De la demodulación a la decodificación: hacia un LoRa completo Comprensión e implementación de PHY

ZHENQIANG XU, SHUAI TONG, PENGJIN XIE, y JILIANG WANG, Escuela de Software,

Universidad de Tsinghua, China

LoRa, como representante de la tecnología de redes de área amplia de baja potencia (LBRA), ha atraído gran atención tanto del ámbito académico como de la industria. Sin embargo, el conocimiento actual de LoRa dista mucho de ser completo, y sus implementaciones presentan una gran brecha de rendimiento en cuanto a relación señal-ruido (SNR) y tasa de recepción de paquetes. Este artículo presenta una comprensión completa del protocolo de capa física (PHY) de LoRa y revela las razones fundamentales de esta brecha de rendimiento. Presentamos la primera implementación completa de LoRa PHY con una garantía de rendimiento demostrable. Mejoramos la demodulación para que funcione con una SNR extremadamente baja (−20 dB) y validamos analíticamente el rendimiento, mientras que muchos trabajos existentes requieren una SNR > 0. Derivamos el orden y los parámetros de las operaciones de decodificación, incluyendo el deblanqueo, la corrección de errores, el desentrelazado, etc., aprovechando las características de LoRa y la manipulación de paquetes. Implementamos un LoRa completo en tiempo real en la plataforma GNU Radio y realizamos experimentos exhaustivos. Nuestro método puede lograr (1) una tasa de éxito de decodificación del 100% mientras que los métodos existentes pueden soportar como máximo un 66,7%, (2) una sensibilidad de -142 dBm, que es la sensibilidad límite del LoRa básico, y (3) un alcance de comunicación de 3600 m en el área urbana, incluso mejor que el LoRa básico en la misma configuración.

Conceptos CCS: • Redes → Protocolos de red; Análisis del rendimiento de la red; Redes de área amplia;

Palabras y frases clave adicionales: Red de área amplia de bajo consumo, LoRa, capa física

Formato de referencia ACM:
Zhenqiang Xu, Shuai Tong, Pengjin Xie y Jiliang Wang. 2022. De la demodulación a la decodificación: Hacia una comprensión e implementación completas de la física LoRa. ACM Trans. Sensor Netw. 18, 4, artículo 64 (diciembre de 2022), 27 páginas.
https://doi.org/10.1145/3546869

1 INTRODUCCIÓN

La tecnología de red de área amplia de bajo consumo (LPWAN) ha demostrado ser muy prometedora para conectar millones de dispositivos en el Internet de las cosas (IoT) , proporcionando comunicación de larga distancia y bajo consumo con una relación señal-ruido (SNR) muy baja. LoRa, como tecnología LPWAN representativa, ha sido ampliamente adoptada recientemente tanto en el ámbito académico como en la industria [1–4]. Se ha utilizado en diversas aplicaciones del IoT, como ciudades inteligentes, agricultura inteligente y logística inteligente [5–8]. También atrae a un gran número de...

Este trabajo está financiado en parte por NSFC No. 62172250, No. 61932013, Programa de Investigación Científica de Iniciativa de la Universidad de Tsinghua.

Dirección de los autores: Z. Xu, S. Tong, P. Xie y J. Wang (autor correspondiente), Facultad de Software, Universidad de Tsinghua, Beijing, China; correos electrónicos: xu-zq17@mails.tsinghua.edu.cn, tl19@mails.tsinghua.edu.cn, xiepengjin@tsinghua.edu.cn, jiliangwang@tsinghua.edu.cn.

1550-4859/2022/12-ART64
https://doi.org/10.1145/3546869

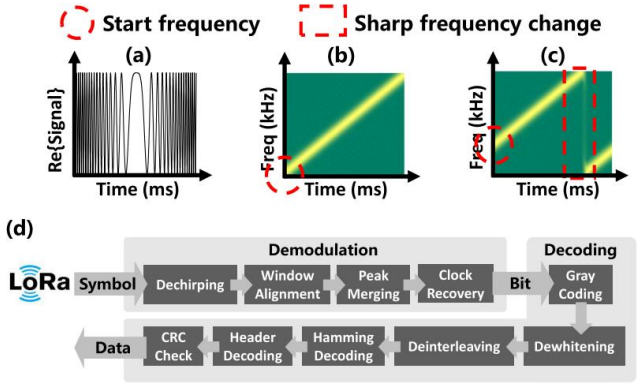

Fig. 1. (a) Parte real de un símbolo de chirrido ascendente de base. (b) Símbolo de chirrido ascendente de base. (c) Símbolo desplazado. (d) Procedimiento completo de LoRa PHY.

Varios trabajos de investigación incluyen resolución de colisiones [9, 10], retrodispersión LoRa [11-13], decodificación de señales débiles [14, 15], etc.

LoRa adopta el mecanismo de modulación Chirp Spread Spectrum (CSS) , que proporciona capacidad de comunicación antiinterferente y de largo alcance. Como se muestra en la Figura 1(a) y (b), un símbolo base en el protocolo de capa física (PHY) de LoRa es un chirrido con una frecuencia que aumenta linealmente con el tiempo. La frecuencia de inicio (fstart) de un símbolo representa la información codificada. Un símbolo de LoRa tiene dos segmentos con una caída de frecuencia pronunciada, como se muestra en la Figura 1(c). Aunque LoRa ha atraído mucha atención en el ámbito académico y la industria, los detalles de LoRa PHY, es decir, cómo LoRa demodula y decodifica la señal recibida, aún nos es desconocido, porque LoRa PHY es un protocolo cerrado propiedad de Semtech Corporation. En este trabajo, pretendemos presentar una comprensión completa de la capa física de LoRa y proporcionar una mejor herramienta para analizar la red LoRa. Revelamos que la mayoría de los conocimientos existentes son incompletos e incluso incorrectos. Como resultado, incluso la implementación de LoRa más utilizada, rpp0/gr-lora (RPP0) [16], tiene una gran brecha de rendimiento con respecto al hardware real, por ejemplo, la tasa de recepción de paquetes (PRR) de RPP0 es solo del 20,0 %, y RPP0 requiere una SNR mucho más alta que LoRa.

Hay dos pasos principales para comprender la física de LoRa: demodulación y decodificación.[1] La demodulación traduce los símbolos a bits sin procesar, mientras que la decodificación traduce estos bits sin procesar a bytes de datos significativos. El objetivo de la demodulación de LoRa es obtener la fstart de cada símbolo. La demodulación determina directamente el rendimiento del receptor. La demodulación de LoRa, actualmente predominante, primero traduce la señal a un solo tono multiplicando cada símbolo por un chirp linealmente decreciente y extrae la frecuencia de este tono como frecuencia de inicio. Esta operación puede provocar una alta pérdida de SNR debido a la pérdida de señal en la traducción y a la imposibilidad de aprovechar las características de LoRa, lo que impide la comunicación de largo alcance con baja SNR. Explicamos las razones exactas de la pérdida de señal en la Sección 3.1.

Para la decodificación, los trabajos existentes [16-21] no pueden derivar el orden ni los parámetros de las operaciones de decodificación. Por lo tanto, presentan una PRR muy baja (p. ej., inferior al 66,7 %) incluso con una relación señal-ruido (SNR) alta, debido a una comprensión incorrecta del proceso de decodificación de LoRa. Además, suelen ser incompletos y exagerados; por ejemplo, los métodos de las referencias [16-18] afirman que admiten todos los factores de propagación (FE), pero nuestros experimentos demuestran que no lo hacen.

1Para el transmisor, modulación y codificación. Aquí solo se describe el comportamiento del receptor para simplificar.

Tabla 1. Comparación de BastilleResearch/gr-lora (BR), RPP0 y tapparelj/gr-lora_sdr (TAPP) y nuestra implementación

	BR [17]	RPP0 [16]	TAPP [18]	Nuestro
Todos los SF Todos los CR CRC Modo explícito Modo implícito Recuperación de reloj Soporte de baja relación señal-ruido Tiempo real				66,7 %
Cobertura de paquetes	4,2%	20.0%		100%

Este artículo proporciona una comprensión completa y un LoRa PHY en tiempo real de pila completa con un Garantía de rendimiento demostrable. En la sección de demodulación, revelamos la razón fundamental de... La pérdida de SNR es una desalineación de fase debido a la desalineación de la ventana y al desfase interno del símbolo. Primero, compense la desalineación de la ventana midiendo con precisión el desplazamiento de la misma. Para solucionarlo Desplazamiento de fase del símbolo interno, a diferencia de los trabajos existentes con frecuencia de muestreo B (ancho de banda), Sobremuestreamos la señal con 2B para calcular los picos de cada segmento por separado. Luego, proponemos una método para fusionar esos dos picos con mínima pérdida de sensibilidad. Demostramos teóricamente que La sensibilidad de nuestro método es muy cercana a la demodulación "perfecta". Además, proponemos una Algoritmo de compensación de deriva de reloj dinámico para dispositivos LoRa de bajo costo y LoRa relativamente largos. paquetes.

En la parte de decodificación, mostramos cómo LoRa PHY traduce los bits de la parte de demodulación. a paquetes significativos. Según las observaciones de la fórmula oficial [22] para calcular el número de símbolos en un paquete LoRa, inferimos y verificamos la estructura del paquete a partir de la LoRa de caja negra mediante la manipulación del contenido del paquete. Con base en la estructura del paquete, analizamos los requisitos de cada proceso de decodificación y derivar el orden de los procesos de decodificación para Gray Codificación, desentrelazado, decodificación de Hamming y deblanqueamiento. Luego, aprovechando el orden de decodificación inferido y la estructura de paquetes, manipulamos los paquetes transmitidos para obtener la configuración. parámetros en el proceso de decodificación, la estructura del encabezado y el polinomio CRC.

Implementamos un PHY LoRa de radio definida por software (SDR) en tiempo real , como se muestra en la Figura 1(d). La Tabla 1 muestra la comparación con los últimos avances. Nuestra implementación supera a estos enfoques al (1) proporcionar decodificación en tiempo real con un requisito de relación señal/ruido muy bajo, (2) ser compatible con todos los... modos y parámetros de LoRa, y (3) trabajar de manera robusta en dispositivos de bajo costo con deriva de reloj y paquetes largos. Demostramos teóricamente que nuestra implementación podría funcionar con una relación señal-ruido (SNR) de -20 dB. El orden de las operaciones de decodificación también es demostrable, lo que se puede verificar mediante el resultado aleatorio generado. Secuencia (Sección 4.3). En nuestros experimentos, el 100 % de los paquetes se decodificaron correctamente. Por lo tanto, consideramos que la configuración es fiable.

Contribuciones.

- Mostramos la brecha de rendimiento entre las implementaciones de LoRa existentes y el producto básico LoRa y revelamos las razones fundamentales. Presentamos una comprensión integral de La demodulación y decodificación de LoRa.
- Implementamos un PHY LoRa completo en tiempo real en la plataforma GNU Radio SDR.2 Basándonos en nuestro análisis, incluso podemos analizar y mejorar el rendimiento de los productos básicos.

2https://github.com/jkadbear/LoRaPHY.

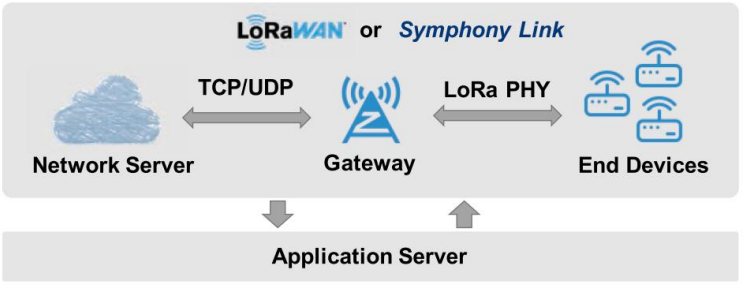

Fig. 2. Arquitectura de red LoRa. El protocolo LoRaWAN o Symphony Link controla cómo funciona un servidor de red.
interactúa con los dispositivos finales.

