Concurrent Interference Cancellation : Decoding Multi-Packet Collisions in LoRa

Muhammad Osama Shahid
University of Wisconsin-Madison
mshahid2@wisc.edu

Millan Philipose
University of Washington
millan.philipose@gmail.com

Krishna Chintalapudi
Microsoft Research
krchinta@microsoft.com

Suman Banerjee
University of Wisconsin-Madison
suman@cs.wisc.edu

Bhuvana Krishnaswamy
University of Wisconsin-Madison
bhuvana@ece.wisc.edu

ABSTRACT

LoRa has seen widespread adoption as a long range IoT technology. As the number of LoRa deployments grow, packet collisions undermine its overall network throughput. In this paper, we propose a novel interference cancellation technique – Concurrent Interference Cancellation (CIC), that enables concurrent decoding of multiple collided LoRa packets. CIC fundamentally differs from existing approaches as it demodulates symbols by canceling out all other interfering symbols. It achieves this cancellation by carefully selecting a set of sub-symbols – pieces of the original symbol such that no interfering symbol is common across all sub-symbols in this set. Thus, after demodulating each sub-symbol, an intersection across their spectra cancels out all the interfering symbols. Through LoRa deployments using COTS devices, we demonstrate that CIC can increase the network capacity of standard LoRa by up to 10× and up to 4× over the state-of-the-art research. While beneficial across all scenarios, CIC has even more significant benefits under low SNR conditions that are common to LoRa deployments, in which prior approaches appear to perform quite poorly.

CCS CONCEPTS

• **Networks** → **Link-layer protocols**; *Network protocol design*;

KEYWORDS

LoRa, Multi-packet collisions, Interference Cancellation

ACM Reference Format:
Muhammad Osama Shahid, Millan Philipose, Krishna Chintalapudi, Suman Banerjee, and Bhuvana Krishnaswamy. 2021. Concurrent Interference Cancellation : Decoding Multi-Packet Collisions in LoRa. In *ACM SIGCOMM 2021 Conference (SIGCOMM '21), August 23–28, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3452296.3472931

1 INTRODUCTION

Deployed globally, LoRa [1] has emerged as a dominant IoT connectivity technology, enabling applications such as smart cities [2, 3], smart agriculture [4, 5], and industrial IoT [6, 7]. In a LoRa deployment, IoT devices, spread across a large spatial extent (several sq. km.), transmit uplink messages to a LoRa gateway. These messages are typically transmitted in response to physical sensory events at unpredictable times. Given their large range, several independently administered LoRa networks often interfere with each other [8]. Consequently, as LoRa gains popularity, packet collisions in LoRa networks significantly undermine their capacity [9, 10].

SIGCOMM '21, August 23–28, 2021, Virtual Event, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8383-7/21/08…$15.00
https://doi.org/10.1145/3452296.3472931

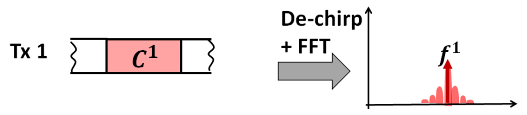


Fig 1: Standard LoRa decoding under no collisions – each symbol maps to one unique frequency

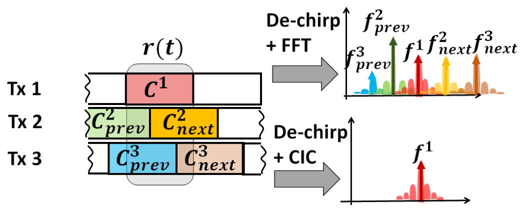


Fig 2: Standard LoRa decoding under collisions results in multiple frequency peaks causing confusion. CIC removes these interfering frequency peaks.

LoRa uses Chirp Spread Spectrum (CSS) modulation, where the data symbols (each data symbol encodes multiple bits) are transmitted as chirps – signals with linearly increasing frequency. Use of chirps enables long distance communication and provides immunity against interference from other transmissions [11]. The standard Commercial Off The Shelf (COTS) LoRa receiver can only decode one packet at a time. Recently, several research efforts have attempted to exploit LoRa's underlying robustness to interference to resolve collisions, thereby improving network capacity [12–14].

The key contribution of this paper is a novel demodulation technique, Concurrent Interference Cancellation (CIC)[1]*, that enables decoding of multiple colliding packets from COTS LoRa devices.* Our deployments indicate that CIC can increase network capacity by a factor of 10× compared to LoRa gateways and 4× compared to the state of the art in research.

[1]Unlike Successive Interference Cancellation that iteratively decodes and cancels packets with varying power levels, CIC decodes symbols in parallel, unaffected by variations in signal power levels.

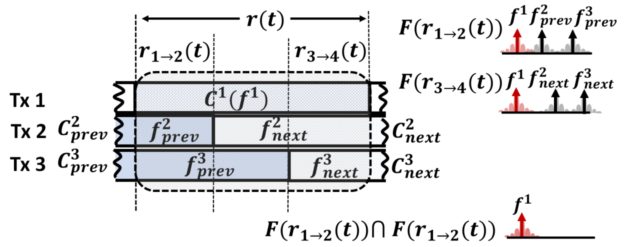


Fig 3: An example illustration of CIC

A standard LoRa receiver "de-chirps" each received symbol by multiplying it with a down-chirp, a chirp of linearly decreasing frequency. De-chirping converts a LoRa symbol into a sinusoid with a fixed frequency, unique to that symbol. Fourier transform (FFT) is then used to identify this frequency and hence the symbol. Figure 1 illustrates the decoding of symbol C^1 of a collision-free transmission (Tx 1), whose de-chirping generates a sinusoid of frequency f^1.

In a multi-packet collision, the received signal is a superposition of several interfering transmissions. De-chirping the received symbol $r(t)$, results in a superposition of multiple sinusoids (instead of one), each with a frequency corresponding to one of the interfering symbols. An FFT of this received symbol then results in a clutter of multiple frequencies. The standard LoRa receiver is unable to determine which of these frequencies corresponds to that of the symbol being decoded. Figure 2 illustrates this for a 3 packet collision scneario.

Existing techniques treat multi-packet collision decoding as a matching problem, where detected peaks are matched to their corresponding transmitters (as discussed in Section 2) based on features unique to symbols originating from a single transmitter. Choir [14] exploits the uniqueness of the transmitters' Carrier Frequency Offsets (CFO), mLoRa [15] and CoLoRa [16] group symbols based on the similarity in their received power levels. FTrack [12] groups symbols based on their start times using a sliding STFT.

CIC takes a distinct approach to multi-packet collision decoding. Rather than treating it as a symbol-to-transmitter matching problem, it cancels out all interfering symbol frequencies, leaving behind only a single frequency peak, that of the symbol being decoded (Figure 2). In order to achieve this cancellation, CIC exploits variations in spectral content across various sub-parts of the symbol.

For every interfering transmission Tx i, as it transitions across its symbol boundary, the previous interfering symbol C^i_{prev} is replaced by the next interfering symbol C^i_{next}. Between any two interfering symbol transitions however, the set of interfering symbols does not change. Based on this observation, CIC defines *sub-symbols* i.e. pieces of $r(t)$, whose boundaries align with symbol boundaries of transmissions. CIC determines these boundaries by determining the start of each received LoRa packet using preamble detection. Since symbol duration is known for a LoRa network, we extrapolate the start of each packet to determine the corresponding symbol boundary. For example, in Figure 3, $r_{1\to2}(t)$ represents the part of $r(t)$ from the start of C^1 to the start of C^2_{next} and $r_{3\to4}(t)$ from the start of C^3_{next} to the end of C^1. CIC then selects the *Interference Cancelling Sub-Symbols Set* (ICSS), a set of different sub-symbols, such that there is no interfering symbol that is common across all of them. In our example, $\{r_{1\to2}(t), r_{3\to4}(t)\}$ is an ICSS. $r_{1\to2}(t)$ experiences interference from symbols $\{C^2_{prev}, C^3_{prev}\}$, since none of the interfering transmissions have crossed their respective symbol boundaries. Similarly, $r_{3\to4}(t)$ experiences interference from symbols $\{C^2_{next}, C^3_{next}\}$, since all of the interfering transmissions have crossed their respective symbol boundaries. The only common symbol to both sub-symbols is C^1 and there are no common interfering symbols.

CIC then de-chirps each of the sub-symbols in the ICSS by computing an FFT. There will be no interfering frequency peak that is common across all these FFTs, since they have no common interfering symbol by construction. Next, CIC performs a *spectral intersection* (Section 5.2) of all these FFTs which is an equivalent of set intersection for spectra. This operation removes all the interfering symbol frequencies, leaving behind only one frequency corresponding to the symbol to be decoded, as illustrated in Fig. 3.

Choosing shorter time-span sub-symbols however, comes with a drawback. Heisenberg's Time-Frequency uncertainty principle [17] dictates that using shorter time-span windows will adversely effect frequency resolution in the FFT. At lower resolutions, cancellation is harder as close peaks merge or overlap with one another. Thus, different choices of ICSS provide different levels of cancellation. Since $r_{1\to2}(t)$ and $r_{3\to4}(t)$ each have a small time span, this choice does not provide the best cancellation. As described in Section 5, guided by the Heisenberg's uncertainty principle, CIC chooses the optimal ICSS that maximizes the cancellation.

