-any with different SF and various coding rates. The outdoor experiments are then conducted to demonstrate the overall performance of SF-any.

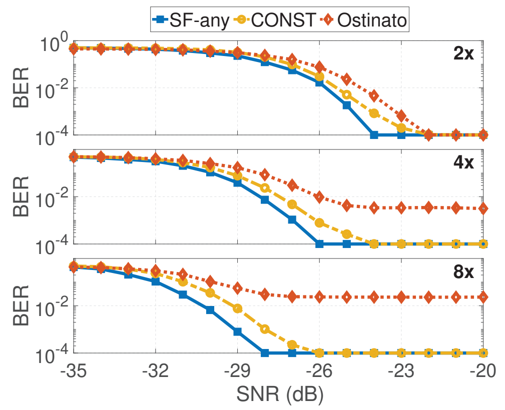


Figure 11: Bit Error Rate (BER) versus SNR performance of SF-any, Ostinato and CONST with 2x, 4x, and 8x repetitions respectively. LDRO mode is enabled in Ostinato.

6.2.1 Detection and synchronization performance. To detect a newly arrived packet, the cumulative amplitude of at least 2^m preambles should exceed a specified SNR margin. In general, the higher the margin, the lower the true positive packets are to be detected, but also the lower the false positive packets (mistakenly determine the noise as the packet), and vice versa.

We first plot the distribution of the Peak-to-Average Power Ratio (PAPR) of the environmental noise so as to study how to prevent false positive from happening. For each quasi-SF situation, the evaluations consist of 10,000 random trials, and the USRP receives environmental noise in each trial. Fig. 9(a) plots the PAPR distribution of the noise. As noise is random, by combining $\max\{6, 2^m\}$ consecutive symbols, the energy of the noise gets averaged out, and the PAPR decreases with higher number of preambles. To maintain a false positive rate below 0.01%, we set the SNR margin to 4dB, 5dB, 6.5dB for quasi-SF$(12+5)$, quasi-SF$(12+4)$, and quasi-SF$(12+3)$ packets respectively. The SNR margin is 7dB for quasi-SF$(12+2)$ and quasi-SF$(12+1)$ packets.

Fig. 9(b) illustrates SF-any's Packet Detection Rate with varying quasi-spreading parameters. For instance, a quasi-SF$(12+5)$ packet with an SNR greater than or equal to -34dB boosts a detection probability exceeding 99.99%. As the SNR decreases, e.g., below -34dB, the packet detection rate decreases. The SNR thresholds for a 99.99% detection probability are -33dB for quasi-SF$(12+4)$, -31dB for quasi-SF$(12+3)$, and -28dB for SF$(12+2)$ and quasi-SF$(12+1)$ packets. It is worth noting that these SNR thresholds remain below the minimum SNR achievable for these quasi-spreading factors, ensuring the efficiency of SF-any's packet detection scheme within the achievable SNR range.

Fig. 10 shows the packet synchronization rate. Note that, the synchronization rate accounts for both successfully detecting the preambles, and decoding the SF-any header. As shown in the figure, once the packets are detected, SF-any can be perfectly synchronized with the introduced SF-any header.

6.2.2 SNR performance with grouped repetition code. We first study the SNR performance without the joint decoding scheme. Fig. 11 compares the BER versus SNR performance of SF-any to that of Ostinato and CONST with various repetition rates (for CONST,

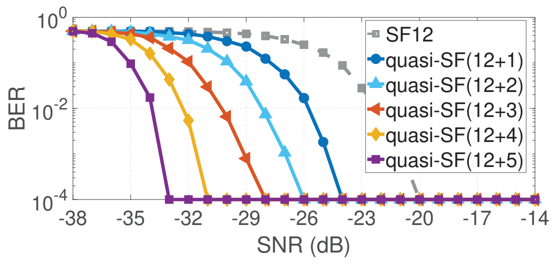


Figure 12: BER versus SNR performance of SF-any and SF12.