LoRa. Por ejemplo, demostramos que el chip LoRa estándar es vulnerable a un ataque de desalineación de fase (Sección 6.5) y mostramos cómo abordar esta vulnerabilidad.

Realizamos experimentos exhaustivos para verificar el rendimiento. Los resultados muestran que... La sensibilidad de nuestra implementación alcanza −142 dBm, que es la sensibilidad límite de LoRa estándar. Nuestra implementación tiene un alcance de comunicación efectivo de 3600 m. en el entorno urbano, mientras que las puertas de entrada del RPI de productos básicos (Sección 5) solo llegan a los Máximo de 2800 m. Nuestra implementación de LoRa tiene una tasa de éxito de decodificación mucho mayor. (100%) que las obras existentes (≤66,7%).

2 ANTECEDENTES

2.1 Datos conocidos y desconocidos de LoRa

La figura 2 muestra la arquitectura de red LoRa convencional. LoRaWAN [23] o Symphony Link [24] El protocolo controla toda la red LoRa y proporciona una interfaz de datos al servidor de aplicaciones. Hay tres entidades de comunicación principales en la red LoRa, es decir, servidor de red, puerta de enlace, y el dispositivo final. El servidor de red se comunica con la puerta de enlace mediante una conexión a Internet tradicional. Conexión mientras los dispositivos finales intercambian datos con la puerta de enlace mediante el uso inalámbrico LoRa PHY Protocolo. Casi todos los componentes y protocolos utilizados en la Figura 2 tienen implementaciones completas de código abierto, excepto LoRa PHY. Si bien existen algunos trabajos de ingeniería inversa, la comprensión de LoRa PHY aún no es completa. A continuación, se muestran algunos conceptos básicos. de LoRa PHY y discutiremos los detalles sobre la demodulación y decodificación en las Secciones 3 y 4.

LoRa PHY emplea CSS para lograr una transmisión de largo alcance. La unidad básica de comunicación en LoRa PHY es un símbolo de chirp. Como se muestra en la Figura 1(b), denotamos este símbolo de chirp sin desplazamiento como base. Cuando la frecuencia aumenta con el tiempo, lo llamamos chirrido ascendente y, en caso contrario, chirrido descendente. La novedad de la modulación LoRa es utilizar la frecuencia de inicio de un símbolo desplazado cíclicamente para codificar Datos. La Figura 1(c) muestra un símbolo LoRa con dos segmentos de chirp. En las especificaciones LoRa [22, 25, 26], El número de bits codificados por un símbolo es SF. El SF satisface $2^{SF} = \frac{B \cdot T}{}$, donde B es el ancho de banda. y T es el período de chirrido. Hay como máximo 2 símbolos de chirrido ascendente desplazados cíclicamente de F para un SF específico. Un chirrido ascendente se puede representar como

$$\text{chirp}(t; f0) = Ae^{j2\pi (f0 + \frac{B}{2T} t)t}, \quad (1)$$

donde A es la amplitud y f0 es la frecuencia de inicio (t = 0).

La Figura 3 muestra la comprensión básica de un paquete LoRa estándar. En general, consta de tres partes: Preámbulo, delimitador de trama inicial (SFD) y símbolos de datos. El preámbulo es una serie de bases

Fig. 3. Un paquete LoRa en el plano tiempo-frecuencia.

Chirridos ascendentes seguidos de dos símbolos de chirrido ascendente que indican la identificación de la red.3 El SFD es 2,25 chirridos descendentes
Indica el inicio de los símbolos de datos. Los símbolos de datos incluyen el encabezado PHY, la carga útil y el CRC de la carga útil.
(el encabezado y el CRC son opcionales).

LoRa admite las siguientes configuraciones. Tasa de código (CR): LoRa aplica el código Hamming de
La tasa de $\frac{4}{5}, \frac{4}{6}, \frac{4}{7}, \frac{4}{8}$ [22]. Optimización de baja velocidad de datos (LDRO): al enviar paquetes largos,
codificación LoRa permite el modo LDRO para una mejor estabilidad a costa de la tasa de datos. Modo implícito/explícito: En explícito
modo, hay un encabezado PHY en el paquete, mientras que en el modo implícito, no hay encabezado PHY.

Desconocido: 4 (1) La demodulación ideal de LoRa, es decir, cómo se traduce un símbolo a bits, es
Aún se desconoce cuál es la clave para que LoRa logre una relación señal-ruido (SNR) baja y una comunicación de larga distancia.
(2) Ninguno de los trabajos existentes propone una comprensión completa de la estructura de paquetes LoRa.
(3) Se desconocen los detalles de los modos específicos de LoRa, por ejemplo, el modo LDRO.

2.2 Implementaciones de código abierto de última generación

El espacio de investigación previa en comunicación LoRa es bastante rico, incluyendo la decodificación de colisiones LoRa [1, 2, 4, 9,
10, 27], la retrodispersión basada en chirridos LoRa [11-13, 28], la optimización de la red LoRa [15, 29],
Y así sucesivamente. Desafortunadamente, la mayoría no publica implementaciones de acceso público. Considerando que la arquitectura
de la red LoRa es completamente nueva y aún no está completamente comprendida por...
Comunidad, cualquier nuevo esfuerzo de código abierto merece reconocimiento. Por lo tanto, primero resumimos...
Proyectos de código abierto relacionados con LoRa disponibles públicamente.

Implementaciones de capa superior. La capa superior de la red LoRa cuenta con implementaciones de código abierto de alto
rendimiento. Para la puerta de enlace y el dispositivo final LoRa, Semtech Corporation proporciona un sistema integrado oficial.
Implementaciones [30, 31]. Para servidores de red, ChirpStack [32] es una implementación ampliamente utilizada.
Compatible con LoRaWAN. LoRaWAN-Server [33] proporciona una combinación compacta de red
Servidor y servidor de aplicaciones. El código de nivel más bajo aquí es el controlador para chips LoRa. Por lo tanto,
no incluye el mecanismo de capa física de comunicación LoRa.

Implementaciones de la capa física. Dado que LoRa PHY es un protocolo propietario que pertenece a
Corporación Semtech, no existe ningún documento público que explique todos los detalles de la capa física LoRa.
Las investigaciones y los sistemas se basan principalmente en tres implementaciones de SDR de código abierto, es decir, BR [17], RPP0
[16] y TAPP [18]. BR se centra en el modo de encabezado implícito del paquete LoRa. BR aplica demodulación basada en descifrado.
Ignora algunos detalles cruciales sobre el descifrado y no es fiable en baja...
SNR. Discutiremos los detalles en la Sección 3. Además, la implementación de BR solo admite
decodificación de paquetes LoRa con SF=8. Nuestras pruebas muestran que BR solo puede decodificar los primeros cuatro bytes de
un paquete y no puede decodificar ningún paquete completo con más de cuatro bytes. RPP0 apunta a explícito
Decodificación de paquetes LoRa en modo encabezado. Como se muestra en la Figura 1(c), RPP0 demodula los símbolos LoRa.
Al encontrar cambios bruscos de frecuencia y luego utilizarlos para estimar las frecuencias de inicio de la

3Los dos símbolos no suelen ser chirridos ascendentes de base. Se utilizan como máscaras de red para separar diferentes redes. Para
Por ejemplo, se establecen en 0x0304 para la red pública en LoRaWAN.
Aunque los trabajos existentes han revelado muchos detalles sobre LoRa PHY, el rendimiento de demodulación no ideal y la tasa de éxito de
decodificación no del 100% significan que sus resultados son incompletos.

símbolos. RPP0 depende de información del dominio del tiempo y no puede decodificar paquetes LoRa para SNR<0.
TAPP es una implementación reciente de LoRa. TAPP no está optimizado para la demodulación de LoRa, lo que genera una alta sobrecarga computacional. La PRR de TAPP disminuye cuando el decodificador no puede consumir los datos de entrada a tiempo. Según nuestra evaluación, su pérdida de paquetes alcanza el 70 % para una velocidad de datos de un paquete por segundo. La PRR promedio de TAPP es de tan solo el 66,7 %. Para la decodificación, ninguno de los trabajos existentes puede proporcionar una capacidad de decodificación completa, por lo que solo pueden decodificar una parte de los paquetes LoRa. Además, existen otros trabajos relacionados con la física de LoRa que ofrecen implementaciones de código abierto. Charm [14] se centra en la decodificación de señales débiles aprovechando la estructura multipuerta de la red LoRa. Sin embargo, el decodificador Charm [34] solo se implementa en modo sin conexión con MATLAB. Además, el autor ignora el procedimiento de decodificación y el problema de la alineación de fase (véase la Sección 3). TinySDR [35] es una plataforma SDR diseñada para redes del Internet de las Cosas (IoT).

El autor implementa un modulador LoRa [36] en FPGA, lo que nos muestra un método conveniente de generación de chirp por hardware. Sin embargo, el proceso de demodulación suele ser mucho más complejo que la modulación, y actualmente no existe un demodulador LoRa de código abierto basado en FPGA. Además, el proceso de decodificación tampoco se considera en TinySDR. Observamos que los trabajos existentes no han proporcionado una implementación completamente utilizable ni una comprensión completa de la física de LoRa.

Limitaciones fundamentales de los trabajos existentes. (1) Los trabajos existentes no funcionan con baja relación señal-ruido (SNR) y, por lo tanto, carecen de la ventaja crucial de LoRa. (2) Los trabajos existentes no pueden proporcionar una capacidad de decodificación completa para LoRa, y su rendimiento es significativamente inferior al del hardware LoRa estándar. No son totalmente compatibles con LoRa, es decir, con el modo LDRO, todos los SF, etc.

Nuestro objetivo. Nuestro objetivo es proporcionar una implementación física completa de LoRa, incluyendo demodulación y decodificación, para permitir la recepción de señales con una relación señal-ruido extremadamente baja y todos los parámetros/modos de LoRa.

3 DEMODULACIÓN

En esta sección, revelamos el proceso de demodulación de LoRa y presentamos cómo romper las limitaciones fundamentales de las implementaciones de LoRa existentes.

3.1 Dechirp alineado en fase El demodulador

LoRa convierte los símbolos chirp en flujos de bits dechirpando primero la señal recibida. El proceso de dechirp consiste principalmente en dos pasos: Primero, multiplica cada chirp recibido con un chirp descendente base, y luego realiza la transformada de Fourier en los resultados de la multiplicación, traduciendo chirps a picos de energía en el dominio de la frecuencia. Idealmente, la multiplicación dará como resultado señales de un solo tono con fase continua, cuya energía puede acumularse en un solo pico, como se muestra en la Figura 4(a3) (denotado como IDEAL). Sin embargo, en la práctica, hay un desalineamiento de fase inevitable cuando la frecuencia de chirp cae de su máximo al mínimo. Este desalineamiento de fase es inducido principalmente por las inestabilidades del hardware de los dispositivos LoRa económicos y se distribuye aleatoriamente entre 0 y 2π. Con la desalineación de fase, los picos de energía de la transformada de Fourier se distorsionan severamente, como se muestra en la Figura 4(a4), lo que limita el rendimiento de demodulación de LoRa, especialmente con baja relación señal-ruido (SNR).

Los trabajos existentes no resuelven el problema de desalineación de fase en el deschirping. Por lo tanto, su rendimiento se deteriora drásticamente en escenarios de baja relación señal-ruido (SNR), lejos de la sensibilidad ideal de la demodulación LoRa. Nuestro objetivo es aproximarnos a la sensibilidad ideal de la demodulación LoRa en condiciones de desalineación de fase. El diseño de nuestro demodulador LoRa consta principalmente de los siguientes componentes.

Alineación de la ventana. Necesitamos alinear con precisión la ventana de demodulación (nivel de puntos de muestra) con cada símbolo para conservar la energía de todo el símbolo durante el proceso de demodulación.
Los trabajos tradicionales utilizan la correlación de preámbulos para la alineación de ventanas. Sin embargo, el simple uso de la correlación de preámbulos no permite lograr una alineación precisa debido al desplazamiento de frecuencia portadora (CFO). Aprovechamos la parte SFD del paquete LoRa para la alineación de ventanas, eliminando al mismo tiempo el impacto de...