In cases where two or more of the interfering transmissions are received at a high power and have symbol boundaries close in time and in constituent frequencies, CIC may only be able to cancel them partially. To resolve such cases we propose a novel technique, Spectral Edge Difference (SED), that relies on the property that while the constituent frequency of the symbol being decoded will be present at both ends of the symbol, this will not be true for interfering transmissions. We find that using additional features unique to each transmitter, such as Carrier Frequency Offset (as proposed in [14]) and received signal power (as employed in [16]) to filter possible candidates improves performance.

We have implemented CIC-based decoding as a python library, and also in Matlab, all available at [2]. A key feature of CIC is that it can be decoded symbol-by-symbol for each transmission independently, making it extremely parallalizable and amenable to efficient multi-threaded implementation. We have tested the efficacy of CIC in various indoor and outdoor environments, while comparing its performance to standard LoRa as well as prior state of the art. Our deployment tests indicate that CIC significantly outperforms existing techniques in both high SNR scenarios as well in large area deployments where the received signal strength can be close or even under the noise floor.

In summary, we make the following contributions.

- We propose a novel LoRa demodulation technique, Concurrent Interference Cancellation (CIC), that can decode multiple concurrent LoRa transmissions by canceling symbols from interfering transmissions.

[2]https://github.com/osama4933/CIC

- Through several indoor and outdoor deployments using COTS devices we show that CIC significantly outperforms LoRa (10×) as well as state of the art collision decoding techniques.
- We provide python as well as Matlab implementations of CIC for public use.

2 RELATED WORK

As discussed in Section 1, during a packet collision, a LoRa decoder sees multiple peaks instead of one (Figure 2), each corresponding to an interfering symbol. *The approach taken by all existing works has been to match and group all discovered symbols to their corresponding transmitters based on certain features common to them.*

Choir [14] exploits the fact that different transmitters have different Carrier Frequency Offsets (CFO) and groups symbols with the most similar CFOs. FTrack [12] generates time-frequency tracks of the various symbols in the received signal by employing sliding window Short Term Fourier Transforms (STFT) on the de-chirped signal. It uses a fixed window size of one symbol duration while sliding, due to which FTrack achieves best possible frequency resolution at the cost of worst time resolution. FTrack performs well in high SNR scenarios but its accuracy degrades in low SNR since its thresholds fail to extract accurate frequency tracks from a noisy spectrogram. Moreover, sliding the window sample by sample incurs computational overheads. Following FTrack, more recent works have also leveraged the packet structure and cyclicity of CSS to estimate the start of a message and decode multiple packets from the collided signal [13, 16]. mLoRa [15] and CoLoRa [16] group symbols based on the similarity in their received power levels. mLoRa proposes a Successive Interference Cancellation (SIC) technique that iteratively assigns the highest powered symbols to a single transmission and then removes them from consideration for the next iteration. CoLoRa [16] groups symbols with similar peak(power) ratios across windows to a transmission with the assumption that the received power is consistent throughout the packet. NScale [13] focuses on collision resolution in low-SNR conditions using non-stationary scaling to match data symbols to packets. NScale achieves the symbol error rate of FTrack over a range of SNRs.

In contrast to past approaches, CIC cancels out the interference by combining spectra obtained from different parts within each symbol. One of the major differences between existing works and CIC is the use of temporal variation of spectral content of a single symbol. Majority of the existing works use a fixed, large window size to perform FFT; although that leads to high frequency resolution, longer windows have limited time resolution. Therefore, timing information of interfering symbols is unavailable in current works. CIC chooses an optimum set of interference cancelling sub-symbols that provides both time and frequency resolution.

3 LORA DECODING BACKGROUND

In this section we provide the necessary background on LoRa PHY layer required in the rest of the paper.

Chirp Modulation in LoRa. In LoRa, every data symbol C_ϕ is derived by shifting the fundamental symbol C_0 by a frequency f_ϕ. C_0 is a chirp with its instantaneous frequency increasing linearly with time t from $-\frac{B}{2}$ to $\frac{B}{2}$ over a symbol duration T_s, where B is the bandwidth of transmission.

$$C_\phi(t) = C_0(t) \cdot e^{j2\pi f_\phi t}; \tag{1}$$

$$C_0(t) = \begin{cases} e^{j2\pi\left(0.5\frac{B^2}{2^{SF}}t^2 - \frac{B}{2}t\right)}, & 0 \le t \le T_s \\ 0, \text{otherwise} \end{cases} \tag{2}$$

In Eqn 1, $\phi \in \{0, 1, \cdots, 2^{SF} - 1\}$ and $T_s = \frac{2^{SF}}{B}$. The spreading factor, $SF \in \{7, 8, 9, 10, 11, 12\}$, fixed for each LoRa packet, dictates the transmission data rate. Using a larger SF extends transmission range at the cost of using a lower data rate.

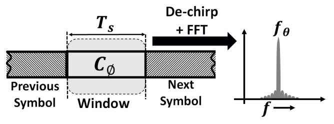


Fig 4: Demodulation in LoRa

Demodulation Using De-chirping in Standard LoRa. To demodulate a symbol, the LoRa demodulator must estimate f_ϕ. A LoRa receiver first de-chirps a symbol by multiplying it with *down-chirp* $C_0^*(t)$, the complex conjugate of C_0, over a window aligned with the symbol's boundaries (Eqn 3). De-chirping converts the symbol into a sinusoid of constant frequency f_ϕ. f_ϕ is recovered by locating the peak in an FFT of the de-chirped symbol (Figure 1).

$$C_\phi(t)C_0^*(t) = e^{j2\pi f_\phi t} \tag{3}$$

$$\Phi(C_\phi(t)) = FFT(C_\phi(t)C_0^*(t)) = sinc(T_s(f - f_\phi)) \tag{4}$$

In Eqn 4, $\Phi(C_\phi(t))$ represents de-chirping and a 2^{SF} point FFT over a symbol $C_\phi(t)$.

Carrier Frequency Offset (CFO). Carrier Frequency Offset (CFO), δf, is the small difference in generated carrier frequencies between the transmitter and the receiver due to manufacturing imperfections. CFO manifests itself as a constant shift in the estimated symbol frequency so that the peak of a symbol C_ϕ would be located as $f_\phi + \delta f$. Consequently, receivers must estimate and subtract δf

Packet Detection using the LoRa Preamble. Before the receiver can begin demodulating, it must first reliably detect the onset of a new transmission, determine the exact positions of the boundaries of the symbols within the packet in order to perform de-chirping and FFT and, estimate f_ϕ. A preamble preceding the data symbols, facilitates all these functions(Fig. 5). The LoRa preamble comprises a sequence of 8 consecutive C_0 symbols, followed by two SYNC symbols C_x, C_y ($x \ne 0, y = x + 8$) and 2.25 down-chirps C_0^*. To detect a new transmission, the receiver continuously de-chirps and performs an FFT until it finds 8 consecutive peaks with the same frequency. The SYNC words and down-chirps then help locate the packet's symbol boundary positions, δf estimation and to confirm onset of a new packet.

4 LORA DECODING UNDER COLLISIONS

In this section we extend the discussion in Section 3 to a multi-packet collision scenario.

LoRa Demodulation during a collision. During a packet collision (Fig. 8), the received signal is a superposition of multiple LoRa receptions, each starting at a different time, with a different symbol

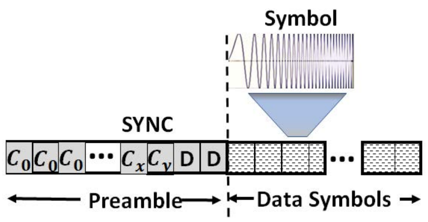


Fig 5: A LoRa Packet

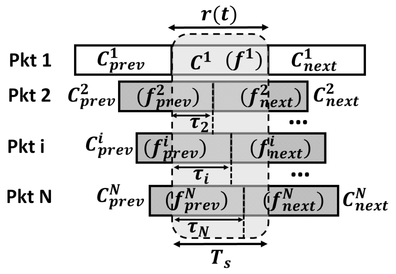

$r(t)$
Pkt 1 C^1_{prev} C^1 (f^1) C^1_{next}
Pkt 2 C^2_{prev} (f^2_{prev}) (f^2_{next}) C^2_{next}
τ_2
Pkt i C^i_{prev} (f^i_{prev}) (f^i_{next})
τ_i
Pkt N C^N_{prev} (f^N_{prev}) (f^N_{next}) C^N_{next}
τ_N
T_s

Fig 6: Collision

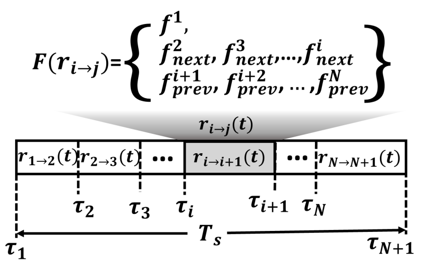

$F(r_{i\to j})=\left\{ f^1, f^2_{next}, f^3_{next}, \dots, f^i_{next}, f^{i+1}_{prev}, f^{i+2}_{prev}, \dots, f^N_{prev} \right\}$
$r_{i\to j}(t)$
$r_{1\to 2}(t)$ $r_{2\to 3}(t)$ ••• $r_{i\to i+1}(t)$ ••• $r_{N\to N+1}(t)$
τ_2 τ_3 τ_i τ_{i+1} τ_N
τ_1 T_s τ_{N+1}