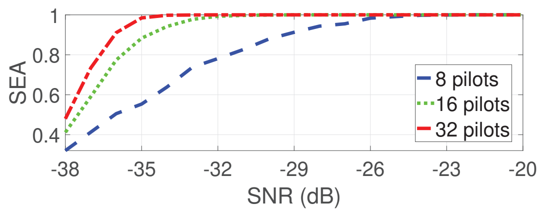


Figure 13: SFO Estimation Accuracy (SEA) of SF-any with different number of pilots.

gateways). Here, to study Ostinato's repetition performance, repetition factor $K = 8$ is implemented by setting all payload symbols to zero. However, it cannot be used to transmit meaningful messages. According to the results, given double repetitions, SF-any achieves a 2dB SNR gain compared with Ostinato and CONST. Moreover, when repetition rate increases, the combining performance of SF-any and CONST scales smoothly as signal offsets are properly compensated for. SF-any achieves a better combining performance of 2dB compared with CONST, because the PILOT strategy used in SF-any estimates the signal offsets by known symbols. CONST, on the other hand, leverages error-prone random symbols for offset estimation, leading to less accurate offset compensation. Ostinato suffers from constant bit errors when its repetition factor $K \geq 4$. This is because a high repetition rate generates long packets, leading to a frequency shift greater than $16/3$ bins at the packet level, which exceeds the tolerable range of LDRO mode.

Fig. 12 further illustrates the combining performance of SF-any with different quasi-SF values as well as that of SF12 of LoRa PHY. Given BER $\leq 10^{-4}$, the supportable SNR for quasi-SF$(12+5)$ packet without joint decoding approaches is -33dB, an SNR improvement of 13dB compared with traditional SF12 packet, and an SNR improvement of more than 11 dB compared with Ostinato. This improvement in combining performance is due to the meticulous compensation for both intra-symbol phase offsets and inter-symbol frequency drifts. As a result, SF-any can tolerate different packet-level frequency drifts, leading to a seamless SNR improvement with increasing repetition rates. Therefore, we argue that SF-any can be extended to large repetition rates, thus effectively enhancing the SNR of an extremely weak link.

As combining performance of SF-any heavily relies on the accurate estimation of signal offsets (especially SFO), we evaluate the estimation accuracy across various SNR levels. We measure the real SFO value in high SNR scenarios (>10dB) and perform 10,000 random trials to measure the SFO Estimation Accuracy (SEA) of LoRa packets for each SNR level. Here each packet contains 256

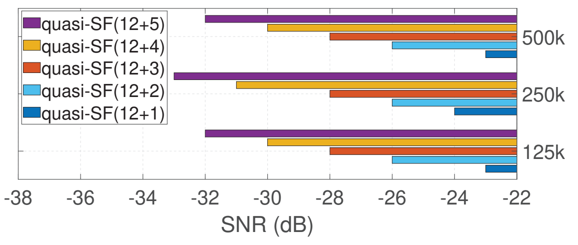


Figure 14: SNR improvement brought by symbol combining under different bandwidths given BER $\leq 10^{-4}$.

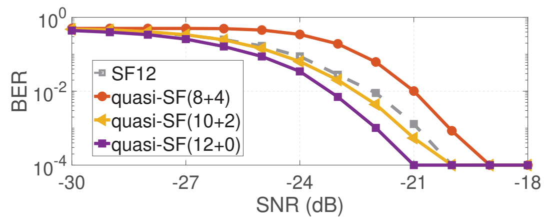


Figure 15: BER versus SNR performance of standard SF12 symbol and simulated quasi-SF12 symbol using 2^0 SF12 symbols, 2^2 SF10 symbols and 2^4 SF8 symbol respectively.

symbols. Specifically, as SFO deviations that cause a packet-level frequency drift within 1 frequency bin still effectively contribute to energy combining, we consider such trials to be **correct**, otherwise **incorrect**. On this basis, the SEA is calculated by the proportion of correct trials. Recall that, As mentioned in Sec. 4.2, we insert 8 pilots for quasi-SF$(12+1)$ packets, 16 pilots For quasi-SF$(12+2)$ packets, and 32 pilots for higher repetitions. Therefore, packets with a higher repetition rate are supposed to support more accurate SFO estimation because more pilots are involved in the SFO estimation. Fig. 13 illustrates the SEA with different number of pilots. Given SEA $\geq 99.99\%$, the supportable SNR limit with 32 pilots reaches -34dB, while that of 16 pilots and 8 pilots is -31dB and -25dB, respectively. Therefore, the SEA at the supportable SNR limit given BER of $\leq 10^{-4}$ remains $\geq 99.99\%$ for all repetition rates. This result demonstrates that SF-any supports an accurate SFO estimation.