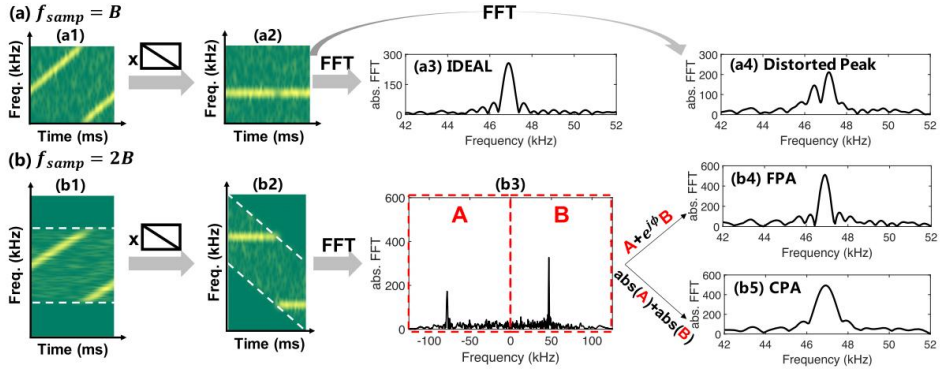

Fig. 4. El paso principal del deschirpeo consiste en multiplicar un chirp descendente y luego aplicar la FFT. (a) Deschirpeo normal con pérdida de SNR. En una situación ideal, se obtiene un pico como (a3). Sin embargo, en una situación no ideal, el pico de frecuencia se distorsionaría como (a4), lo que genera resultados de demodulación erróneos. (b) Nuestro deschirpeo con alineación de fase propuesto. Utilizamos el filtro paso bajo y el sobremuestreo para separar los dos segmentos de chirp en un símbolo LoRa y luego combinarlos. (b4) y (b5) proporcionan los resultados del deschirpeo con alineación de fase fina y gruesa, respectivamente, que son estables en cualquier situación.

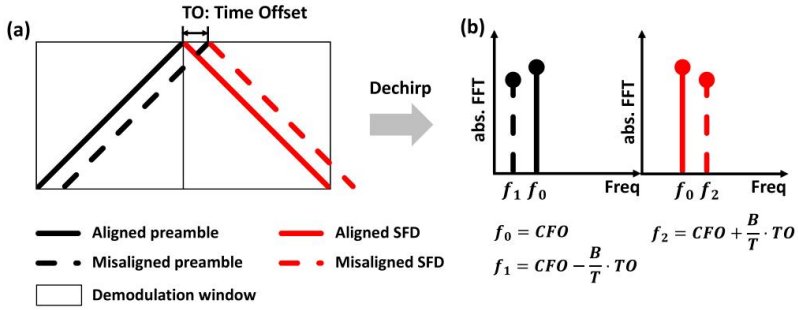

Fig. 5. Alineación de la ventana utilizando chirridos de base ascendentes y descendentes.

CFO. La operación clave consiste en combinar los chirps descendentes en SFD con los ascendentes en el preámbulo. Aplicamos deschirp a ambos (el chirp descendente se multiplica por un chirp ascendente base). Si la ventana de demodulación está perfectamente alineada con la señal, el pico del chirp ascendente y el pico del chirp descendente aparecen en la misma frecuencia, independientemente del CFO, como se muestra en la Figura 5. De lo contrario, existe una diferencia significativa en la frecuencia de los dos picos, dependiendo del desfase temporal entre el chirp y la ventana de demodulación. Por lo tanto, podemos lograr una alineación precisa basada en la frecuencia de los picos correspondientes al preámbulo y los SFD.

Sobremuestreo y fusión de picos. Anteriormente, hemos revelado que la causa fundamental de la pérdida de relación señal-ruido (SNR) en la demodulación es la distorsión de pico causada por el desalineamiento de fase cuando la frecuencia de chirp cae de su máximo a su mínimo. Para abordar este problema, proponemos un enfoque basado en sobremuestreo que recupera la distorsión de pico en presencia de desalineamiento de fase y se acerca a la sensibilidad ideal de la demodulación LoRa.

Como se ilustra en la Sección 2, LoRa modula las señales desplazando cíclicamente la frecuencia de un chirp ascendente de base. Cada chirp modulado consta de dos segmentos de chirp, uno con la frecuencia inicial de f0 y el otro con f0 − B. Los trabajos existentes demodulan los símbolos de chirp recibidos utilizando una frecuencia de muestreo igual al ancho de banda de chirp B. Por lo tanto, tras multiplicar por la base

En el chirp descendente, ambos segmentos de chirp se traducen a señales de un solo tono de la misma frecuencia debido al aliasing de frecuencia. Sin embargo, existe una inevitable desalineación de fase entre estos dos segmentos de chirp. Al aplicar la transformada de Fourier a todo el símbolo, la energía de estos dos segmentos de chirp se suma de forma destructiva, lo que genera distorsión de pico y pérdida de relación señal/ruido (SNR) para la demodulación LoRa. Los estudios existentes desconocen el valor de la desalineación de fase. Por lo tanto, no pueden distinguir la señal de estos dos segmentos de chirp ni corregir la distorsión de pico.

Proponemos una estrategia basada en sobremuestreo para separar eficientemente los dos segmentos de chirp en el dominio de la frecuencia. Específicamente, dado que las frecuencias iniciales de los dos segmentos de chirp son f0 y f0−B, al sobremuestrear la señal con una frecuencia superior a 2B (es decir, Fs ≥ 2B), el deschirp genera dos picos distintos, ubicados por separado en f0 y Fs − B + f0, como se muestra en la Figura 4(b3).
Dado que no hay desalineación de fase en cada segmento de chirp, ambos picos están libres de distorsión. Esto nos permite concentrar perfectamente la energía de todo el chirp sumando coherentemente estos dos picos de los segmentos, eliminando así la influencia de la desalineación de fase. En el resto de esta sección, ilustramos cómo fusionar estos dos picos de energía en el dominio de la frecuencia de forma eficiente con una pequeña pérdida de sensibilidad para la demodulación LoRa.

Debido a la existencia de desalineación de fase, la fase de los dos picos en la Figura 4(b3) está fuera de coherencia. Por lo tanto, la suma directa de estos dos picos complejos no aumentará la altura del pico o incluso se cancelará entre sí debido a la diferencia de fase de estos dos picos. Proponemos la Alineación de Fase de Grano Fino (FPA) para buscar todos los valores posibles de desalineación de fase $\Delta\varphi = \frac{i\times 2\pi}{k}$ ($i = 0, 1, \ldots, k-1$) y compensarlos antes de sumar dos picos. Iteramos k posibles desfases y seleccionamos el $\Delta\varphi$ que genera el pico más alto. El resultado de FPA se muestra en la Figura 4(b4). En comparación con la Figura 4(a3), FPA puede aproximarse al rendimiento de IDEAL ya que la diferencia de fase se compensa.

El proceso de búsqueda para determinar el valor del desalineamiento de fase presenta una alta complejidad computacional. Para reducir esta sobrecarga, proponemos además la Alineación de Fase de Grano Grueso (CPA), que requiere mucha menos sobrecarga computacional y, por lo tanto, puede funcionar en hardware de bajo costo con recursos computacionales muy limitados. La observación clave es que la amplitud del pico es proporcional a la energía del segmento chirp correspondiente. Dado que la energía de todo el símbolo es exactamente la suma de los dos segmentos chirp, si sumamos directamente el valor absoluto (amplitud) de la parte A y la parte B, entonces el pico resultante tendrá la misma altura que el pico IDEAL. El resultado de CPA se muestra en la Figura 4(b5). Vemos que CPA cambia significativamente la forma del pico, es decir, aumenta el ancho del lóbulo principal. Además, CPA también eleva el nivel de ruido durante el proceso de suma, degradando así ligeramente el rendimiento de demodulación en comparación con FPA.

Dependiendo de la potencia de cálculo del receptor, podemos alternar entre los métodos FPA y CPA. A continuación, analizamos teóricamente la tasa de error de símbolo (SER) con respecto a diferentes SNR para IDEAL, FPA y CPA.

Suponemos que la señal LoRa se transmite a través de un canal de ruido gaussiano blanco aditivo (AWGN) , donde el ruido se modela siguiendo la distribución gaussiana compleja.
Durante el proceso de descifrado, tanto los símbolos de chirrido como el ruido se transforman en picos de energía en el dominio de la frecuencia. Se produce un error de símbolo cuando cualquiera de los picos de ruido supera el pico del símbolo objetivo. Supongamos que la altura del pico del símbolo objetivo es hd y el pico máximo correspondiente al ruido es hn; podemos obtener la tasa de error de símbolo esperada como

$$SER = P(hd < hn). \tag{2}$$

La relación señal-ruido (SER) es teóricamente una función de la relación señal-ruido (SNR). Los detalles del cálculo de hd y hn para IDEAL, FPA y CPA se detallan en la Sección 8. Graficamos el rendimiento teórico de IDEAL, FPA y CPA bajo SF8/12 en

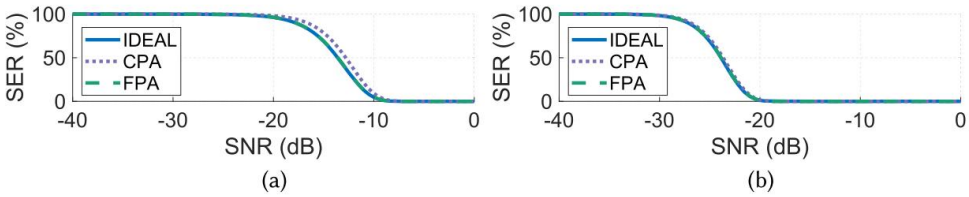

Fig. 6. Curva SER-SNR teórica para (a) SF8 y (b) SF12.

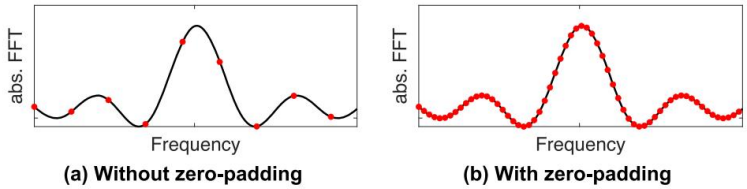

Fig. 7. Efectos de relleno de ceros. Los puntos rojos representan los resultados de la DFT tras la descodificación. Las líneas negras representan el resultado de la DTFT tras la descodificación.

[5] La longitud del paso de búsqueda en FPA (es decir, 1/k) se establece empíricamente en 1/16 (una compensación entre el rendimiento (Figura 6) y la sobrecarga computacional). Observamos que tanto FPA como CPA tienen un rendimiento muy cercano al IDEAL para SF12, mientras que la SER teórica de CPA en SF8 es ligeramente mayor que la de los otros dos métodos. En resumen, el análisis teórico indica que FPA y CPA son dos métodos prácticos y eficaces para la demodulación LoRa, lo cual se valida aún más en los resultados experimentales de la Sección 6. Nuestro análisis teórico también muestra que LoRa podría funcionar con una relación señal-ruido (SNR) extremadamente baja (es decir, de hasta -20 dB), como se puede observar en la Figura 6(b).

Refinamiento de picos. Anteriormente, ilustramos cómo convertir los chirps recibidos en picos en el dominio de la frecuencia. A continuación, necesitamos estimar la frecuencia pico más alta para recuperar los bits de datos modulados. Sin embargo, en la práctica, la estimación de la altura de los picos está estrechamente relacionada con la resolución de frecuencia después de la transformada de Fourier. La línea negra en la Figura 7(a) muestra el resultado de la Transformada de Fourier de Tiempo Discreto (DTFT) ideal en la señal objetivo, que es una función continua de la frecuencia. En la práctica, el receptor solo realiza la Transformada de Fourier Discreta (DFT), una versión muestreada de la DTFT en el dominio de la frecuencia, en la señal objetivo. La salida de la DFT está marcada con puntos rojos en la Figura 7. La DFT ingenua tiene una resolución de baja frecuencia. Por lo tanto, sus salidas se distribuyen dispersamente sobre la curva DTFT ideal, lo que disminuye la altura de pico derivada y también afecta la estimación de pico. Aplicamos relleno de ceros a la señal en el dominio temporal para obtener una estimación de pico precisa, equivalente a la interpolación en el dominio frecuencial. Como se muestra en la Figura 7(b), con el relleno de ceros, la resolución de frecuencia de la DFT mejora significativamente, lo que beneficia nuestra estimación de picos. En la Sección 6.3, mostramos que el relleno de ceros cuádruple logra un equilibrio entre la sobrecarga computacional y la sensibilidad de demodulación. El beneficio de un mayor relleno de ceros es marginal.