Fig 7: CIC - Sub-Symbol Sections

boundary and at a different received power level. Fig. 6 illustrates the demodulation of a LoRa symbol C^1 from transmitter 1 during an N packet collision. Since symbol boundaries of colliding transmissions are not aligned, C^1 overlaps partially with two consecutive symbols from each interfering transmissions – C^i_{prev} and C^i_{next} from transmitters $2 \le i \le N$ corresponding to de-chirped frequencies f^i_{prev} and f^i_{next} respectively. τ_i depicts the time difference of symbol boundaries between 1st and i^{th} transmission. We assume that the receiver has accurately determined the symbol boundaries as well as the CFO for transmitter 1 using preamble detection (as described in Section 5.8) δ^i represents the difference of the i^{th} transmitter's CFO from that of transmitter 1. The received signal symbol $r(t)$ (with $t = 0$ at the start of C^1), during the collision is given by,

$$r(t) = A_1 C^1_{\phi(\tau)}(t) + I(t) \tag{5}$$

$$I(t) = \sum_{i=2}^{i=N} A_i e^{j2\pi\delta^i t} \left[\begin{array}{l} C^i_{prev}(t + T_s - \tau_i) W(\frac{t}{\tau_i}) \\ +C^i_{next}(t - \tau_i) W(\frac{t-\tau_i}{T_s-\tau_i}) \end{array} \right] \tag{6}$$

$$W(t) = \begin{cases} 1 & if \quad 0 \le t \le 1 \\ 0 & otherwise \end{cases} \tag{7}$$

In Eqn 5, $I(t)$ is the interference. The window functions $W(\frac{t}{\tau_i})$ and $W(\frac{t-\tau_i}{T_s-\tau_i})$ represent the partial overlap regions of C^1_{prev} and C^1_{next} with C^1 and have widths $T_s - \tau_i$ and τ_i (Figure 6) respectively. A_i is received signal amplitude from i^{th} transmitter. Demodulating $r(t)$ by de-chirping and FFT, results in $2(N-1)$ peaks corresponding to the $2(N-1)$ partially overlapping interfering symbols, C^i_{prev} and C^i_{next} and one peak corresponding to symbol C^1.

$$\Phi(r(t)) = A_1 sinc\left(T_s(f - f^1)\right) + I(f) \tag{8}$$

$$I(f) = \sum_{i=2}^{i=N} \begin{array}{l} \frac{A_i}{\tau_i} sinc\left(\tau_i(f - f^i_{prev} - \Delta f_i - \delta^i)\right) \\ + \frac{A_i}{T_s-\tau_i} sinc\left((T_s - \tau_i)(f - f^i_{next} - \Delta f_i - \delta^i)\right) \end{array} \tag{9}$$

$$\Delta f_i = \tau_i \left(\frac{B}{2^{SF}}\right) \tag{10}$$

In Eqn 8 $I(f)$ is the interference, f^1,f^i_{next}, and f^i_{prev} are the chirp start frequencies for C^1,C^i_{prev} and, C^i_{next}. Eqn 8, shows that the height of the interfering peaks are $\frac{A_i}{T_s-\tau_i}$ and $\frac{A_i}{\tau_i}$ depending on the received power as well as the time difference in symbol boundaries τ_i. Fig. 12 depicts LoRa demodulation of 6 colliding transmissions (at SF=8) using COTS LoRa devices. Red circles indicate interfering transmissions while the green circle indicates the true peak. In Fig. 12, there are 3 peaks higher than the true peak corresponding

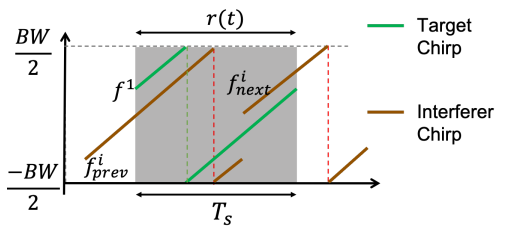


Fig 8: Collisions : Illustration

to the symbol being decoded. This occurs because, the received powers of the interfering transmissions can be stronger than the one being decoded.

5 CIC

As described in Section 4, in the event of an N packet collision, the received signal of the symbol being decoded, $r(t)$ (Eqn 5), comprises a superposition of $2(N-1)$ interfering symbols in addition to the symbol of interest (C^1). Consequently, the de-chirped signal comprises $2N - 1$ frequencies instead of one. CIC exploits temporal variations in spectral content of the symbol as interfering transmissions transition through their respective symbols. *The key observation that drives the design of CIC is that none of the interfering symbols span the entire symbol duration while C^1 does.*

CIC selects a specific set of sub-symbols (parts of the symbol being decoded) such that none of the interfering symbols is common across all these sub-symbols. Such a set of sub-symbols is deemed the Interference Cancelling Sub-Symbol Set (ICSS). This means that the only common frequency across all sub-symbols in an ICSS will be C^1, as it is present in all sub-symbols of $r(t)$. By estimating spectra for all sub-symbols in the ICSS and extracting the common frequency, CIC removes all interfering symbols while retaining the frequency f^1 corresponding to C^1.

Sub-Symbols. A sub-symbol $r_{i\to j}(t)$ is a part of the received symbol $r(t)$ that start at $t = \tau_i$ and ends at $t = \tau_j$. Recall that $t = \tau_i$ is the symbol boundary of the i^{th} transmission when it transitions from C^i_{prev} to C^i_{next} (Section 4,Figure 6).

$$r_{i\to j}(t) = \begin{cases} r(t) & if \quad \tau_i < t < \tau_j \\ 0 & otherwise \end{cases} \Bigg| \tau_1 = 0, \tau_{N+1} = T_s \tag{11}$$

As an example, Figure 7 depicts the set of sub-symbols $r_{i\to i+1}(t)$. The key property common to these sub-symbols is that each of the sub-symbols comprises exactly $N - 1$ interfering symbols –

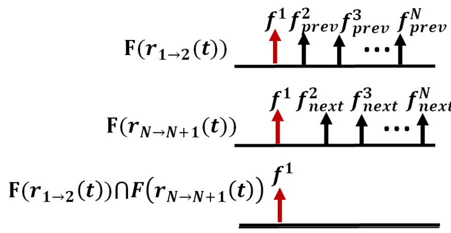

$F(r_{1\to2}(t))$
$f^1\ f^2_{prev}\ f^3_{prev}\ f^N_{prev}$
$F(r_{N\to N+1}(t))$
$f^1\ f^2_{next}\ f^3_{next}\ f^N_{next}$
$F(r_{1\to2}(t))\cap F(r_{N\to N+1}(t))$
f^1

Fig 9: Strawman CIC

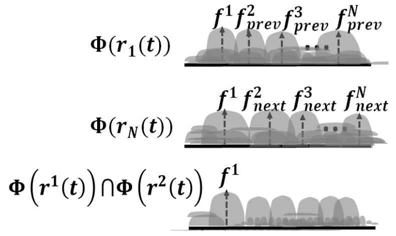

$\Phi(r_1(t))$
$f^1\ f^2_{prev}\ f^3_{prev}\ f^N_{prev}$
$\Phi(r_N(t))$
$f^1\ f^2_{next}\ f^3_{next}\ f^N_{next}$
$\Phi\left(r^1(t)\right)\cap\Phi\left(r^2(t)\right)$
f^1

Fig 10: Effect of poor resolution on Strawman-CIC

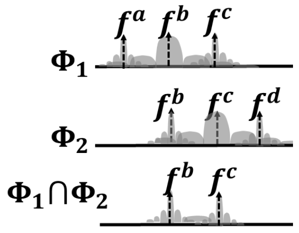


Fig 11: Illustration of Property P2

$\{C^2_{next},\cdots,C^i_{next}\}\cup\{C^{i+1}_{prev},\cdots,C^N_{prev}\}$. This is because, transmissions $2\le k\le i$, have already transitioned from C^k_{prev} to C^k_{next} prior to $t=\tau_i$ and thus all of f^k_{prev} are absent and all f^k_{next} are present. On the other hand, transmissions $i<k\le N$ have not yet transitioned to transmitting the symbol C^k_{next}, thus for all these transmissions, all f^k_{next} are absent and f^k_{prev} are present. Thus, the spectrum of their de-chirped versions $\Phi(r_{i\to i+1}(t))$ (Eqn 4), comprises a set of exactly the N frequencies $F(r_{i\to i+1})=\{f^1\}\cup\{f^2_{next},\cdots,f^i_{next}\}\cup\{f^{i+1}_{prev},\cdots,f^N_{prev}\}$ (Figure 7). *f^1, the frequency of the symbol to be decoded will be present in all sub-symbols since it is present throughout $r(t)$.*

Interference Cancelling Sub-Symbol Set (ICSS). Consider the set of sub-symbols $\{r_{1\to2}(t),r_{N\to N+1}(t)\}$. $F(r_{1\to2}(t))$ comprises of the frequencies $\{f^1\}\cup\{f^1_{prev},\cdots,f^N_{prev}\}$ as none of the symbols have transitioned to their next symbol. $F(r_{N\to N+1}(t))$ comprises of the frequencies $\{f^1\}\cup\{f^1_{next},\cdots,f^N_{next}\}$ as all of the symbols have transitioned to their next symbol. Thus, no interfering symbol frequencies are common across this set of frequencies. Consequently, $\{r_{1\to2}(t),r_{N\to N+1}(t)\}$ is an example of ICSS.