We further examine the impact of bandwidth on combining performance of SF-any. Fig. 14 presents a robust scaling performance with various quasi-SF values under different bandwidths, with an up to 1dB performance deviation for the same repetition rate. For each bandwidth setting, given BER $\leq 10^{-4}$, we observe a 2dB to 3dB improvement in performance as the repetition rate doubles. This result demonstrates that SF-any can be effectively applied to various bandwidth settings.

Finally, to verify SF-any's capability of versatile transmission rate, we simulate an SF12 symbol with 2^0 SF12 symbol, 2^2 SF10 symbols, and 2^4 SF8 symbols (denoted by quasi-SF$(8+4)$, quasi-SF$(10+2)$, quasi-SF$(12+0)$ respectively). We then compare the SNR performance of SF-any symbols with that of standard SF12 symbols. As illustrated in Fig. 15, given BER $\leq 10^{-4}$, the supportable SNR limit of standard SF12 symbols is -20dB. quasi-SF symbols achieve a combining performance from -19dB to -21dB, comparable to that of SF12 symbol in LoRa PHY. Moreover, the quasi-SF$(12+0)$ outperforms SF12 of LoRa PHY by 1 dB by compensating for signal

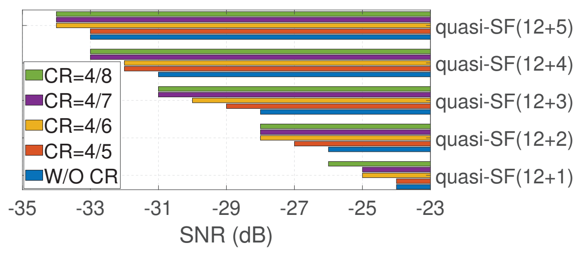


Figure 16: SNR improvement brought by the joint demodulation and decoding scheme given BER $\leq 10^{-4}$.

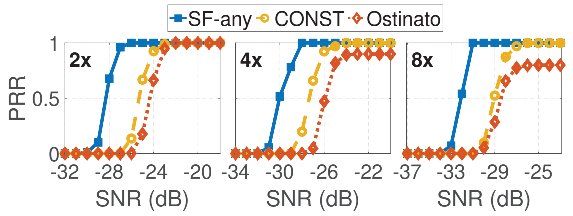


Figure 17: Packet Reception Rate (PRR) versus SNR performance of SF-any (CR4/8), Ostinato and CONST with 2x, 4x, and 8x repetitions respectively.

offsets. The SNR difference between each pair of quasi-SF packets is caused by differnet number of coherently combined chips inside each symbol. E.g, the SF$(8+4)$ scheme coherently combines 256 chips each time and directly adds up the results, while SF$(12+0)$ coherently combines all the 4096 chips, leading to a 2dB gain. This result demonstrates that SF-any can effectively simulate high-SF symbols by combining symbols of lower SF values, thus supporting versatile transmission rate.

6.2.3 SNR performance with joint demodulation and decoding scheme. Based on the grouped repetition code, the proposed decoding scheme further extends the supportable SNR limits for SF-any. Fig. 16 shows the supportable SNR for various quasi-SF values and various coding rates given BER $\leq 10^{-4}$. According to the figure, the proposed decoding scheme achieves an improvement of 1dB to 3dB SNR gain.

Fig. 17 compares the overall PRR (Packet Reception Rate) of Ostinato, CONST and SF-any with different repetition rates. Here, SF-any adopts CR4/8 coding. SF-any outperforms CONST and Ostinato by more than 4dB with the same repetition rate. Ostinato, though remains effective for a lower repetition rate (i.e., $K = 2$) as it yields a 2dB improvement on PRR compared with LoRa, becomes inefficient with higher repetition rates. This inefficiency arises as higher repetition rates significantly increase the packet length, leading to greater packet-level frequency drift.

Fig. 18 further depicts the overall PRR of SF-any as well as the standard LoRa SF12 packet. As the quasi-SF value increases, SF-any's SNR improvement scales smoothly by 1dB to 3dB. When SNR=-34dB, the packet reception rate with quasi-SF$(12+5)$ approaches 1, outperforms the LoRa SF12 packet by 14dB, which is a composed result of aggregated energy with meticulous compensation for signal offsets (14dB) and coding gain (1dB). For CONST, SF-any with the same repetition rate outperforms them by more than 4dB for offset compensation and coding gain. For Ostinato, SF-any with highest repetition rate achieves an up to 12dB improvement contributed by effective energy combining with offsets compensation (11dB), and coding gain (1dB).