3.2 Recuperación de Reloj.

Dado que LoRa se aplica comúnmente en dispositivos de bajo costo, los osciladores equipados pueden no tener alta precisión, lo que resulta en valores de bin inexactos. La Tabla 2 muestra los primeros ocho bins de pico descifrados de dos paquetes de un dispositivo LoRa con un CFO de aproximadamente 16 kHz. Observamos que la parte fraccionaria

Los parámetros de 5 bits, como CR y CRC, son irrelevantes para la relación señal-ruido (SNR) a nivel de símbolo. Cabe destacar que el ancho de banda no aparece en la fórmula de la relación señal-ruido (SNR) anterior, ya que la relación señal-ruido (SNR) es una función del ancho de banda (si aumentamos el ancho de banda sin aumentar la potencia de transmisión, la relación señal-ruido (SNR) en banda disminuirá).

Tabla 2. Deriva del bin de pico bajo SF10 y SF12 (con ancho de banda de 125 kHz)

		First 8 Peak Bins							
SF10 (LDRO off)	Expected	677.0	333.0	861.0	93.0	853.0	149.0	789.0	597.0
	Received	677.**2**	333.**2**	861.**3**	93.**3**	853.**4**	149.**4**	789.**4**	597.**5**
SF12 (LDRO on)	Expected	1757.0 = 011011 011**101**	3453.0 = 110101 111**101**	673.0 = 001010 100**001**	3757.0 = 111010 101**101**	2409.0 = 100101 101**001**	809.0 = 001100 101**001**	2981.0 = 101110 100**101**	2545.0 = 100111 110**001**
	Received	1757.8 = 011011 011**101** **+0.8**	3453.9 = 110101 111**101** **+0.9**	674.0 = 001010 100**010** **+0.0**	3758.2 = 111010 101**110** **+0.2**	2410.3 = 100101 101**010** **+0.3**	810.4 = 001100 101**010** **+0.4**	2982.5 = 101110 100**110** **+0.5**	2546.6 = 100111 110**010** **+0.6**

Esperado: los bins ideales sin error de reloj; recibido: los bins recibidos reales.

de los intervalos de pico sigue desplazándose con el tiempo. Si no cancelamos esa deriva, entonces los intervalos fraccionarios... Se acumularán errores en el intervalo de enteros. Estas desviaciones del intervalo se deben principalmente al muestreo. Desplazamiento de frecuencia (SFO). Debido a la diferencia en la frecuencia de muestreo, el símbolo LoRa real es τ más corto (o más largo) que el período de símbolo estándar T. Por lo tanto, la frecuencia de inicio del siguiente El símbolo tiene una deriva adicional como

$$\Delta f = \frac{B}{T} \cdot \tau . \tag{3}$$

Debido a que la frecuencia portadora y la frecuencia de muestreo son generadas por el mismo oscilador, tener

$$\frac{\tau}{T} = \frac{OFS}{fsamp} = \frac{\Delta fosc}{fosc} = \frac{\text{director de Finanzas}}{fRF}, \tag{4}$$

Donde fRF es la frecuencia portadora de referencia, fosc es la frecuencia del oscilador de referencia y Δfosc es la desviación de la frecuencia del oscilador. Tras la conversión a valores de bin, la deriva estimada de bin entre dos valores consecutivos... símbolos is6

$$\Delta bin = \frac{\Delta f}{B/2S\,F} = \frac{\text{director de Finanzas}}{fRF} \cdot 2S\,F . \tag{5}$$

Por ejemplo, cuando CFO = 16 kHz, fRF = 470 MHz y SF = 10, la deriva de bin estimada Δbin ≈ 0,035, que coincide con la tendencia de la Tabla 2. Restamos el Δbin acumulado de los bins de pico Antes de decodificar.

Otro aspecto a considerar aquí es el modo LDRO. Normalmente, cuando un período de chirrido es... demasiado largo (es decir, T > 16 ms), LDRO se habilita automáticamente en los chips LoRa [22]. En modo LDRO, como Como se ve en la parte inferior de la Tabla 2, los valores bin esperados siempre tienen la forma de 4n+1, lo que significa Los dos bits menos significativos (LSB) no codifican datos. Este diseño busca proteger mejor Los datos, ya que los bits inferiores son más vulnerables en la transmisión de paquetes largos. Para señales LoRa que duran... En poco tiempo, los dos LSB son lo suficientemente estables para codificar datos (por ejemplo, SF10 y ancho de banda de 125 kHz). Sin embargo, en el experimento, encontramos que los primeros ocho símbolos están siempre en modo LDRO. Proteger la información del encabezado (Sección 4).

6Sin relleno de ceros, un contenedor entero en LoRa siempre representa una frecuencia de puntos o $\frac{B}{2S\,F}$ independientemente del número de FFT una frecuencia de muestreo.

4 DECODIFICACIÓN

4.1 Descripción general

Esta sección sobre decodificación no es una revisión de trabajos previos. Nuestro objetivo es proporcionar una inferencia demostrable sobre la estructura de los paquetes LoRa, el orden de decodificación y las configuraciones. De la versión oficial de LoRa. En las hojas de datos [22, 25, 37] y las patentes [26], conocemos la ecuación (6) para calcular el número de símbolos en un paquete. En cambio, se desconoce la estructura del paquete. También sabemos que hay cuatro Los principales procesos de decodificación son: codificación Gray (G), desentrelazado (I), decodificación Hamming (H) y desblanqueamiento (W). En contraste, el orden de los procesos y los parámetros de configuración de cada uno... Se desconocen los procesos. Para mostrar cómo LoRa PHY traduce los bits del módulo de demodulación Para paquetes significativos, analizamos la decodificación LoRa en tres pasos. Primero, inferimos el paquete. Estructura de la caja negra LoRa que aprovecha la fórmula del número de símbolos. A continuación, revelamos la Orden de los cuatro procesos principales de decodificación. Finalmente, inferimos los parámetros de configuración de cada uno. proceso de decodificación.

4.2 Inferencia de la estructura del paquete

El documento oficial [22] proporciona una fórmula para calcular el número de símbolos en un paquete:

$$\text{nsym} = 8 + \max\left(\frac{2\text{PL} + 4\text{CRC} - 5\text{IH} - \text{SF} + 7}{\text{SF-2DE}} \cdot \frac{4}{\text{CR}}, 0\right), \tag{6}$$

donde PL es el número de bytes de carga útil, SF es el factor de expansión, CRC es 1 si la verificación de CRC está habilitada y de lo contrario 0, IH es 1 si el modo de encabezado implícito (sin encabezado) está habilitado y 0 si el modo explícito (con encabezado) está habilitado y DE es 1 si LDRO está habilitado y, de lo contrario, 0.

Podemos derivar las siguientes propiedades de la ecuación (6):

- Un paquete LoRa tiene al menos ocho símbolos de datos como nsym ≥ 8.
- 2PL + 4CRC: El nsym de un paquete con bytes de carga útil PL y verificación CRC habilitada es igual a El nsym de un paquete con PL + 2 bytes de carga útil y la comprobación CRC deshabilitada. Podemos inferir que La comprobación CRC ocupa 2 bytes en el paquete.
- 2PL + 4CRC − 5IH: De manera similar a CRC, podemos inferir que el encabezado ocupa 2,5 bytes.
- SF − 2DE: SF es el número de bits en un símbolo de datos. Habilitar LDRO provoca la reducción de dos bits por símbolo de datos.
- $\frac{4}{\text{CR}}$:El número de símbolos de datos aumenta con la unidad de $\frac{4}{\text{CR}}$ ($\text{CR}=4_5, 4_6, 4_7,$ o 4_8). Por ejemplo, $\text{CR} = 7_n$ $\frac{4}{7}$ significa que el paquete aplica el código Hamming (7,4) y nsym tiene la forma $\frac{4}{\text{CR}}n + 8 = +8 (n = 0, 1, 2, \ldots)$.

Con base en las propiedades, manipulamos los paquetes de datos para inferir su estructura. Primero aumente gradualmente el tamaño del paquete y observe un aumento escalonado del número de símbolos de datos. En señales reales. Si CR = $\frac{4}{7}$, entonces el número de símbolos nsym aumentaría siguiendo una ecuación aritmética. secuencia (es decir, 8, 15, 22,...). También observamos que al aumentar continuamente el tamaño del paquete, Los bytes recién agregados se codifican en los últimos $(n \frac{4}{\text{CR}})$ símbolos, y el primero $\frac{4}{\text{CR}}$ Cambio. Revela que los bytes de datos están codificados por un $\frac{4}{\text{CR}}$ bloque de símbolos.

Además, encontramos que los primeros ocho símbolos reciben un tratamiento especial. Ya sea de forma explícita o implícita. En el modo de encabezado, los valores de los primeros ocho símbolos siempre aparecen en la forma 4k + 1 (k = 0, 1, 2,...), lo que significa que se descartan los dos últimos bits de datos y hay SF − 2 bits de datos en cada símbolo. Podemos inferir que los primeros ocho símbolos están en modo LDRO, es decir, DE = 1. LDRO da mejor protección, y el encabezado puede estar en los primeros ocho símbolos. Sabemos que LoRa utiliza la tasa de codificación CR para proteger la carga útil, y los bytes de la carga útil se codifican mediante un bloque de $\frac{4}{\text{CR}}$ símbolos. De forma similar, Los primeros ocho símbolos forman un bloque de ocho símbolos y suponemos que utiliza una tasa de codificación para cifrar $\frac{4}{8}$ El encabezado tiene la máxima protección (4 bits de paridad para un nibble). También observamos que los primeros ocho

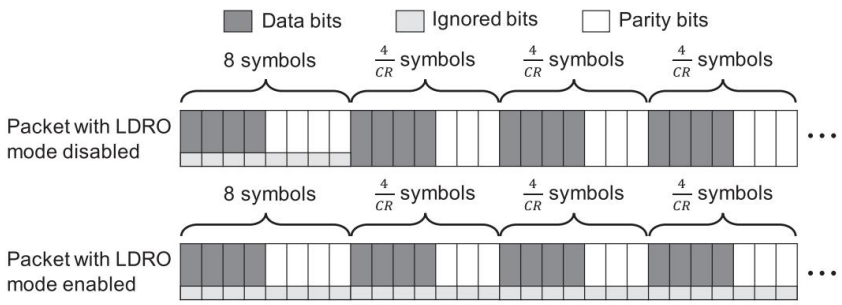

Fig. 8. Estructura de paquetes LoRa inferida para el modo LDRO deshabilitado/habilitado. Cuando el modo LDRO está deshabilitado, Solo los primeros ocho símbolos tienen bits ignorados (protegiendo la información del encabezado). Cuando el modo LDRO está habilitado, Los dos bits menos significativos se ignoran para todos los símbolos.

Los símbolos codifican bits de carga útil en modo de encabezado implícito. Este diseño puede reducir la complejidad. de la complejidad del hardware de decodificación, porque ahora se puede usar el modo de encabezado explícito e implícito. procesado con el mismo procedimiento.

La Figura 8 muestra la estructura de paquete inferida de un paquete con el modo LDRO deshabilitado/habilitado. En En el modo LDRO, se descartan los dos últimos bits de cada símbolo. Independientemente de si el modo LDRO está... Habilitados o deshabilitados, los primeros ocho símbolos están en modo LDRO y utilizan una tasa de codificación de 8. Los siguientes Los símbolos se encuentran en bloques que contienen símbolos. Los primeros cuatro símbolos codifican bits de datos, mientras que los Los símbolos de descanso codifican bits de paridad.