A Strawman-CIC. To provide an intuition into how CIC works, we consider a Strawman-CIC that uses the ICSS $\{r_{1\to2}(t),r_{N\to N+1}(t)\}$. The only constituent frequency common to these two sections is f^1. Thus, f^1 can be extracted by estimating all the frequencies in $F(r_{1\to2}(t))$ and $F(r_{N\to N+1}(t))$ and finding $F(r_{1\to2}(t))\cap F(r_{N\to N+1}(t))$ (Figure 9). As we discuss later in this section, Strawman-CIC's performance is adversely affected by loss of spectral resolution as dictated by Hiesenberg's Time-Frequency uncertainty principle.

5.1 Time-Frequency Uncertainty

Heisenberg's time-frequency uncertainty principle states that estimating frequencies over signals with short time-spans result in poor frequency resolution. Specifically, estimating the frequency spectrum using a sub-symbol with a smaller time-span $\frac{T_s}{K}$, will reduce the spectral resolution to $\frac{B}{K}$ (B is the bandwidth of the signal and K is a constant), making it difficult to distinguish between two symbols whose frequencies are less than $\frac{B}{K}$ apart.

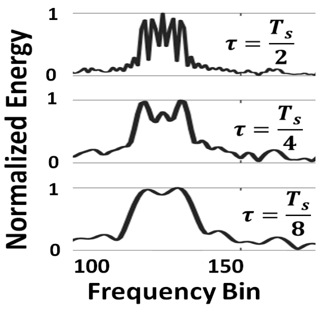


Fig 15: Heisenberg's Principle

This is illustrated in Figure 15, which shows the spectrum estimated for a signal with 5 interfering symbols using progressively smaller sized time-spans τ. As seen in Figure 15, when τ is large ($T_s/2$), all the five peaks corresponding to the five symbols are distinct. However, as τ decreases, the frequency resolution of the spectrum decreases and the peaks merge into a single peak at $\tau=\frac{T_s}{8}$. *This loss of resolution adversely effects CIC, as the peaks in spectral estimates merge into one another when using sub-symbols with a short time-span.*

5.2 Spectral Intersection

CIC attempts to find the common frequency across all sub-symbols in an ICSS, each estimated with a different frequency resolution since their time-spans may be different. The process of extracting frequencies in spectra is achieved by detecting peaks in the spectra; it requires careful choice of thresholds to reject false peaks arising out of noise, without missing the right ones, and can be error prone. Finding constituent frequency peaks separately for each of the spectra in an ICSS and then set intersection results in errors incurred at each step to accumulate, leading to poor cancellation.

Rather than extract peaks for each of the spectra separately, CIC first computes a Spectral Intersection by computing *the minimum energy across all the spectra at each frequency.* Spectral intersection is illustrated for Strawman-CIC in Figure 11, where, after taking the minimum energy in the spectra of Φ_1 and Φ_2 at each frequency, only peaks that are located at common frequencies f^b and f^c remain. Now peak detection needs to be performed only once to extract the common frequencies, making the process less error prone. Note that, prior to computing the intersection, all estimated spectra must be normalized to have unit energy to eliminate scaling effects due

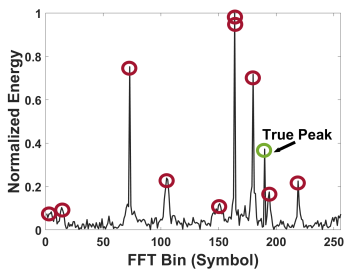


Fig 12: Collision

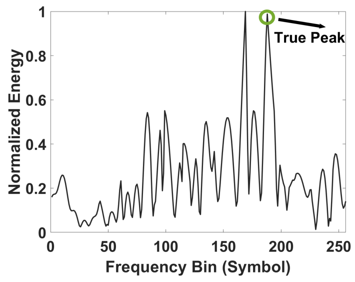


Fig 13: Effect of loss of resolution in Strawman CIC

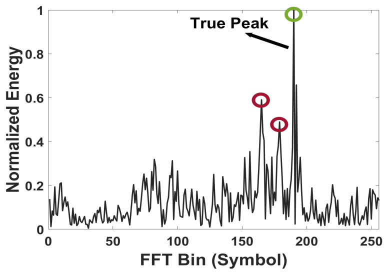


Fig 14: Interference Cancellation in CIC

to different sized windows. In this paper, we shall use $\Phi_1 \cap \Phi_2$ to denote the spectral intersection operation using minimum.

Properties of Spectral Intersection. Spectral intersection has some key properties that CIC exploits in its design.

- P1 : it is commutative and associative, as inherited from the Minimum operation.
- P2 : when two spectra have different frequency resolutions, the operation preserves the higher resolution at each constituent frequency.

The illustration in Figure 11 provides an intuition into property P2. In Figure 11, Φ_1 has a lower resolution estimate on frequency f^b but a higher resolution estimate for f^c while the vice-versa is true for Φ_2. Computing the spectral intersection takes the minimum at each frequency and hence preserves the best resolution for both frequencies.

5.3 Effect of Poor Frequency Resolution on Strawman-CIC

In order to motivate our design of CIC we start by showing how time-frequency uncertainty adversely effects Strawman-CIC. The time-spans of $r_{1\to 2}(t)$ and $r_{N\to N+1}(t)$ are τ_1 and $T_s - \tau_N$ respectively. Assuming that the symbol start times of all colliding transmissions are uniformly distributed in the interval $(0, T_s)$, the expected values of τ_1 and $T_s - \tau_N$ are both $\frac{T_s}{N}$ [3]. Consequently, the corresponding spectral estimates have a frequency resolution of $\frac{B}{N}$, making it hard to separate two constituent frequencies that are within this resolution. Figure 10 illustrates the effect of this lower frequency resolution estimates on CIC, where energy from each frequency spills over in its neighborhood. Instead of a clean sharp peak as illustrated in Figure 9, the resulting spectrum comprises multiple wide interfering peaks in Figure 10, rendering cancellation ineffective.

5.4 Design of CIC

CIC aims to pick an ICSS that best preserves the frequency resolution for each of the constituent frequencies. This choice not only preserves the maximum resolution for f^1, making it easy to extract its peak, but also aids in sharply canceling the other interfering peaks.

[3]The expectation of the minimum of N uniform random variables

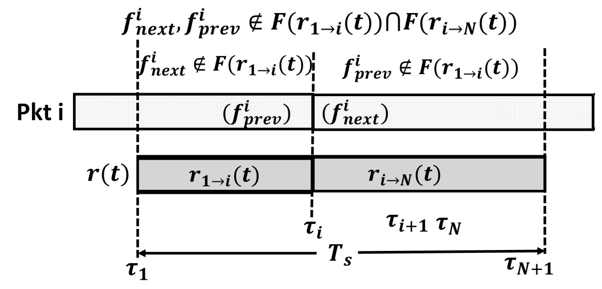

$f^i_{next}, f^i_{prev} \notin F(r_{1\to i}(t)) \cap F(r_{i\to N}(t))$
$f^i_{next} \notin F(r_{1\to i}(t))$
$f^i_{prev} \notin F(r_{1\to i}(t))$
Pkt i
(f^i_{prev})
(f^i_{next})
$r(t)$
$r_{1\to i}(t)$
$r_{i\to N}(t)$
τ_i
τ_{i+1} τ_N
τ_1
T_s
τ_{N+1}

Fig 16: Canceling a single interferer in CIC

How to cancel a specific interfering transmission at the best possible resolution. An interfering frequency f^i_{prev} is present throughout the section $r_{1\to i}(t)$ - a time-span of τ_i. Thus, the maximum achievable frequency resolution for f^i_{prev} can be obtained in the spectrum estimate $\Phi(r_{1\to i}(t))$ is given by $B\frac{\tau_i}{T_s}$. Similarly, the best frequency resolution for f^i_{next} is achieved from the spectrum estimate $\Phi(r_{i\to N+1}(t))$ is given by $B\frac{T_s - \tau_i}{T_s}$. As depicted in Figure 16, $F(r_{1\to i}(t))$ does not have f^i_{next} since the transmission only switches to f^i_{next} for $t > \tau_i$. Similarly, $F(r_{i\to N+1}(t))$ does not have f^i_{prev} since the the transmission has already switched to f^i_{next} at $t = \tau_i$. Thus, $F(r_{1\to i}(t)) \cap F(r_{i\to N+1}(t))$ will not have the frequencies f^i_{prev} and f^i_{next}. Further, $\Phi(r_{1\to i}(t))$ and $\Phi(r_{i\to N}(t))$ will also have the highest possible frequency resolutions for f^i_{prev} and f^i_{next}, following property P2 of spectral intersection, $\Phi(r_{1\to i}(t)) \cap \Phi(r_{i\to N+1}(t))$ *will remove* f^i_{prev} *and* f^i_{next} *at their respective maximum possible frequency resolutions.*

The Optimal Choice for ICSS. CIC constructs ICSS with $2N - 1$ sub-symbols, comprising all pairs $r_{1\to i}(t), r_{i\to N+1}(t)$ for $2 \le i \le N$ and finally $r(t)$ itself. Each pair, $r_{1\to i}(t), r_{i\to N+1}(t)$ cancels the i^{th} transmission at their corresponding highest possible frequency resolutions. Following properties P1 and P2 of spectral intersection, computing a spectral intersection over the ICSS will cancel all frequencies at their highest possible frequency resolutions. Finally, the inclusion of $r(t)$ ensures that f^1 is recovered at the highest possible resolution. Thus, CIC computes the spectral intersection