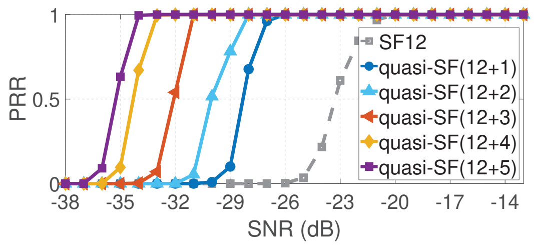


Figure 18: PRR versus SNR performance of SF-any (CR4/8) and SF12 of LoRa PHY.

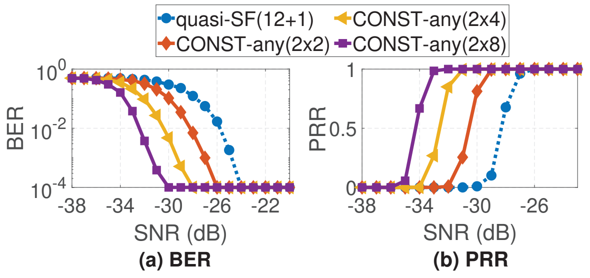


Figure 19: Performance of CONST-any. (a). BER versus SNR (b). PRR versus SNR with CR4/8 coding.

6.2.4 Combining SF-any with Multi-gateway Approach. We evaluate SF-any's performance when combining with the orthogonal method CONST. The combination approach is named **CONST-any** for short. As depicted in Fig. 19(a)[1], given double repetitions, the combined performance scales smoothly with the number of gateways, achieving 2dB gain each time the number of gateways doubles. Even under the SNR of -30dB, CONST-any(2×8) achieves BER of $\leq 10^{-4}$, outperforming quasi-SF$(12+1)$ by 6dB.

The PRR vs SNR performance, as shown in Fig. 19(b), demonstrates the effectiveness of the coding strategy, yielding an additional 1dB to 2dB gain. CONST-any(2×8) with CR4/8 coding achieves a supportable SNR limit of -32dB given PRR of 100%, and 6dB gain compared with quasi-SF$(12+1)$. These results indicate SF-any can be combined with orthogonal methods to perform better.

6.2.5 Outdoor experiments. We next perform outdoor experiments in a 1500 meters × 650 meters campus area (shown in Fig. 20). The gateway is denoted with a yellow star in the figure, and the transmitting nodes (labeled with yellow triangles) are scattered around the gateway. The maximum communication range is about 600 meters. While some nodes are situated in regions with a clear line of sight path to the gateway, the connections of other nodes to the gateway are hindered by buildings, trees, and various impediments. As a result, different links, even when at a similar distance from the gateway, encounter distinctly varied channel conditions.

Specifically, we choose 6 positions for this outdoor experiment, among which position #1, #3 and #6 can establish LOS links with the gateway, position #2 is blocked by two buildings, position #4

[1]The suffix of CONST-any is formatted as repetition rate × number of gateways

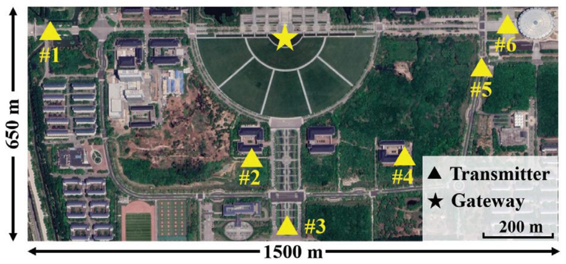


Figure 20: Deployment of the gateway and transmitters.

is inside one building, and position #5 is blocked by a long stretch of woods. We keep the basic experimental parameters same as in indoor experiment, and compare the PRR and throughput performance of SF-any with Ostinato and the standard LoRa packet (SF12). We use quasi-SF$(12+1)$ with CR4/5. The repetition rate K of Ostinato is set to 2 and the LDRO mode is enabled.

As shown in Fig. 21(a) in position #1, #3 and #6, all three schemes achieves perfect PRR performance since LoRa inherently takes a great advantage at long range, especially for LOS communication. in each place, the SNR fluctuates hugely from -15dB to -4dB, partly caused by moving objects such as trucks, cars and passengers.