Verificación de la Estructura del Paquete. Verificamos la estructura del paquete anterior comparando la Ecuación (6) y el número de símbolo calculado utilizando la estructura del paquete inferida. Para SF ≥ 7, Los primeros ocho símbolos contienen 8 · (SF − 2) · = 4SF 8 ≥ 20 bits de datos. Por lo tanto, los primeros ocho Los símbolos siempre pueden incluir toda la información del encabezado. Excepto el encabezado, los primeros ocho símbolos... Puede contener 4(SF −2) −20(1−IH) = 4SF + 20IH −28 bits de carga útil. Supongamos que hay bytes PL (8PL) bits) de carga útil en total. Si el CRC está habilitado, se necesitan 16 bits adicionales. Excepto los primeros ocho símbolos, el número total de bits de datos requeridos para los símbolos restantes es 8PL+16CRC−20IH−4SF +28. El blanqueamiento y la decodificación Gray no afectan el número total de bits ni el número total de símbolos. La operación de entrelazado solo requiere que rellenemos los bytes redundantes cuando el resto de la carga útil... Los bytes no pueden llenar un bloque, por lo que no afectan el número total de bloques de símbolos. Cada bloque tiene símbolos, y cada símbolo del bloque podría codificar bits SF-2DE. Debido a la existencia de paridad... bits, solo 4 (SF − 2DE) bits en un bloque son bits de datos reales. Entonces, el número total de símbolos en el El paquete es

$$8 + \frac{8PL + 16CRC - 20IH - 4SF + 28}{4(SF - 2DE)} \cdot \frac{4}{CR}. \quad (7)$$

Es obvio que la fórmula (7) es la misma que la ecuación (6), lo que significa que la estructura del paquete inferida es correcto

4.3 Orden de las operaciones de decodificación

En esta sección, intentamos revelar el orden de los cuatro procesos principales en la decodificación de LoRa: codificación Gray (G), desentrelazado (I), decodificación de Hamming (H) y desblanqueamiento (W). Es fácil saber que Gray La codificación debe ser el primer paso, ya que pretende resolver el problema de deriva adyacente del símbolo (ver Codificación Gray en la Sección 4.4). La decodificación de Hamming opera en un flujo de bytes, mientras que un símbolo LoRa... Contiene bits SF-2DE. Por lo tanto, se basa en el flujo de bytes transformado después del desentrelazado. Por lo tanto, El orden de “G, I, H” debe ser “G→I→H”. Por lo tanto, el desblanqueamiento tiene tres posiciones posibles, es decir, después

“G”, “I” o “H”. Las implementaciones de BR, RPP0 y TAPP asumen tres posiciones de deblanqueamiento diferentes, respectivamente.

Demostramos que la posición del desblanqueo no afecta los resultados de la decodificación. Dado que el orden del proceso de codificación es inverso al de decodificación, utilizamos esta operación en la codificación para mostrar la influencia de la posición de "W". Considerando los datos como un vector de bits D, el entrelazado es una operación que reorganiza la posición de los bits. Podemos representar el entrelazado como una matriz I, donde cada fila o columna contiene solo un "1". La codificación de Hamming, como método de codificación lineal, puede representarse como una matriz H. Tenemos H · D = [P · DD], donde P es la matriz de paridad. El blanqueamiento es W D, donde W es un vector de bits aleatorio y es una operación exclusiva (XOR). A continuación, se presentan los tres posibles órdenes de decodificación y las operaciones de codificación correspondientes:

“W→I→H”: W1 (I · [P · DD]), (8a)

“Yo→An→Alto”: Yo · (An2 [P · DD]), (8b)

“I→H→W”: I · [P · (W3 D) (W3 D)], (8c)

Donde W1, W2 y W3 son tres vectores de bits aleatorios diferentes. La fórmula (8b) puede reescribirse como

(I · W2) (I · [P · DD]). (9)

Por lo tanto, las fórmulas (8a) y (9) son equivalentes (sea W1 = I · W2). La fórmula (8c) puede reescribirse como

Yo · ([P · W3 W3] [P · DD]). (10)

La fórmula (10) es un caso especial de la fórmula (8b) (sea W2 = [P · W3 W3]). Por lo tanto, tanto “I→W→H” como “I→H→W” pueden representarse mediante “W→I→H”, lo que significa que la posición de “W” no afecta el resultado final de la decodificación. Mostraremos la posición correcta de “W” en la Sección 4.4.

Para facilitar el análisis, asumimos primero que el orden de decodificación es “G→W→I→H”. Dado que la paridad par de todos los ceros es cero y que el entrelazado en LoRa consiste simplemente en una realineación diagonal de todos los bits, las palabras de código de salida después de “H” e “I” son ceros cuando los bits de entrada son todos ceros. Aquí asumimos que LoRa adopta la comprobación de paridad par comúnmente utilizada, y los resultados finales muestran que esta suposición es correcta. “W” es el XOR de los valores de los datos y una secuencia pseudoaleatoria. Cuando el orden de decodificación es “G→W→I→H”, como x 0 = x, si establecemos todos los bits de transmisión a cero, los valores de salida después de “G” son la secuencia de desblanqueamiento. Entonces, podríamos usar la secuencia de desblanqueamiento derivada para recuperar los resultados temporales anteriores a “I” y “H” para un análisis posterior.

4.4 Configuración de cada operación. Esta sección

revela las configuraciones clave de las cuatro operaciones de decodificación LoRa: codificación Gray, desentrelazado, decodificación Hamming y deblanqueamiento. También mostramos cómo revelar la estructura del encabezado y el polinomio CRC.

Codificación Gray. La codificación Gray se utiliza ampliamente en muchos sistemas de comunicación inalámbrica y consiste en la asignación de un vector de bits a una representación binaria. Las representaciones adyacentes de la codificación Gray solo presentan una diferencia de un bit. En los sistemas de comunicación inalámbrica, es más probable que se identifique erróneamente un símbolo con su símbolo adyacente que con otro símbolo aleatorio. Por ejemplo, en la modulación LoRa, la deriva de bin adyacente es común, como se describe en la Sección 3.2. Con la codificación Gray, el error de bit causado por la identificación errónea de un símbolo adyacente se reduce a un bit por símbolo, lo cual tiene una alta probabilidad de ser corregido por el mecanismo de corrección de errores. La codificación Gray estándar se puede expresar como

v = v0 (v0 >> 1), (11) donde v0 representa el valor del bin demodulado

sin procesar, >> es la operación de desplazamiento a la derecha bit a bit y v es la salida de codificación Gray. Sin embargo, al continuar nuestro análisis basado en símbolos naturales

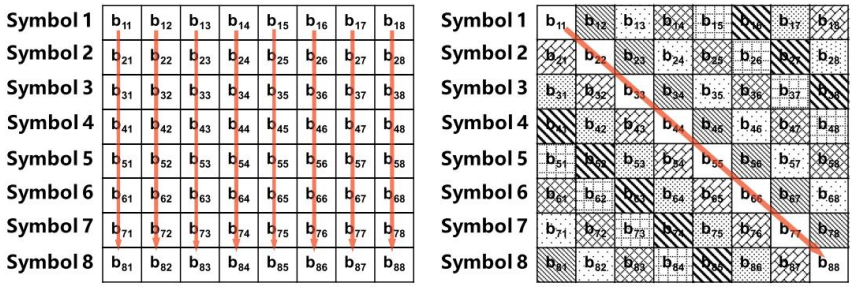

Fig. 9. Ejemplo de bloque de desentrelazado para 8 símbolos SF8 (CR = $\frac{4}{8}$). Cada fila del bloque es un bloque de 8 bits. Representación de un símbolo LoRa según la codificación Gray. bi,j representa el bit j del símbolo i. Por ejemplo, bi,1 es el LSB del símbolo i. (a) Intercalado de fila-línea de orden principal de columna, la n-ésima palabra de código se compone de bits bi,n (i = 1,..., 8). (b) Entrelazado diagonal utilizado en LoRa, la n-ésima palabra de código está compuesta de bits bi,(i+n−1)%8+1 (i = 1,..., 8).

Si tomamos como base el mapeo y la codificación Gray estándar, no podemos decodificar paquetes con una tasa de éxito de decodificación del 100 %. El proyecto RPP0 sufre este problema y aún contiene un error sin corregir.7 Para entender la razón, Recordemos el modo LDRO de LoRa. Cuando LDRO está habilitado, el codificador colocará bytes de datos en Alto SF −2 bits de un símbolo y luego añadir "1" para reducir la influencia de la deriva de bin. Desde el hardware Desde esta perspectiva, es razonable agregar “1” bajo cualquier condición, porque podemos ahorrar el costo del circuito. Determinar si LDRO está habilitado. Suponemos que todos los símbolos emitidos por el codificador tienen un único contenedor. desplazamiento. Si es cierto, antes de aplicar la codificación Gray, debemos restar "1" a la demodulación. Resultados. Afortunadamente, nuestra suposición se ve respaldada por los resultados de la decodificación. El proceso de "G" en LoRa La decodificación podría modificarse como

$$v = (v0 - 1) \quad ((v0 - 1) >> 1). \quad (12)$$

Tenga en cuenta que TAPP también aplica operaciones similares en codificación Gray como las nuestras, pero lo explican como un mecanismo de codificación Gray diferente y utiliza un algoritmo de fuerza bruta para derivar el "uno" adicional. Sin embargo, parece poco natural utilizar una codificación Gray no estándar. Por el contrario, nuestra explicación es... más razonable.

Desentrelazado. El entrelazado se utiliza para reducir el impacto de los errores de ráfaga. Con el entrelazado, Los errores podrían distribuirse a múltiples grupos de bits y corregirse mediante corrección de errores hacia adelante. (FEC). Se menciona [26] que LoRa aplica entrelazado diagonal en lugar del entrelazado fila-línea convencional. La Figura 9(a) muestra el entrelazado fila-línea de orden mayor de columna de ocho símbolos. Para el entrelazado de filas y líneas, los LSB (bi,1) de los ocho símbolos se ensamblan en un byte. Sección 3, sabemos que los LSB de un símbolo son más frágiles que los bits más significativos (MSB) del símbolo. Desde la perspectiva de FEC, agrupar bits frágiles es un mal diseño. La Figura 9(b) muestra el entrelazado diagonal utilizado en LoRa. El entrelazado diagonal distribuye la frágil LSB en diferentes bytes y es más robusto. Luego manipulamos los paquetes transmitidos para derivar El mapeo diagonal detallado. Por ejemplo, enviamos paquetes con SF = 8 y CR = implícito. $\frac{4}{8}$ Modo de encabezado. Por lo tanto, el bloque de intercalación es un bloque de 8 × 8, como se muestra en la Figura 9(b). Primero, suponemos El FEC utilizado en CR es el $\frac{4}{8}$ código Hamming estándar (7, 4) con una extensión de un bit. Por lo tanto, después de “G” y “W”, la palabra clave para el nibble “0000” es “00000000”, y la palabra clave para “1111” es “11111111.”8 Supongamos que los bytes de envío son todos ceros excepto que el cuarto byte es 0x0F, observamos

7https://github.com/rpp0/gr-lora/issues/99.