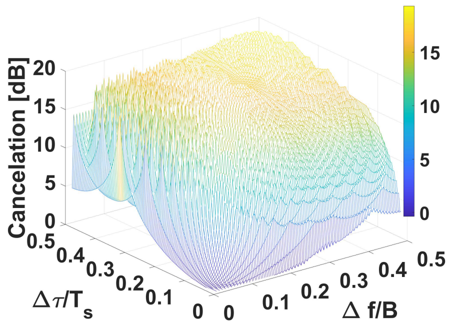


Fig 17: Cancellation in CIC

as,

$$\Phi_{CIC}(r(t)) = \left[\bigcap_{i=1}^{i=N} (\Phi(r_{1\to i}(t)) \cap \Phi(r_{i\to N+1}(t))\right] \cap \Phi(r(t)) \quad (12)$$

To provide an intuition into how CIC removes interfering symbols, Figures 12, 13,and 14 depicts demodulation of 6 colliding transmissions (at SF=8) using COTS LoRa devices using Standard LoRa, Strawman-CIC, and CIC respectively. Red circles indicate interfering transmissions while the green circle indicates the true peak. In Figure 12, four interfering peaks have higher energy than the true peak corresponding to the symbol being decoded. This occurs because, the received powers of the interfering transmissions can be stronger than the one being decoded. While Strawman-CIC does eliminate some of interfering symbols, it suffers due to lack of frequency resolution (Figure 13). However, CIC is able to extract the correct frequency while preserving the highest frequency resolution.

5.5 Extent of CIC Cancellation

In this section we try and answer the question, "How much cancellation can CIC offer and what factors does it depend on?" The conjunction of two events make cancellation hard. First, an interfering symbol overlaps to a large extent with the symbol of interest ($Min(\tau_i, T_s - \tau_i)$ is small). Second, the symbols' chirp start frequencies are very close. Thus, the combination of both close time proximity $\Delta\tau$ and frequency proximity Δf of an interfering symbol adversely effects cancellation. For a symbol with start chirp frequency $f1$ We define $\Delta\tau$ and Δf, from a symbol of frequency f as,

$$\Delta\tau \quad = \quad \begin{array}{ccc} \tau_i & for & C^i_{next} \\ T_s - \tau_i & for & C^i_{prev} \end{array} \quad (13)$$

$$\Delta f \quad = \quad \begin{array}{ccc} |f^1 - f^i_{prev}| & for & C^i_{prev} \\ |f^1 - f^i_{next}| & for & C^i_{next} \end{array} \quad (14)$$

The extent of cancellation for CIC can be analytically computed, however, we avoid presenting it due to lack of space. Figure 17 depicts the extent to which CIC can cancel a particular symbol as a function of $\Delta\tau$ and Δf for $SF = 8$. As seen from Figure 17, while the cancellation can be as high as 20dB when $\frac{\Delta\tau}{T_s} = \frac{\Delta f}{B} = 0.5$, there is almost no cancellation when both these values are close to 0. The cancellation increases to 5dB by the time these values reach 0.1. This shows that even after CIC, one or more peaks may only be partially cancelled and in case of a strong interfering transmission, they may be hard to discard.

5.6 Spectral Edge Difference (SED)

CIC may only be able to cancel certain symbols partially when $\Delta\tau$ and Δf are small. This can be seen in Figure 14, where while the peak corresponding to f^1 is the significant highest peak, some of the interfering peaks with high transmit powers (indicated by red circles) remain. In such a case, CIC has more than one potential candidate and CIC must pick one among them. For this, CIC computes the Spectral Edge Difference (SED), the absolute difference in energies of the candidate frequency spectra between left and right halves of $r(t)$, for each candidate frequency. It then picks the frequency with the least value of SED. The key intuition behind computing SED is that, SED will be zero only for f^1 since unlike interfering symbols, this frequency exists uniformly across the entire symbol.

We illustrate the intuition into SED in Figure 18 for the symbol C^i_{next}. Let E_i be the total energy per symbol in the i^{th} packet (based on the received signal strength). $r_{lh}(t)$ and $r_{rh}(t)$ represent the left and right halves of $r(t)$. Since the total duration that C^i_{next} is present in $r_{lh(t)}$ is $\frac{T_s}{2} - \tau_i$, the energy of the peak corresponding to f^i_{next} denoted by $\lambda_{lh}(f^i_{next}) = E_i\left(\frac{\frac{T_s}{2}-\tau_i}{T_s}\right) = \frac{1}{2}E_i\left(1 - \frac{\tau_i}{T_s}\right)$. f^i_{next} is continuously present in the entire right half $r_{rh}(t)$, consequently, the energy of the peak in this half, $\lambda_{rh}(f^i_{next}) = \frac{E_i}{2}$. The SED $\Lambda(f^i_{next})$ for this frequency is given by $|\lambda_{rh}(f^i_{next}) - \lambda_{lh}(f^i_{next})| = \frac{1}{2}E_i\frac{\tau_i}{T_s}$. Since, f^1 is present in both halves completely, $\lambda_{rh}(f^1) = \lambda_{lh}(f^1) = \frac{1}{2}E^1$ and $\Lambda(f^1) = 0$. SED exploits this difference and picks the candidate with the least $\Lambda(f)$.

To make the SED estimate robust, instead of relying only on a single pair (left and right), in practice, CIC uses multiple sliding windows of span ($\frac{T_s}{2}$) over the signal (10 in our implementation) from the left and right ends of the symbol and computes their spectral intersection.

$$\Lambda(f) \quad = \quad |\lambda_{rh}(f) - \lambda_{lh}(f)| \quad (15)$$

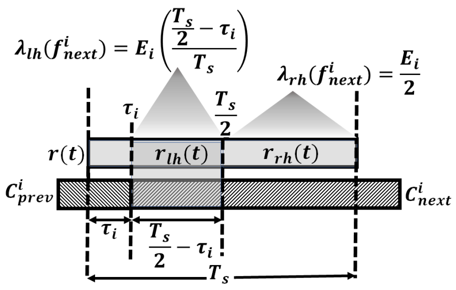
$$\lambda_{lh}(f) \quad = \quad \bigcap_{i=1}^{i=n}\Phi\left(r(t).W\left(\frac{2(t-i\epsilon)}{T_s}\right)\right) \quad (16)$$

$$\lambda_{rh}(f) \quad = \quad \bigcap_{i=1}^{i=n}\Phi\left(r(t).W\left(\frac{2(t+i\epsilon-\frac{T_s}{2})}{T_s}\right)\right) \quad (17)$$

In Eqn 16 and 17, $W(t)$ is the rectangular window function as defined in Eqn 7.

5.7 Using Additional Features in CIC

Prior work has exploited the uniqueness of features such as CFO [14] and received power [16] for each packet to group symbols. These features can also be used to pick out the appropriate symbol in the event of multiple candidates. We estimate CFO and RSSI from the preambles (as described in Section 5.8) and use these as additional features to filter out partially cancelled interfering symbols. In our implementation, we use the technique in Choir [14] to estimate the fractional CFO for each of the candidate symbols and eliminate



$\lambda_{lh}(f^i_{next}) = E_i\left(\frac{\frac{T_s}{2} - \tau_i}{T_s}\right)$
$\lambda_{rh}(f^i_{next}) = \frac{E_i}{2}$
τ_i
$\frac{T_s}{2}$
$r(t)$
$r_{lh}(t)$
$r_{rh}(t)$
C^i_{prev}
C^i_{next}
τ_i
$\frac{T_s}{2} - \tau_i$
T_s

Fig 18: SED Illustration

symbols with fractional-CFO-error more than $\frac{B}{4*SF}$. We use a size 16× FFT instead of 256× FFT since we find that is more computationally efficient without sacrificing on performance. Similarly, we also filter symbols whose received power deviates by more than 3dB from the estimated value in the preamble. In our evaluations, we examine the improvements due to each of these features compared to basic CIC.

5.8 Down-Chirp Based Preamble Detection

As described in Section 3, preamble detection in LoRa exploits its repetitive structure and searches for a sequence of 8 consecutive C_0 symbols. A key challenge in employing this scheme in the event of collisions, arises from the fact that all data symbols, as well as SYNC symbols in LoRa are merely frequency shifted versions of C_0. Consequently, data symbols from ongoing concurrent transmissions interfere with preambles, creating a clutter of peaks resulting in preamble detection errors. To provide an insight into this clutter, Figure 19 depicts the clutter of peaks during preamble detection of a new packet due to ongoing concurrent transmissions.

In our implementation, we take a different approach of searching for the two down-chirps in the preamble instead. The key insight that drives this choice, is that down-chirps do not correlate with C_0 and consequently do not correlate with data symbols in ongoing concurrent transmissions. Thus, to detect preambles we correlate with the down-chirp C_0^* instead and look for two consecutive peaks. Figure 20 depicts the peaks as detected by using down-chirps. Comparing Figures 19 and 20, using down-chirps significantly clears the clutter of peaks. Having found the two down-chirps, to confirm the preamble, we detect a preceding sequence of 8 C_0s and two SYNC words by employing an up-chirp as with standard preamble detection. This approach besides improving preamble detection compared to existing methods, also reduces the computational complexity of considering all peaks arising out of data symbols from ongoing interfering transmissions.