When obstructed by two buildings at position #2, standard LoRa SF12 packet achieves a PRR of 12% at an SNR of approximately -24dB. Ostinato experiences a 20% PRR degradation (PRR=80%), while SF-any maintains a PRR of 100%. At position #4, the SNR decreases to around -28dB, as multiple concrete walls block the LoRa communication. Due to the weak SF12 symbol energy, LoRa's functionality is severely compromised, with PRR dropping to zero. Ostinato successfully receives 13% of the packets, while SF-any achieves around 90% PRR. The primary reason for the decline in PRR for Ostinato is its ineffective signal combining approach. At position #5, While woods obstruct the line of sight, they do so to a lesser extent than concrete walls, leading to a similarly low SNR (around -24dB) but higher channel variations compared to position #2. PRR performance for standard LoRa and Ostinato is 12% and 37%, respectively. SF-any consistently achieves over 90% PRR for all outdoor transmissions. Ostinato, while improving SNR for some links, exhibits significantly lower PRR for links with SNR $\leq$ -24dB.

Fig. 21(b) shows the throughput of the three schemes. LoRa undoubtedly achieves the highest throughput in LOS scenarios, as it does not suffer from PRR loss and has no repetition cost. Osinato and SF-any achieve about half the throughput as LoRa. Nevertheless, when SNR is low, the advantage of LoRa on transmission rate vanishes as it suffers from a poor PRR. The throughput of Ostinato and SF-any is better because they are more resilient to noise, while SF-any is notably more robust and remains stable consistently.

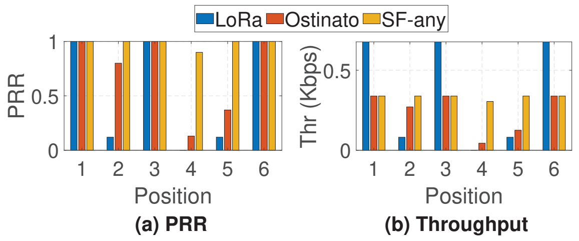


Figure 21: Performance of SF-any in outdoor environments. (a). PRR performance and (b). throughput performance at different positions.

7 RELATED WORKS

Communication on weak links is an important issue in reliable LoRa communications. Generally, current works can be classified into two categories: (a), enhancing signal quality in physical layer, and (b), repairing corrupted information from coding redundancy.

Methods for enhancing signal quality. These methods exploit the diversity of received signals, constructively adding up these signals helps to average out the noise while enhancing target signals. Charm [7] collects multiple signal copies from different gateways and combines them in the cloud. However, it suffers from high edge-cloud communication costs and non-coherent signal combinations. Nephalai [15] designs a compressive sensing algorithm to compress IQ signals, alleviating upload pressure. CONST [29] coherently combines signals from different gateways, suppressing the power of noise rapidly while aggregating the signal power. MALoRa [9] recovers weak signals by coherently combining signals received by synchronized antennas. Choir [8] suggests simultaneously sending similar data by multiple transmitters to improve the received energy. However, synchronization among transmitters is a great challenge. Ostinato [26] and Xcopy [25], however, leverage the time diversity for signal enhancement, avoiding additional hardware costs. However, as discussed, Ostinato struggles to scale due to stringent hardware imperfections. Xcopy resolves low-SNR packets reception problem via multiple retransmissions. This incurs extremely long delays under the 1% duty cycle imposed by EU regulation [2], potential collisions due to the cumulative length of all retransmitted packets, and violation of LoRa MAC-layer protocol. In contrast, SF-any implements repetition coding within one packet, which supports the repetition rate up to 32 given the max length of 255 Bytes specified in LoRa PHY [21]. This not only avoids the long delay and extra overhead of retransmissions but also complies with LoRa MAC-layer protocol. Moreover, SF-any harnesses the time diversity, provided by the repetition within a long packet, for better demodulation and decoding.

Methods based on coding. These methods recover corrupted information from coding redundancy. DaRe [17] designs a combination of Convolution code and Fountain code to perform rateless coding. OPR [3] votes for payload bits with RSSI information for packet received by multiple gateways. Similar to OPR, ECRLoRa [18] selects candidate packet segments at gateway, performing group weighted voting to recover corrupted data. LLDPC [27] designs a low density parity check coding scheme to build constraints among payload bits, and decodes with the LLR information using a graph neural network. DC [28] combines Reed-Solomon codes with LoRa FEC mechanism to protect error-prone bytes in a LoRa packet. These coding methods struggle at extremely low SNR levels as noise corrupts most symbols, leaving little usable redundancy.