8Tenga en cuenta que aquí hemos eliminado la influencia del blanqueamiento.

b11 = b22 = ··· = b88 = 1 en la Figura 9(b). Por lo tanto, la diagonal principal representa el cuarto byte.
0x0F. El problema de desplazamiento de un contenedor mencionado en la codificación Gray refleja aquí que no siempre podemos obtener
Ocho unos en un bloque si aplicamos directamente la codificación Gray estándar. Desplazando la asignación en uno
Resuelve este problema y se adapta perfectamente a nuestro siguiente proceso de decodificación. Al cambiar el todo-1
bits de datos en el paquete transmitido, podemos derivar el mapeo completo para el entrelazado como se muestra en
Figura 9(b). Para otros parámetros, el proceso de desentrelazado es similar. La única diferencia es que
El tamaño del bloque se convierte en $\frac{4}{CR} \times (SF - 2DE)$. Como resultado, resumimos el proceso de desentrelazado como

$$c_{i,j} = b_{j,(i+j-1)\%(SF-2DE)+1}, \quad (13)$$

donde ci,j es el bit j de la palabra de código i después del desentrelazado, bj,i es el bit i del símbolo j
Tras la codificación Gray, y $i \in \{1, 2, ..., \frac{4}{CR}\}$, $j \in \{1, 2, ..., SF - 2DE\}$.

decodificación de Hamming. Tras el desentrelazado, obtenemos los datos codificados en forma de palabras de código.
pero la posición de los bits de datos y los bits de paridad en una palabra de código aún se desconoce. LoRa proporciona cuatro
valores CR válidos ($\frac{4}{5}, \frac{4}{6}, \frac{4}{7}, \frac{4}{8}$), que determinan la longitud de la palabra de código y, por lo tanto, la intensidad de FEC.
Cuando CR está configurado en $\frac{4}{7}$ o $\frac{4}{8}$, LoRa aplica el código Hamming(7, 4) o el código Hamming extendido con
Un bit de paridad adicional. Cuando CR se establece en $\frac{4}{5}$ o $\frac{4}{6}$, LoRa puede detectar errores de bits pero no puede corregirlos
ellos. Primero, asumimos que un nibble con tasa de código $\frac{4}{8}$ está protegido por el estándar Hamming(8, 4).
código. Pero los resultados de nuestro análisis final muestran que esta suposición requiere algunas modificaciones.
determinar la posición de cada bit de datos/paridad en la palabra de código, podríamos variar los bytes de envío
Pero se mantiene fijo un bit específico. Por ejemplo, para comprobar la posición del bit menos significativo del cuarto byte,
Establecemos los bytes de transmisión como 0x01, 0x03, 0x05. El bloque，0x0F.9 Entonces el bit que siempre es 1 en el
de intercalación es nuestro objetivo, es decir, LSB en este caso. Encontramos que los cuatro LSB en la palabra de código...
son bits de datos mientras que los cuatro MSB son bits de paridad, lo que difiere del estándar Hamming (8, 4)
Código p1p2d1p3d2d3d4p4, donde di es el i-ésimo bit de datos y pi es el i-ésimo bit de paridad. Conjunto de ecuaciones (14)
muestra la relación entre los bits de datos y los bits de paridad en el código Hamming(8, 4) estándar,

$$\begin{aligned} p1 &= d1 \oplus d2 \oplus d4 \\ p2 &= d1 \oplus d3 \oplus d4 \\ p3 &= d2 \oplus d3 \oplus d4 \\ p4 &= d1 \oplus d2 \oplus d3 \oplus d4. \end{aligned} \quad (14)$$

Denotamos los cuatro LSB como d1,d2,d3,d4 secuencialmente. Para encontrar qué bit en los MSB es pi , variamos
los bits de datos para mantener pi = 1. Por ejemplo, al seleccionar las palabras de código con $d1 \oplus d2 \oplus d4 = 1$,
El bit de paridad que siempre es igual a "1" es el bit de paridad p1. De forma similar, se derivan las posiciones de p2 y p3 .
Sin embargo, el bit de paridad restante no se ajusta a la definición de p4. Tras una cuidadosa observación,
encontramos que es una paridad que cubre d1,d2 y d3, es decir,

$$p5 = d1 \oplus d2 \oplus d3. \quad (15)$$

Para otras velocidades de código, podemos derivar la posición del bit de forma similar. Cuando CR = $\frac{4}{7}$ se utiliza el bit de paridad p1
se abandona. Cuando CR = $\frac{4}{6}$ Se utiliza, se omiten los bits de paridad p1 y p2 . Cuando se utiliza CR = $\frac{4}{5}$,
Solo hay un bit de paridad y es natural usar p4 para cubrir todos los bits. La conclusión no...
cambian cuando varía la SF. La Figura 10 muestra las posiciones de bits para diferentes velocidades de código. Tenga en cuenta que LoRa
aplica código Hamming no estándar, el número de nombre de paridad es arbitrario y nuestro nombre es
Sólo un tipo de ellos.

Deblanqueamiento. En la Sección 4.3, asumimos que la operación de deblanqueamiento ocurre después de “G” y
Demostrar que la posición de desblanqueamiento no afecta los resultados finales. Aquí analizamos la posición correcta para el
desblanqueamiento. El proceso de desentrelazado y decodificación de Hamming ha sido...

9Tenga en cuenta que no podemos controlar directamente los bits de paridad.

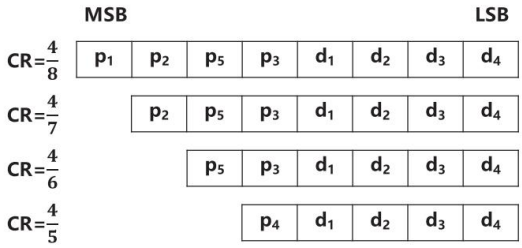

Fig. 10. Posición de los bits de paridad y de los bits de datos en una palabra de código con CR = $\frac{4}{8}, \frac{4}{7}, \frac{4}{6}, \frac{4}{5}$.

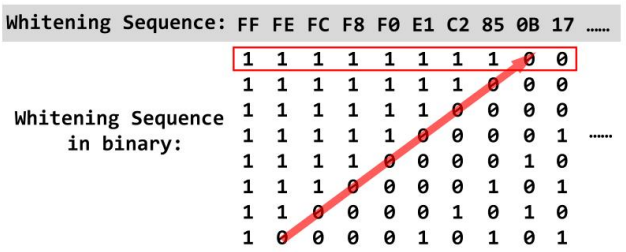

Fig. 11. Secuencia de blanqueamiento de “G→I→H→W”.

Lo anterior se interpretó claramente. Movemos la posición de "W" a las otras dos posiciones y enviamos Bytes de ceros para derivar la secuencia de blanqueamiento bajo diferentes selecciones de orden. Encontramos que “G→W→I→H” proporciona diferentes secuencias de blanqueamiento para varias combinaciones de SF y CR mientras La secuencia de blanqueamiento de “G→I→W→H” y “G→I→H→W” se mantiene igual. Es más razonable usar la misma secuencia para todos los paquetes. Por lo tanto, primero excluimos “G→W→I→H”. Luego, necesitamos… Elija el orden correcto entre “G→I →W→H” y “G→I →H→W”. La hoja de datos del chip LoRa [22] menciona que la secuencia de blanqueamiento en modo FSK se genera mediante un registro de desplazamiento de retroalimentación lineal . (LFSR). Suponemos que también existe un LFSR que genera la secuencia de blanqueamiento para el modo LoRa. Aplicamos el algoritmo Berlekamp–Massey [38] en las secuencias de blanqueamiento de los dos decodificadores. Órdenes para obtener el LFSR correspondiente. Para “G→I→W→H”, no importa cómo cambiemos la entrada orden de los bits (por ejemplo, LSB primero o MSB primero), el tamaño mínimo de LFSR que obtenemos del Berlekamp– El algoritmo de Massey tiene al menos 64. Pero sabemos que siempre se puede construir una secuencia de 2n bits a partir de un LFSR de n bits. Cualquier secuencia de 128 bits podría representarse como la salida de un LFSR de 64 bits. El LFSR de la secuencia "G→I→W→H" es demasiado largo. Se está revisando cuidadosamente la secuencia de blanqueamiento. de “G→I→H→W” como se muestra en la Figura 11, encontramos que no es como los bits aleatorios típicos, y cada El byte parece ser el estado de un LFSR. Recopilamos los MSB de los bytes de secuencia, como se muestra en la Recuadro rojo de la Figura 11, para ejecutar el algoritmo de Berlekamp-Massey. El polinomio LFSR para derivar Dicha secuencia es x8 + x6 + x5 + x4 + 1. Se puede observar en la literatura [39] que es el polinomio de longitud máxima para un registro de desplazamiento de 8 bits. La Figura 12 muestra la estructura del LFSR, y podemos Ver que los ocho bits del registro componen un byte de la secuencia de blanqueamiento, y cada bit del registro se utiliza para blanquear datos de un bit. Dado que la secuencia de blanqueamiento de “G→I→H→W” puede ser generado por un LFSR con una longitud mucho menor, consideramos que es el orden correcto.

4.5 CRC

Después de cuatro pasos de codificación Gray, desentrelazado, codificación Hamming y deblanqueamiento, el LoRa sin procesar Se muestran los paquetes físicos. En secciones anteriores, enviamos paquetes en modo de encabezado implícito y deshabilitamos Comprobación CRC. A continuación, analizamos el algoritmo CRC utilizado en la carga útil de LoRa. Según nuestra observación en la Sección 4.2, la suma de comprobación CRC ocupa 16 bits al final de un paquete. Por lo tanto, primero...

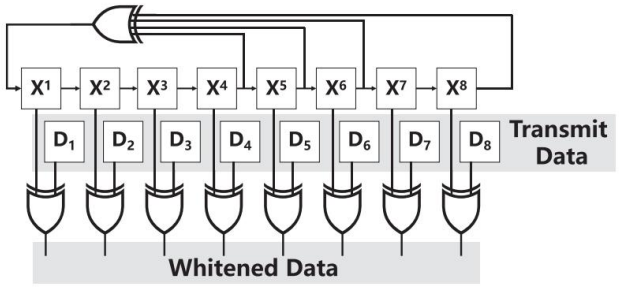

Fig. 12. LFSR x8 + x6 + x5 + x4 + 1 utilizado en LoRa.

Envía paquetes de encabezado explícito e implícito con el mismo contenido de datos para verificar la cobertura del CRC. Los resultados muestran que el CRC solo se realiza en la carga útil. Calcular la suma de comprobación CRC de una serie de bits, en teoría, implica considerarla como un número binario grande y calcular el resto con respecto a un "divisor". El divisor generalmente se representa como un polinomio basado en GF (2) (campo de Galois de dos elementos), p. ej., x3 + x + 1. Analizar la parte CRC de LoRa PHY es, de hecho, encontrar el polinomio utilizado. Una característica del CRC es que, para un polinomio de grado n, si el dividendo original es 1, entonces el resto es el propio polinomio. Si construimos un paquete cuyo último bit sea solo 1, entonces la suma de comprobación CRC en los dos últimos bytes nos dice exactamente qué polinomio se utiliza. Mientras tanto, el polinomio CRC no puede seleccionarse aleatoriamente; debe seguir ciertos principios para garantizar ciertos requisitos específicos. Por lo tanto, es natural elegir el polinomio de entre los polinomios CRC estándar. Al comparar el polinomio que derivamos con los conjuntos de polinomios estándar, se requiere poco esfuerzo para encontrar el polinomio real utilizado. Con base en el análisis anterior, habilitamos el CRC de carga útil en modo de encabezado implícito y establecemos los dos últimos bytes de la carga útil como 0x0001, 0x0080, 0x0100 y 0x8000, respectivamente (los demás bytes se establecen en cero). Dado que no estamos seguros del orden endian y LSB-MSB en las implementaciones de hardware de CRC, probamos las cuatro combinaciones posibles. Sin embargo, el resultado, sorprendentemente, superó nuestras expectativas. Independientemente de lo que establezcamos en los dos últimos bytes de carga útil, la suma de comprobación de CRC es exactamente la misma que la de los dos últimos bytes de carga útil.
En la implementación común de CRC, antes de calcular el resto, se rellenan n bits de cero después de los datos para garantizar que los últimos n bits también estén completamente protegidos por CRC.¹⁰ El fenómeno observado significa que el paso de relleno de ceros no está implementado en la capa física LoRa. Por lo tanto, enviamos 0x0001, 0x0080, 0x0100, 0x8000 en el tercer y cuarto byte desde el último para derivar el polinomio. Los bytes de CRC recibidos son 0x1021, 0x9188, 0x3331 y 0x1B98, respectivamente. 0x1021 se refiere al polinomio CCITT-16, que equivale a x16 + x12 + x5 + 1. Aplicamos la comprobación de CRC con este polinomio y probamos paquetes con bytes aleatorios. Las sumas de comprobación de CRC calculadas por nuestra implementación son consistentes con los bytes de CRC en todas las muestras. Debido a las características del CRC, es difícil que una implementación de CRC incorrecta tenga la misma suma de comprobación que la correcta, incluso en un grupo pequeño de muestras, lo que significa que nuestro resultado es correcto.