Estimation of CFO and Received Power. As discussed in Section 5.7, performance of CIC improves by using CFO and received power to filter partially cancelled symbols. In order to enable this, for each detected preamble, we estimate CFO (as in Choir [14]) by averaging over all the preamble up-chirp symbols for a robust estimate and maintain a list of CFOs for all ongoing transmissions. Similarly, we also estimate the FFT peak height for each preamble up-chirp symbol and average across them for a robust estimate. These peak heights are also maintained in a similar manner as CFOs.

6 IMPLEMENTATION

CIC is implemented at the LoRa gateways and does not require any changes to the COTS LoRa sensor devices. We envision CIC as either being co-located with a Software Defined Radio-based gateway at the edge or as a virtual gateway in the cloud in case of a C-RAN [18] architecture. In general, a LoRa receiver comprises three separable parts – a radio front end, a demodulator, and a decoder as depicted in Figure 21. The radio front end receives radio waves and converts them into raw digital baseband samples. The demodulator is responsible for preamble detection and converting the raw samples into LoRa symbols. Finally, the decoder maps the obtained LoRa symbols to bits based on the LoRa standard specification for deinterleaving, Forward Error Correction(FEC), and Cyclic Redundancy Check(CRC).

The Radio Front End. We used USRP B200 [19] as our radio front end at 2MHz bandwidth. Typically the received signal is oversampled i.e the sampling rate employed is significantly higher than the required Nyquist rate (4-10×) to allow better averaging. Since our COTS devices were configured at 250 KHz bandwidth, we have an oversampling of 8×. The received samples are then input to the demodulator.

Need for Distinct Demodulator and Decoder Modules. CIC replaces the standard LoRa demodulator for preamble detection and converting samples to symbols. Since standard LoRa demodulators expect to receive only one packet at a time, the demodulator and decoder are usually integrated into a single implementation. Unlike standard LoRa, CIC however, can process concurrent transmissions and thus, generate multiple streams of symbols, one for each packet simultaneously. This means that multiple decoders might be needed to concurrently process the output of a single sample stream. Consequently, we provide separate implementations for a demodulator and a decoder.

Demodulator. We have implemented CIC demodulator in three different environments – Matlab, GNU Radio [20] and Python. Matlab is often the first choice for a large number of communication researchers as it allows quick trials, modifications, simulations, and experimentation to gain experience. For experimental deployments and trials, GNU Radio is a popular choice, as it allows for quick configuration of the receiver through a GUI. We have implemented CIC demodulator as a GNU Radio block. Finally, our python implementation is useful for practical deployments in the cloud as a C-RAN module or at the gateway edge. We also provide data sets collected in our experiments for testing and verification.

Decoder. Since the decoder needs to be LoRa compliant, we modified rppo/gr-lora [21], a popular, open-source GNURadio block for LoRa reception. Since demodulation and decoding are integrated in rpp0, we extracted the decoder C++ code and created a separate GNU Radio module for the decoder that takes symbols as input, and outputs bits. This allows researchers to mix and match decoders with different demodulators and decode multiple packets concurrently.

LoRa Devices. We used the commercially available LoRa transmitters – Adafruit Feather M0 with RFM 95 [22]. These devices allow

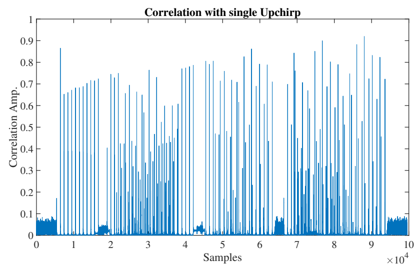


Fig 19: Upchirp based preamble detection

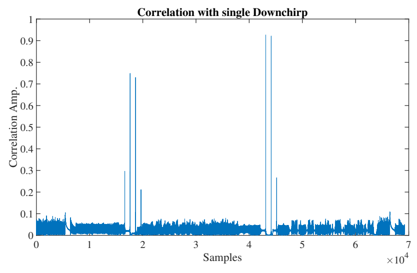


Fig 20: DownChirp based preamble detection

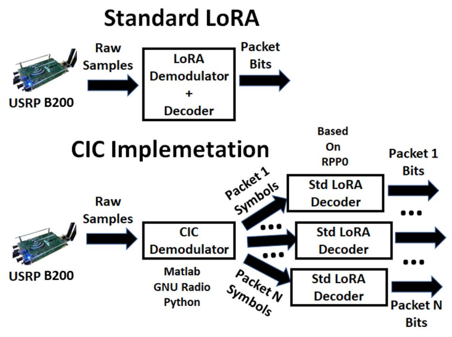


Fig 21: Our Implementation

us to configure various transmission parameters such as Spreading Factor (SF), Bandwidth (BW), Coding Rate using Arduino Library RadioHead [23].

7 EVALUATION

In this section we evaluate and demonstrate the efficacy of CIC. We answer the following questions.

- How does CIC improve network capacity compared to standard LoRa as well as state of the art techniques?
- How does CIC perform in various deployments including indoor/outdoor, low/high noise, and Los/NLoS scenarios?
- How does employing down-chirp based preamble detection improve packet detection over conventional preamble detection techniques?
- How do various additional discriminating features such as CFO, and Received Power improve CIC performance?
- How does temporal proximity of packet collisions effect CIC's ability to cancel inteference?

7.1 Deployments and Experimental Setup

To test CIC under varying conditions such as high/low SNR, indoor/outdoor, LoS/NLoS, we evaluated CIC in four different test deployments as described below. Each deployment comprised 20 LoRa devices (depicted as circles) and a gateway (depicted as a triangle).

D1: Small Indoor Space - High SNR, LoS All state of the art techniques perform their best in High SNR and LoS scenarios. To provide the best benefit to state of the art, we deployed 20 LoRa nodes within a large laboratory (Fig. 22,Fig. 26). The received SNR from the devices was approximately 30-40dB as seen in Fig. 27 and they were all in Line of Sight.

D2: Small Floor Space - High SNR, NLoS Next we evaluate CIC across the floor of an indoor space (Fig. 23). The nodes were stuck 6 feet high on the walls, some inside rooms and others outside. Here the received SNR was also between 30-40dB (Fig. 27), however many of the devices did not enjoy a direct line of sight to the gateway.

D3: Large Floor Space - Low SNR, NLoS While many state of the art techniques flounder in low SNR scenarios, CIC continues to perform well. In this deployment we chose a large floor space (Fig. 24). There was significant variation in received SNRs across the various devices ranging between 5dB to 30dB (Fig. 27) and most devices had no line of sight. This deployment is representative of a realistic large scale indoor deployment.

Fig 26: Nodes in Indoor Deployments

D4: Outdoor Wide Area Deployment - Sub-Noise, NLoS In order to test the performance of CIC in a practical wide area outdoor scenario, motivated by the smart street lighting application, we deployed our LoRa devices on street lights over an area of 2 Sq. Km in an urban environment, as depicted in Fig. 25. Most of our packet receptions in this deployment were below the noise floor and signal strength fluctuations were common as pedestrians and traffic passed by (Fig. 27). Consequently, CIC was tested to its utmost in this deployment.

Traffic Generation and Experimental Methodology. IoT traffic is generated in response to unpredictable random physical events e.g. cars arriving at a parking lot often modeled as Poisson [24, 25] arrivals. Consequently, in our experiments, devices were configured to generate packets with exponentially distributed intervals. Each sensor node generates an exponentially distributed random variable ΔT ($pdf(\Delta T) = \mu e^{-\mu \Delta T}$) to determine the time interval for transmitting the next packet. In order to generate Poisson traffic in the network with an aggregate rate of R packets/second, we choose $\lambda = \frac{R}{20}$ since we had 20 nodes in each of our deployments. To record the actual number of packets transmitted, we recorded the transmissions at each node. Finally, the number of correctly received packets (based on all bits being correct) measures the network throughput.

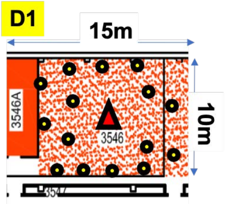


Fig 22: Indoor LOS

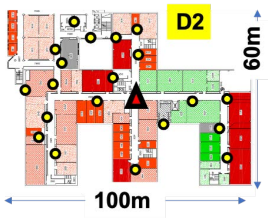


Fig 23: Indoor NLOS(small)

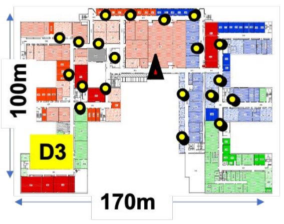


Fig 24: Indoor NLOS(larger)

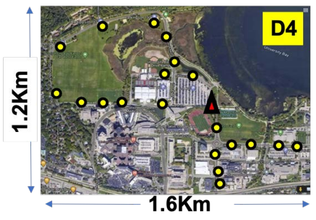


Fig 25: Outdoor

Each device is configured to transmit packet with 28 bytes of a randomly generated data packet at SF=8, BW=250KHz, 4/5 coding rate, lasting a duration of 45ms. These choices are anecdotally, the most popular in LoRa deployments. Thus, a single device could transmit a maximum of 22 packets each second back-to-back. We increased the aggregate rate R from 5 Pkts/sec to 100 Pkts/sec by changing λ from 0.25 Pkts/sec to 5 Pkts/sec to measure network capacity in each experiment. A USRP B200 was used to collect received samples at 2 MHz sampling rate providing us 8× oversampling. In each deployment, packets were transmitted at each rate for a duration of 1 minute. Thus, at the highest rate of 100 Pks/sec a total of 6000 packets were transmitted, while at the lowest rate of 5 packets/sec, 300 were transmitted.