Beside obtaining various gains from extra hardware or protocol designs, NELoRa [13] designs a Deep Neural Network to distinguish the time-frequency pattern of LoRa symbol from background noise in spectrumgram. Falcon [23] deliberately interferes or maintains on-going LoRa packets with another transmitter to modulate binary bits, completely sacrificing data rates. A similar work [22] also leverages coherent interference among signals to support long-range cross-technology communication.

8 DISCUSSION AND FUTURE WORK

Compatibility with Standard LoRa. At the heart of SF-any is restructuring the LoRa coding and decoding process, implemented via only software updates without hardware modification. SF-any could be built into a LoRa gateway's software stack, which supports both SF-any and standard LoRa reception. Under a high-SNR scenario, loRa packets are decoded via a standard LoRa PHY receiver. In contrast, under the extremely low SNR scenario, SF-anny is activated for reliable reception.

Effect of Hardware with Higher Precision. In fact, using high-precision crystals, VCOs, and a higher sampling rate benefits signal combining as these methods reduce the STO significantly. As a result, SF-any's performance would be further improved with the hardware upgrade, which would be considered in our future work.

9 CONCLUSION

In this paper, we present SF-any, a software-based coding paradigm, that enables modulating a LoRa packet with any spreading factor and compatible with orthogonal methods, facilitating reliable city-wide communications. SF-any employs 2^m SFk symbols to emulate a quasi-SF$(k+m)$ symbol, incorporating pilot symbols to compensate for time-varying frequency drifts and phase offsets. It also introduces a grouped repetition code for SF-any symbol modulation and utilizes a joint demodulation and decoding scheme to seamlessly integrate energy distribution into the decoding process. This transformation effectively strengthens the Hamming decoding, significantly improving packet reception rates, especially for ultra-low SNR packets. Extensive evaluations showcase SF-any's exceptional improvement by up to 14dB over SF12 of LoRa PHY. Moreover, SF-any outperforms the leading solutions for PHY layer reliable communications, achieving an improvement of up to 12dB compared with Ostinato, and 4dB to 5dB improvement over CONST with same repetitions. Importantly, SF-any is the first work to consistently deliver high performance across various spreading factors.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62172092 and the National Natural Science Foundation of China under Grant No. 62272098.

REFERENCES

[1] LoRa Alliance. 2020. TS001-1.0.4 LoRaWAN L2 1.0.4 Specification. (2020).

[2] LoRa Alliance. 2022. LoRaWAN® regional parameters rp002-1.0.4. (2022).

[3] Artur Balanuta, Nuno Pereira, Swarun Kumar, and Anthony Rowe. 2020. A cloud-optimized link layer for low-power wide-area networks. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 247–259.

[4] Gonglong Chen, Jiamei Lv, and Wei Dong. 2020. Exploiting rateless codes and cross-layer optimization for low-power wide-area networks. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*. 1–9.

[5] Weiwei Chen, Shuai Wang, and Tian He. 2023. Magnifier: leveraging the fine-grained hardware information and their temporal patterns for concurrent LoRa decoding. *IEEE Transactions on Mobile Computing* (2023), 1–14.

[6] Shilpa Devalal and A. Karthikeyan. 2018. LoRa Technology - An Overview. In *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. 284–290.

[7] Adwait Dongare, Revathy Narayanan, Akshay Gadre, Anh Luong, Artur Balanuta, Swarun Kumar, Bob Iannucci, and Anthony Rowe. 2018. Charm: exploiting geographical diversity through coherent combining in low-power wide-area networks. In *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 60–71.

[8] Rashad Eletreby, Diana Zhang, Swarun Kumar, and Osman Yağan. 2017. Empowering low-power wide area networks in urban settings. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. 309–321.

[9] Ningning Hou, Xianjin Xia, and Yuanqing Zheng. 2023. Don't miss weak packets: Boosting LoRa reception with antenna diversities. *ACM Transactions on Sensor Networks* (2023), 1–25.