4.6 Encabezado

La parte desconocida de la capa física LoRa es el encabezado. En la Sección 4.2, ya sabíamos que el encabezado en modo explícito tiene 20 bits. Nuestro objetivo es determinar la organización y el significado de los 20 bits. Se dice que la longitud de carga útil (PL) está codificada en el encabezado y ocupa al menos 8 bits, ya que la longitud máxima de carga útil es de 255. Variamos la longitud de carga útil e intentamos decodificar el paquete utilizando el mismo proceso de decodificación en modo de encabezado implícito, asumiendo que el encabezado también está blanqueado. Desafortunadamente, no podemos recuperar los bytes de datos en modo de encabezado implícito.

10Aunque la implementación a nivel de código puede no contener explícitamente el paso de relleno de ceros [40].

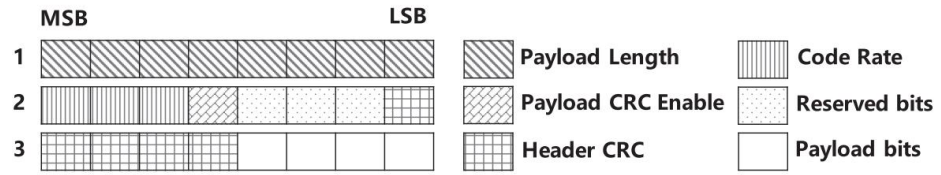

Fig. 13. Estructura del encabezado de LoRa en los primeros tres bytes.

El problema proviene de la secuencia de blanqueamiento utilizada para el encabezado. Sin embargo, al observar los bits que cambian con la longitud de la carga útil, podemos identificar la posición de PL. Nótese que en los primeros 2.5 bytes, solo los bits de PL y el CRC del encabezado cambiarán al variar la longitud de la carga útil del paquete. Observe que los bits en las posiciones 1 a 8 y 16 a 20 son posibles bits PL. Además, el intermedio Los resultados nos muestran que el encabezado no está blanqueado y el primer byte de un encabezado antes del blanqueamiento es Exactamente PL. Nuestro LFSR derivado solo puede generar 255 bytes de blanqueamiento posibles. No hay más. Bytes de blanqueamiento adicionales para la desblancura del encabezado. Por lo tanto, es razonable no blanquear el paquete. Encabezado. Nuestras siguientes pruebas para encontrar otros bits refuerzan esta suposición. Por lo tanto, no... Aplicar la operación de desblanqueo al encabezado en lo siguiente. Cambiar un parámetro y corregir otros. Los parámetros, la posición de la tasa de código y el bit de habilitación de CRC de carga útil se determinan de manera similar. Los bits de velocidad de código son 001, 010, 011 y 100 para $CR = \frac{4}{5}, \frac{4}{6}, \frac{4}{7}, \frac{4}{8}$ respectivamente. El bit de habilitación de CRC de carga útil es Se establece en 1 si se activa el CRC de la carga útil. En nuestros experimentos, hay cinco bits que cambian rápidamente cuando Estableciendo diferentes parámetros. Asumimos que estos cinco bits son el CRC del encabezado. Hay tres bits que mantienen... cero, independientemente de la configuración del parámetro, y los consideramos bits reservados. La figura 13 ilustra La estructura del encabezado en los primeros tres bytes.

No pudimos encontrar un polinomio CRC5 estándar que satisfaga los resultados CRC del encabezado utilizando el método en la Sección 4.5. Pero gracias a la linealidad del algoritmo CRC, podríamos usar una matriz CRC para representar de forma equivalente el cálculo CRC.[11] Denotamos 12 bits de parámetro (PL, CR y

El bit de habilitación de CRC de carga útil se representa como un vector de bits v1. Los cinco bits de CRC de encabezado se representan como un vector de bits v2. Nuestro objetivo es encontrar una matriz M que satisfaga v2 = M · v1. Dado que el valor de v1 está bajo nuestro control, Podemos diseñar una serie de v1, por ejemplo, $v_1^{(1)}, v_1^{(2)}, \ldots, v_1^{(12)}$ Forman una matriz V1. La serie relativa v2 es $v_2^{(1)}, v_2^{(2)}, \ldots, v_2^{(12)}$ Forman una matriz V2. Por lo tanto, V2 = M · V1. Si V1 es una matriz identidad, entonces M = V2.

¿Cómo crear una matriz identidad V1? Aunque podemos controlar el contenido del encabezado, No se puede controlar con precisión el estado de cada bit. En nuestro caso, parece imposible enviar un paquete con v1 que contiene solo un "1". Sin embargo, para nuestra sorpresa, el chip LoRa inesperadamente admite enviando un paquete con longitud de carga útil cero. Los ocho bits PL se convierten entonces en ceros. Si deshabilitamos carga útil CRC y establece CR = entonces $\frac{4}{5}$, solo un bit de tasa de código $\frac{4}{6}$ o CR = $\frac{4}{8}$, en v1 es 1. ¿Es? ¿Es posible que solo el bit de habilitación de CRC de carga útil sea "1"? La respuesta es, de nuevo, usar la linealidad de CRC. Supongamos que enviamos dos paquetes con carga útil cero y CR = 5. El paquete A tiene CRC de carga útil, pero El paquete B no lo hace. Entonces los vectores de bits correspondientes son $vA_1 = 000000000011, vA_2 = 01100, vB_1 = 000000000010, vB_2 = 00111$. Por lo tanto,

$$\begin{aligned} M \cdot (000000000001) = M \cdot vA_1 \; vB_1 \\ = M \cdot vA_1 \quad M \cdot vB_1 \\ = vA_2 \quad vB_2 \\ = 01011. \end{aligned} \tag{16}$$

[11] Aquí ignoramos los bits reservados.

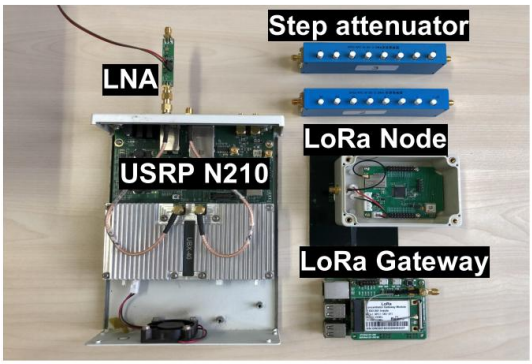

Fig. 14. Dispositivos en nuestros experimentos: un dispositivo SDR USRP N210 con LNA adicional, atenuadores de paso para medir la sensibilidad en entornos cableados, nodos LoRa básicos y una puerta de enlace LoRa básica.

Se puede aplicar un proceso similar para los bits PL. Finalmente, resumimos el cálculo del CRC del encabezado como v2 = M · v1, donde

$$M = \begin{matrix} 111100000000 \\ 100011100001 \\ 010010011010 \\ 001001010111 \\ 000100101111 \end{matrix} . \quad (17)$$

5 IMPLEMENTACIÓN

Implementamos la física completa de LoRa en C++ en la plataforma de radio definida por software GNU Radio. También proporcionamos código en MATLAB para simulación. Diseñamos el receptor como dos bloques separados: demodulador y decodificador (para el transmisor: modulador y codificador). De esta forma, en el futuro podremos integrar nuevos mecanismos de demodulación en nuestra implementación sin modificar el bloque decodificador. Por ejemplo, si reemplazamos el demodulador LoRa por un demodulador de colisión, podremos decodificar paquetes de colisión directamente sin modificar la lógica del bloque decodificador. Todos los experimentos de la Sección 6 se realizan en USRP N210 en tiempo real (no guardamos las señales de banda base sin procesar para su procesamiento fuera de línea). El USRP está conectado a un portátil con Ubuntu 19.10, CPU i5-7200U y 8 GB de RAM. Utilizamos dispositivos LoRa básicos para evaluar el rendimiento de nuestra capa física LoRa implementada, incluyendo nodos finales LoRa con SX1268 [25]/SX1278 [22] y puertas de enlace LoRa Raspberry-Pi 3B+ (RPI) con SX1301 [37]. La Figura 14 muestra los dispositivos utilizados en nuestro El atenuador de pasos se utiliza en experimentos con cables para mediciones de sensibilidad.

6 EVALUACIÓN

6.1 Tasa de éxito de decodificación.

Verificamos la eficacia de nuestra capa física LoRa implementada probando si puede decodificar los paquetes LoRa transmitidos desde los dispositivos LoRa estándar. Configuramos un dispositivo LoRa estándar para transmitir repetidamente paquetes aleatorios a nuestro receptor LoRa implementado. La verdad fundamental de los bytes transmitidos se registra a través del puerto serie. Para una evaluación exhaustiva, ajustamos el transmisor LoRa con diferentes parámetros, incluyendo seis SF ($\frac{4}{5}$, $\frac{4}{6}$, $\frac{4}{7}$, $\frac{4}{8}$), cuatro CR ( ) y dos modos de encabezado (explícito/implícito), con/sin CRC, es decir, 6 × 4 × 2 × 2 = 96 combinaciones. Utilizamos una frecuencia portadora de 475 MHz y un ancho de banda de 250 kHz.12 La longitud de la carga útil se establece en

12La frecuencia portadora y el ancho de banda no afectan la lógica de decodificación.

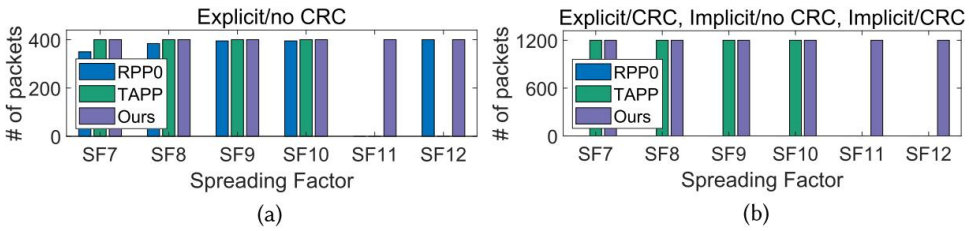

Fig. 15. Prueba de tasa de éxito de decodificación. Se envían un total de 9600 paquetes y se prueban todas las tasas de código $\binom{4}{5}, \binom{4}{6}, \binom{4}{7}, \binom{4}{8}$. RPP0 no puede decodificar ningún paquete SF11. TAPP no puede decodificar ningún paquete SF11/12. Nuestro decodificador pudo decodificar todos los paquetes. (a) Número de paquetes recibidos correctamente de tres implementaciones en modo de encabezado explícito con CRC desactivado. (b) RPP0 no admite la configuración de {modo de encabezado explícito con CRC activado, modo de encabezado implícito con CRC desactivado, modo de encabezado implícito con CRC activado} y, por lo tanto, no recibe ninguno.

16 bytes. Para cada configuración, la transmisión se repite 100 veces. Colocamos el transmisor LoRa cerca del receptor USRP, por lo que todas las señales se reciben con una relación señal/ruido (SNR) alta. Dado que el entorno de comunicación es ideal, cualquier paquete perdido indica que la implementación es incompleta.

El resultado del experimento se muestra en la Figura 15. Entre todas las configuraciones, BR solo funciona en SF8 con encabezado implícito, mientras que RPP0 solo admite el encabezado explícito. Ninguno de estos dos métodos admite CRC. Dado que los procesos de decodificación de BR y TAPP son computacionalmente costosos, no funcionaron con paquetes de gran longitud a altas velocidades de transmisión. Específicamente, BR no puede decodificar paquetes de más de 5 bytes, y TAPP comienza a perder paquetes cuando el ciclo de trabajo del transmisor es superior a 0,5. Debido a todas estas limitaciones, el PRR general de BR solo alcanza el 4,2 %. Además, podemos observar que TAPP no decodifica ningún paquete SF11/SF12, y RPP0 no puede resolver paquetes SF11. En resumen, RPP0 cubre 1924/9600 ≈ 20,0 % de los paquetes y TAPP cubre 6400/9600 ≈ 66,7 % de los paquetes. Por el contrario, nuestro decodificador puede decodificar todos los paquetes bajo cualquier configuración LoRa.