Comparison with State of the art. We compared the performance of CIC with standard LoRa as a baseline, and two popular state of the art in research – Choir [14] and FTrack [12]. Choir is probably the first significant effort towards multi-packet collision decoding in LoRa. To the best of our knowledge, FTrack has the best performance of all existing literature. We thank the authors of FTrack for providing us their implementation and supporting us. We implemented Choir based on the description in the paper.

7.2 Network Throughput

Figs. 28, 29, 30, 31 depict the number of successfully received packets per second as the aggregate network traffic increases from 5 Pkts/sec to 100 Pks/sec. Note that since each packet lasts 45ms in our deployment allowing for a maximum of 22 Pkts/sec, if a single node were transmitting back-to-back, the maximum rate of 100 Pkts/sec is 5× greater than what any single node could transmit.

D1 : High SNR, LoS. This scenario gives the best benefit of doubt to all schemes and establishes the limits of their performance. As seen from Fig. 28, CIC significantly outperforms FTrack (by 4×), standard LoRa (by 5×) and Choir. CIC is able to decode 45 Pkts/sec, 2× greater than 22 Pkts/sec (the maximum possible for any single node) when the aggregate network traffic is 100 Pkts/sec. Standard LoRa is able to achieve a highest throughput of about 8 Pkts/sec, roughly one third of 22 Pkts/sec (the maximum). FTrack outperforms Standard LoRa, achieving about 12 Pkts/sec when the aggregate rate is 50 Pkts/sec (25% of the offered load). However, at aggregate network rates greater than 50 Pkts/sec, its performance degrades; this is due to the increase in collisions, which in turn leads to higher chance of collisions where majority of the packets overlap. In such scenarios, FTRack fails to distinguish between the corresponding frequency tracks due to its poor frequency resolution.

D2,D3 : High/Low SNR, NLoS. In this scenario as well, CIC significantly outperforms FTrack, LoRa as well as Choir by 4×. CIC is able to receive about 40 Pkts/sec, slightly less than the high SNR, LoS scenario. Standard LoRa's performance is consistent and almost the same as that of high SNR scenario as it successfully captures the higher SNR packets in case of a collision. As the authors of FTrack themselves claim, Ftrack fails to detect packets with low SNR, especially in the presence of stronger transmitters. Consequently, FTrack's performance degrades in the low SNR scenario.

D4 : Wide Area Deployment - SubNoise, NLoS. This deployment stress tests every scheme the most as the received SNRs are below noise levels. CIC's performance really shines in this regime providing almost 10× the throughput of standard LoRa. As expected FTrack is unable to decode at these SNRs and completely fails. Choir and Standard LoRa suffer heavy packet losses due to low SNR as well as collisions. Curiously, the net throughput of LoRa increases slightly at higher aggregate rates. This is because, the gateway successfully captures more packets with higher signal strengths and most of these packets are from a small subset of transmitters whose aggregate rate also increases proportionally.

Conclusion. As seen from the above experiments, CIC outperforms FTrack as well as other schemes significantly (4×), especially in wide area deployments where received signals strengths can be below noise levels, where it achieves about 10× gains. This superior performance of CIC stems from its interference cancellation mechanism guided by the Heisenberg's Time-Frequency uncertainty principle.

7.3 Preamble Detection Accuracy

Packet detection using preambles is the first and most crucial step to decoding a packet. As discussed in Section 5.8, CIC modifies the commonly used preamble detection using up-chirps to using down-chirps. In this section we ask the question "How does CIC's packet detection perform under packet collision scenarios and compare against the conventional approaches?" Figs. 32, 33, 34, 35, show the packet detection rate, the ratio of the number of packets detection (not necessarily correctly decoded) to the total number of packets transmitted, for each of the deployments and aggregate transmit rates. We compare the detection performance of CIC to that of FTrack and Standard LoRa. We are unable to compare the preamble detection of Choir since the authors do not describe their preamble detection in their paper; in the implementation of Choir, we therefore assume standard LoRa-based packet detection for Choir.

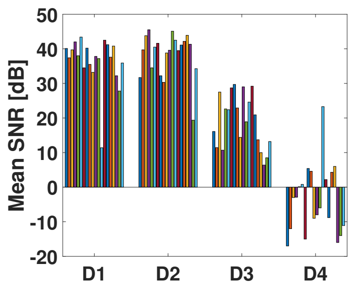


Fig 27: SNR Distribution for each of the deployments

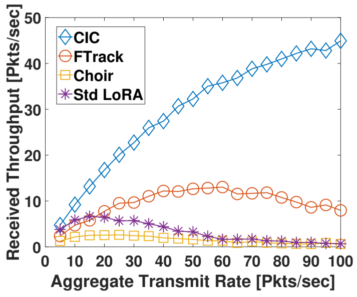


Fig 28: Network Capacity for D1 (High SNR, LoS)

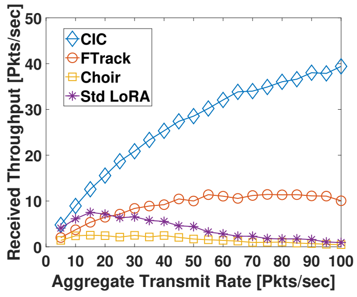


Fig 29: Network Capacity for D2 (High SNR, NLoS)

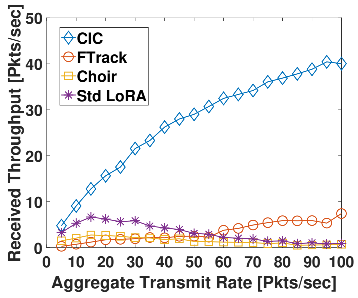


Fig 30: Network Capacity for D3 (Low SNR, NLoS)

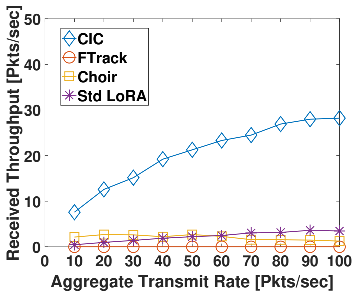


Fig 31: Network Capacity for D4 (Outdoor, SubNoise SNR, NLoS)

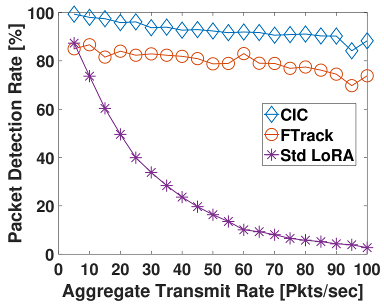


Fig 32: Packet Detection : D1 (High SNR, LoS)

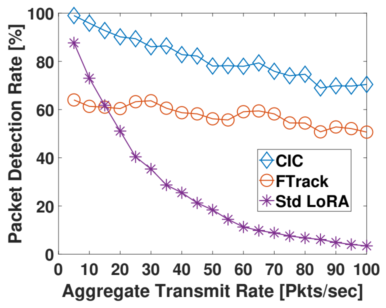


Fig 33: Packet Detection : D2 (High SNR, NLoS)

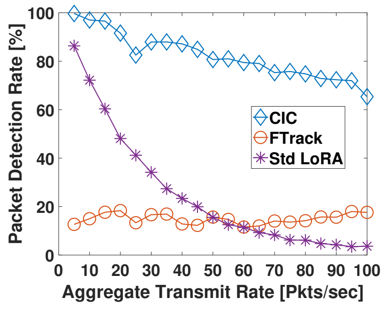


Fig 34: Packet Detection : D3 (Low SNR, NLoS)

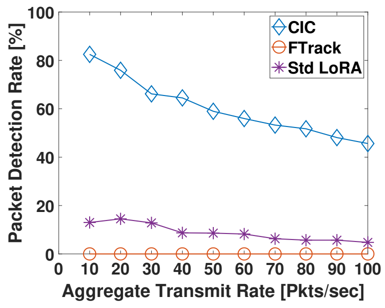


Fig 35: Packet Detection : D4 (Outdoor, SubNoise SNR, NLoS)

D1: High SNR, LoS. As seen in Fig. 32, CIC outperforms FTrack by a margin of about 20% steadily as the aggregate network traffic (hence packet collision rate) increases. Standard LoRa's packet detection quickly suffers and degrades with aggregate packets transmitted in the network.

D2: High SNR, NLoS. As SNR decreases, the packet detection rates of both CIC and FTrack suffer, however, CIC still performs better with a margin of over 20%.

D3: Low SNR, NLoS. As SNR decreases further, the packet detection rates of FTrack completely flounders and in fact falls below that of standard LoRa for high aggregate network traffic scenarios. CIC however, offers close to 80% detection even at very high aggregate network traffic scenarios.

D4: SubNoise SNR, NLoS. In this deployment, FTrack is simply unable to detect packets while standard LoRa has a detection rate of about 5%. While CIC's preamble detection performance decreases, it still offers up to 80% in low traffic and 50% in very high traffic scenarios.

Conclusions. Based on this analysis, using down-chirps as the first step for packet detection significantly improves packet detection rate under collisions and especially in low and sub-noise SNR scenarios that are common to both indoor and outdoor LoRa deployments in practice.