[10] Bin Hu, Zhimeng Yin, Shuai Wang, Zhuqing Xu, and Tian He. 2020. SCLoRa: Leveraging multi-dimensionality in decoding collided LoRa transmissions. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. 1–11.

[11] Mookeun Ji, Juyeon Yoon, Jeongwoo Choo, Minki Jang, and Anthony Smith. 2019. Lora-based visual monitoring scheme for agriculture iot. In *2019 IEEE sensors applications symposium (SAS)*. 1–6.

[12] Jinyan Jiang, Zhenqiang Xu, Fan Dang, and Jiliang Wang. 2021. Long-range ambient LoRa backscatter with parallel decoding. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 684–696.

[13] Chenning Li, Hanqing Guo, Shuai Tong, Xiao Zeng, Zhichao Cao, Mi Zhang, Qiben Yan, Li Xiao, Jiliang Wang, and Yunhao Liu. 2021. NELoRa: Towards ultra-low SNR LoRa communication with neural-enhanced demodulation. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 56–68.

[14] Zhijun Li and Tian He. 2017. Webee: Physical-layer cross-technology communication via emulation. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. 2–14.

[15] Jun Liu, Weitao Xu, Sanjay Jha, and Wen Hu. 2020. Nephalai: towards LPWAN C-RAN with physical layer compression. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–12.

[16] Wenliang Mao, Zhiwei Zhao, Kaiwen Zheng, and Geyong Min. 2023. Recovering Packet Collisions below the Noise Floor in Multi-gateway LoRa Networks. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. 1–10.

[17] Paul J Marcelis, Vijay Rao, and R Venkatesha Prasad. 2017. DaRe: Data recovery through application layer coding for LoRaWAN. In *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*. 97–108.

[18] Luoyu Mei, Zhimeng Yin, Shuai Wang, Xiaolei Zhou, Taiwei Ling, and Tian He. 2023. ECRLoRa: LoRa Packet Recovery under Low SNR via Edge-Cloud Collaboration. *ACM Transactions on Sensor Networks* (2023), 1–25.

[19] Jithu G Panicker, Mohamed Azman, and Rajiv Kashyap. 2019. A LoRa wireless mesh network for wide-area animal tracking. In *2019 IEEE International Conference on Electrical, Computer and Communication Technologies*. 1–5.

[20] Marko Radeta, Miguel Ribeiro, Dinarte Vasconcelos, Hildegardo Noronha, and Nuno Jardim Nunes. 2020. LoRaquatica: Studying range and location estimation using LoRa and IoT in aquatic sensing. In *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 1–6.

[21] Semtech. 2020. SX1276/77/78/79—137 MHz to 1020 MHz Low Power Long Range Transceiver. (2020).

[22] Shuai Tong, Yangliang He, Yunhao Liu, and Jiliang Wang. 2022. De-spreading over the air: long-range CTC for diverse receivers with LoRa. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 42–54.

[23] Shuai Tong, Zilin Shen, Yunhao Liu, and Jiliang Wang. 2021. Combating link dynamics for reliable lora connection in urban settings. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 642–655.

[24] D Tse. 2005. Fundamentals of Wireless Communication. *Cambridge University Press google schola* (2005).

[25] Xianjin Xia, Qianwu Chen, Ningning Hou, Yuanqing Zheng, and Mo Li. 2023. XCopy: Boosting Weak Links for Reliable LoRa Communication. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.

[26] Zhenqiang Xu, Pengjin Xie, Jiliang Wang, and Yunhao Liu. 2022. Ostinato: Combating LoRa Weak Links in Real Deployments. In *2022 IEEE 30th International Conference on Network Protocols (ICNP)*. 1–11.

[27] Kang Yang and Wan Du. 2022. LLDPC: A low-density parity-check coding scheme for LoRa networks. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*. 193–206.

[28] Niloofar Yazdani, Nikolaos Kouvelas, Daniel E Lucani, and R Venkatesha Prasad. 2022. Divide and Code: Efficient and Real-time Data Recovery from Corrupted LoRa Frames. In *2022 19th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 235–243.

[29] Zeyu Zhang, Weiwei Chen, Junwen Wang, Shuai Wang, and Tian He. 2022. CONST: Exploiting Spatial-Temporal Correlation for Multi-Gateway based Reliable LoRa Reception. In *2022 IEEE 30th International Conference on Network Protocols (ICNP)*. 1–11.