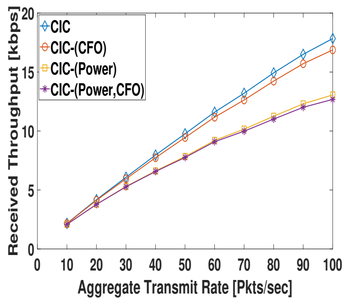


Fig 36: Effect of Removing Various Features from CIC for D1

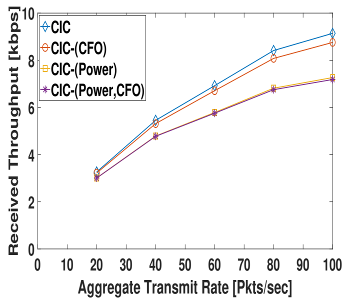


Fig 37: Effect of Removing Various Features from CIC for D4

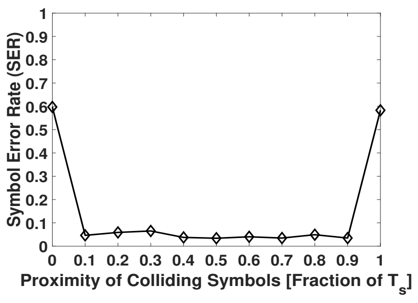


Fig 38: Simulation study of CIC as two packets collide closer in time.

7.4 Effect of the additional features of CIC

As discussed in Section 2, existing works relies on grouping symbols from the same transmitter by exploiting various discriminating features that are distinct to transmitters such as received power and CFO. As discussed in Section 5.7, CIC makes use of two additional features – Received Power and Carrier Frequency Offset (CFO), to filter out candidate frequencies, when CIC is unable to cancel them sufficiently. In this section we ask the question, *How much do these additional features contribute to the overall performance of CIC?* To this end, we use four different versions of CIC.

- *CIC* : Implementation of CIC with both Received Power and CFO included as discriminating features.
- *CIC-(CFO)* :Implementation of CIC with only Received Power i.e. without CFO as a discriminating feature.
- *CIC-(Power)* : Implementation of CIC with only CFO i.e without Received Power as additional feature.
- *CIC-(Power, CFO)* : Implementation of CIC without either Received Power or CFO as discriminating features.

Figs. 36 and 37 depicts the aggregate network throughput obtained for each of these options in deployments D1 and D4 respectively. D1 (High SNR, LoS) represents the easiest scenario for CIC and D4 (Outdoor, SubNoise SNR, NLoS) the hardest. Thus, these two represent the two extreme cases.

D1. As seen in Fig. 36, CIC gains by about 20% using both Received Power and CFO as discriminating features. However, most of these gains are due to the Received Power feature rather than CFO. Using CFO helps CIC marginally, about 2%, whereas using Received Power helps CIC by almost 18%.

D4. As seen from Fig. 37, even though the net achieved throughput is lower, the relative gains (in %) due to each of these features is almost the same.

Conclusion: Received power used as a feature helps CIC the most and provides close to 18% gains. While CFO also assists CIC, it does so rather modestly by about 1-2%.

7.5 Effect of Temporally Close Collisions

As discussed in Section 5.5, CIC can effectively cancel transmissions whose symbol boundaries are far apart. In this section, we try and understand how CIC is affected by the proximity of the interfering symbol boundaries. Since synchronizing two COTS LoRa devices to transmit within sample level accuracies is hard, we rely on simulations. In our simulations, we generate packets with random bits and generate raw signals as a LoRa transmitter would. Then we superimpose two such packets with varying sub-symbol time offsets (< 1ms). We increase time offsets in increments of 10% of the symbol ($\approx 100\mu$s). Then we use our implementation of CIC to recover all the symbols in each packet and measure the symbol error rate. In order to avoid effects due to noise, the signals are generated at 30dB SNR. Fig. 38 depicts the dependence of symbol error rate (SER) as a function of inter-symbol separation $\Delta\tau$ as a fraction of the symbol time T_s. As seen in the figure, CIC is able to cancel efficiently for $\frac{\Delta\tau}{T_s} > 0.1$. As the two colliding packets are closer than 10% of the symbol time, CIC starts to experience high SER.

8 CONCLUSION

In this work, we proposed Concurrent Interference Cancellation, a novel demodulation technique at the LoRa receiver to decode multiple colliding packets concurrently. CIC cancels out interfering symbols by selecting the optimal set of sub-symbols. CIC incorporates prior work on LoRa packet collisions as additional features. CIC implemented in real outdoor and indoor deployments using 20 COTA LoRa transmitters and USRP as gateway shows significant improvements in network throughput performance compared to standard LoRa receiver as well as state-of-the-art research works. We show that CIC is robust to variations in SNR across LoRa devices, making it suitable for practical LoRa deployments. This work does not raise any ethical issues.

9 ACKNOWLEDGEMENTS

Suman Banerjee was supported in part through the following US National Science Foundation(NSF) grants : CNS-1838733, CNS-1719336, CNS-1647152, CNS-1629833, and CNS-2003129 and an award from the US Department of Commerce with award number 70NANB21H043. Bhuvana Krishnaswamy and Muhammad Osama Shahid were partially supported through the following NSF grants : CCSS - 2034415.

REFERENCES

[1] LoRa. https://www.semtech.com/lora.
[2] E. Asimakopoulou and N. Bessis. Buildings and crowds: Forming smart cities for more effective disaster management. In *2011 Fifth International Conference on*

Innovative Mobile and Internet Services in Ubiquitous Computing, pages 229–234, 2011.

[3] María V Moreno, Miguel A Zamora, and Antonio F Skarmeta. User-centric smart buildings for energy sustainable smart cities. *Transactions on emerging telecommunications technologies*, 25(1):41–55, 2014.

[4] Achim Walter, Robert Finger, Robert Huber, and Nina Buchmann. Opinion: Smart farming is key to developing sustainable agriculture. *Proceedings of the National Academy of Sciences*, 114(24):6148–6150, 2017.

[5] Climate Smart Agriculture. https://www.worldbank.org/en/topic/climate-smart-agriculture.

[6] Fei Tao, Qinglin Qi, Ang Liu, and Andrew Kusiak. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, 2018.

[7] Harsha V Madhyastha and Chinedum Okwudire. Remotely controlled manufacturing: A new frontier for systems research. In *Proceedings of the 21st International Workshop on Mobile Computing Systems and Applications*, pages 62–67, 2020.

[8] Aloÿs Augustin, Jiazi Yi, Thomas Clausen, and William Mark Townsley. A study of lora: Long range & low power networks for the internet of things. *Sensors*, 16(9):1466, 2016.

[9] Branden Ghena, Joshua Adkins, Longfei Shangguan, Kyle Jamieson, Philip Levis, and Prabal Dutta. Challenge: Unlicensed lpwans are not yet the path to ubiquitous connectivity. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–12, 2019.

[10] Martin C Bor, Utz Roedig, Thiemo Voigt, and Juan M Alonso. Do lora low-power wide-area networks scale? In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 59–67, 2016.

[11] Brecht Reynders and Sofie Pollin. Chirp spread spectrum as a modulation technique for long range communication. In *2016 Symposium on Communications and Vehicular Technologies (SCVT)*, pages 1–5. IEEE, 2016.

[12] Xianjin Xia, Yuanqing Zheng, and Tao Gu. Ftrack: Parallel decoding for lora transmissions. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, pages 192–204, 2019.

[13] Shuai Tong, Jiliang Wang, and Yunhao Liu. Combating packet collisions using non-stationary signal scaling in lpwans. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 234–246, 2020.

[14] Rashad Eletreby, Diana Zhang, Swarun Kumar, and Osman Yağan. Empowering low-power wide area networks in urban settings. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '17, page 309–321. Association for Computing Machinery, 2017.

[15] Xiong Wang, Linghe Kong, Liang He, and Guihai Chen. mlora: A multi-packet reception protocol in lora networks. In *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, pages 1–11. IEEE, 2019.

[16] Shuai Tong, Zhenqiang Xu, and Jiliang Wang. Colora: Enabling multi-packet reception in lora. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 2303–2311. IEEE, 2020.

[17] Leon Cohen. *Time-frequency analysis*, volume 778. Prentice Hall PTR Englewood Cliffs, NJ, 1995.

[18] Aleksandra Checko, Henrik L Christiansen, Ying Yan, Lara Scolari, Georgios Kardaras, Michael S Berger, and Lars Dittmann. Cloud ran for mobile networks—a technology overview. *IEEE Communications surveys & tutorials*, 17(1):405–426, 2014.

[19] USRP B200. https://www.ettus.com/all-products/ub200-kit/.

[20] GNU Radio. https://www.gnuradio.org/.

[21] RPPO. https://github.com/rpp0/gr-lora.

[22] Adafruit Feather M0 with RFM95 LoRa Radio. https://www.adafruit.com/product/3178.

[23] RadioHead RFM95 Library. https://www.airspayce.com/mikem/arduino/RadioHead/classRH__RF95.html.

[24] Orestis Georgiou and Usman Raza. Low power wide area network analysis: Can lora scale? *IEEE Wireless Communications Letters*, 6(2):162–165, 2017.

[25] Jiangbin Lyu, Dan Yu, and Liqun Fu. Achieving max-min throughput in lora networks. In *2020 International Conference on Computing, Networking and Communications (ICNC)*, pages 471–476. IEEE, 2020.