6.2 Sensibilidad. En

esta sección, verificamos la sensibilidad de nuestro decodificador LoRa. Conectamos un transmisor LoRa estándar a nuestro decodificador mediante un cable RF de 20 m con atenuadores de dos pasos. De esta forma, podemos controlar con precisión la atenuación de la señal ajustando el valor de estos atenuadores.

Usar el valor de los atenuadores de paso como la atenuación del canal no es preciso, porque los enlaces cableados, incluyendo cables y conectores de RF, también introducen atenuación que conduce a errores de estimación de sensibilidad. Por lo tanto, antes del experimento, usamos un analizador de espectro, Keysight N9322C, para estimar y calibrar con precisión la atenuación del enlace. Luego, ajustamos la potencia de transmisión a su nivel más bajo, es decir, −7.9 dBm, para evitar fugas del canal inalámbrico. Experimentamos con la configuración de SF8 y un ancho de banda de 250 kHz. Dado que BR no puede decodificar paquetes de más de cuatro bytes, no comparamos su rendimiento para el resto de esta sección. La Figura 16(a) muestra los resultados de sensibilidad de cuatro decodificadores SDR. El criterio de sensibilidad en LoRa se define como el RSSI mínimo que logra PRR > 90% cuando la carga útil es de 32 bytes. Dado que RPP0 y TAPP no aprovechan las características de LoRa para la demodulación, su PRR disminuye rápidamente a medida que disminuye el RSSI de la señal recibida. Las sensibilidades de RPP0 y TAPP son de -106 y -108 dBm, respectivamente. En cambio, el rendimiento de nuestro decodificador se mantiene estable y alcanza una sensibilidad de tan solo -126 dBm.

Realizamos un experimento adicional con un factor de seguridad (SF) mayor, es decir, SF12. El resultado se muestra en la Figura 16(b). Nuestra sensibilidad medida es de -142 dBm, que se acerca a la sensibilidad óptima de LoRa. Curiosamente, la sensibilidad de la puerta de enlace RPI es de tan solo -139,5 dBm. Esto se debe a que la puerta de enlace RPI no adopta el diseño óptimo de la puerta de enlace LoRa. Por lo tanto, nuestro decodificador ofrece un mejor rendimiento.

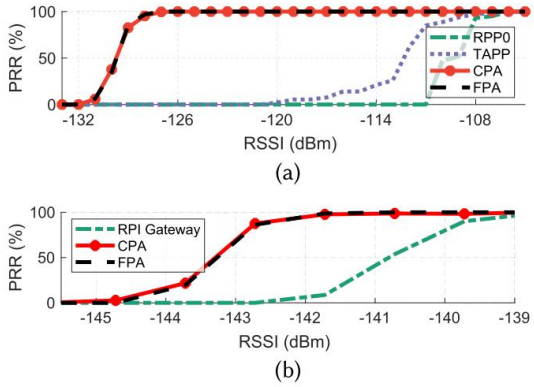

Fig. 16. Pruebas de sensibilidad de diferentes decodificadores con (a) carga útil de 10 bytes, SF8, BW = 250 kHz y (b) carga útil de 32 bytes, SF12, BW = 125 kHz.

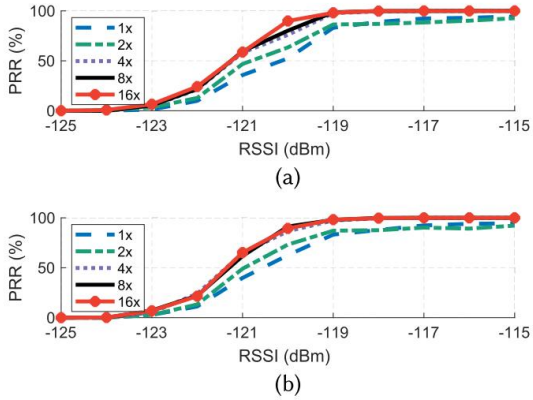

Fig. 17. Influencia de la relación de relleno de ceros en (a) CPA y (b) FPA con k = 8.

6.3 Influencia del relleno de ceros. A

continuación, verificamos la validez de nuestra estrategia de refinamiento de picos propuesta, que mejora la resolución de frecuencia mediante el relleno de ceros. Utilizamos la misma configuración experimental que en la Sección 6.2, evaluando los métodos FPA y CPA. La Figura 17 muestra el rendimiento del receptor con diferentes niveles de RSSI y relaciones de relleno de ceros. Tanto para FPA como para CPA, observamos que el PRR aumenta con una relación de relleno de ceros r mayor. La mejora del rendimiento para r de 1 a 4 es observable, mientras que para r ≥ 4 la mejora del rendimiento se vuelve insignificante.

6.4 Prueba del codificador

En esta sección, presentamos la evaluación de nuestro codificador LoRa implementado, que es una versión inversa del decodificador LoRa. La única diferencia entre la implementación del codificador y la del decodificador reside en los bytes redundantes para rellenar los últimos símbolos. Aunque estos bytes $\frac{4}{CR}$ parecen fijos en la implementación del chip LoRa, carecen de significado en el receptor. En nuestra implementación, rellenamos con ceros los bytes restantes. Cuando el codificador está activo, abre un puerto UDP y espera datos.

Enviamos 16 bytes al proceso del codificador mediante un script de Python. Posteriormente, la señal LoRa generada se envía desde USRP a un dispositivo LoRa. Verificamos las salidas del puerto serie del dispositivo para comprobar si los datos se transfieren correctamente. Utilizamos diferentes parámetros (es decir, diferentes SF, CR y

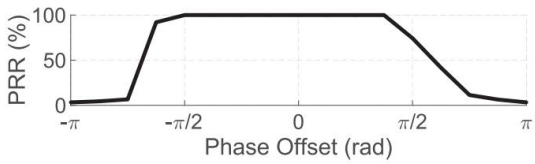

Fig. 18. PRR del chip LoRa al recibir señales LoRa especialmente construidas con diferentes valores de desalineación de fase.

Ancho de banda) para el envío de datos. Además, para cada parámetro, la transmisión se repite 100 veces. Los resultados finales muestran que 9600 paquetes enviados desde nuestro codificador se decodificaron correctamente, lo que, desde otra perspectiva, demuestra la fiabilidad de nuestro análisis e implementación.

6.5 Influencia de la desalineación de fase En la

Sección 3, proponemos FPA y CPA para la demodulación LoRa. Demostramos que el rendimiento de ambos métodos se acerca teóricamente a la demodulación IDEAL. Esto se debe a que son resistentes a la fluctuación de fase causada por hardware imperfecto y multitrayecto. Entonces, ¿cuál es el método de demodulación real implementado en el chip LoRa comercial? ¿Adopta FPA y CPA? Curiosamente, encontramos que el algoritmo de demodulación utilizado en el chip LoRa no es tan bueno como FPA o CPA. Construimos paquetes LoRa que contienen cinco bytes especialmente diseñados. Como resultado, los cuatro símbolos de datos consecutivos son los mismos con fstart = 0, es decir, un símbolo con una caída de frecuencia pronunciada en el medio.
Este símbolo es más vulnerable a la desalineación de fase entre los dos segmentos de chirp. Antes de enviar el paquete vía SDR, añadimos manualmente el desfase en ambos segmentos. Nuestro receptor, que utiliza FPA y CPA, es resistente a la desalineación de fase y su PPR es del 100 %. Sin embargo, observamos que la puerta de enlace LoRa experimenta pérdidas frecuentes de paquetes al recibir paquetes con una desalineación de fase significativa. La Figura 18 muestra la PRR con respecto a la desalineación de fase. Su PRR es casi del 100 % cuando el desfase está entre $(-\pi \quad \frac{\pi}{2}, \frac{\pi}{2})$. Sin embargo, disminuye rápidamente cuando el desfase es grande. Como resultado, el chip LoRa adopta un método de demodulación más débil que nuestros métodos.
En otras palabras, el chip LoRa comercial es vulnerable a ataques de desalineación de fase. A pesar de la dificultad de este tipo de ataque, cualquier pequeño defecto en el área de seguridad es importante tener en cuenta. Suponemos que el chip LoRa puede compensar un desfase fijo Δφ durante el proceso de demodulación; por ejemplo, estima Δφ a partir del preámbulo y lo aplica a los siguientes símbolos de datos. Si Δφ cambia, el chip LoRa no demodula el paquete correctamente. Por lo tanto, el PRR disminuye cuando cambia el desfase. Por supuesto, desconocemos la implementación interna del chip LoRa a menos que Semtech publique los documentos. En resumen, la implementación de demodulación del chip LoRa es más débil que nuestros métodos FPA y CPA.

6.6 Prueba al aire libre

Finalmente, comparamos nuestro decodificador con una puerta de enlace RPI estándar en un entorno real. Colocamos el receptor fuera de una ventana en el segundo piso. Seleccionamos cuidadosamente la ubicación de los transmisores, asegurándonos de que no haya una línea de visión directa entre el transmisor y los receptores, lo cual refleja la situación habitual de la comunicación LoRa. El transmisor envía repetidamente un paquete de 32 bytes en modo de encabezado explícito con la configuración SF12, BW = 125 kHz, CR = y CRC habilitado. $\frac{4}{8},$
Elevamos el transmisor a 1,8 m de altura con un trípode. La Figura 19 muestra la configuración del experimento mencionada anteriormente. La Figura 20 muestra el PRR a diferentes distancias de comunicación. Con un PRR del 95 % como barra, nuestro decodificador admite un alcance de comunicación de hasta 3600 m, mientras que la puerta de enlace RPI estándar solo alcanza un máximo de 2800 m. Un detalle interesante es la fluctuación del PRR de la puerta de enlace RPI a 2400 m. Suponemos que la causa es el desfase causado por el complejo...

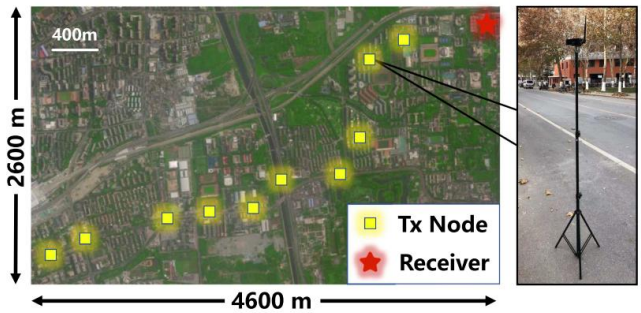

Fig. 19. Despliegue del transmisor y el receptor. El transmisor está colocado sobre un trípode de 1,8 m.

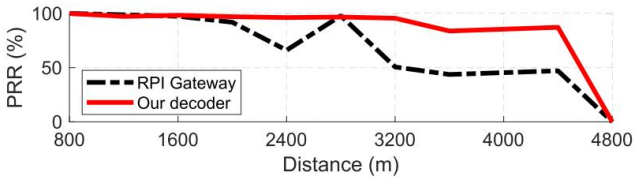

Fig. 20. Experimentos de alcance de comunicación al aire libre.

Entorno. Como se mencionó en la Sección 6.5, el chip LoRa estándar es vulnerable a problemas de desalineación de fase. Por lo tanto, nuestro decodificador parece ser más confiable en comunicaciones exteriores.

7 TRABAJOS RELACIONADOS

Ingeniería inversa de protocolos inalámbricos. El rápido desarrollo del Internet de las Cosas (IoT) ha dado lugar a una pluralidad de protocolos propietarios. La motivación endógena para desarrollar protocolos cerrados en lugar de utilizar los protocolos estándar existentes proviene de diversas razones. Por ejemplo, un protocolo propietario podría ahorrar costes de licencia a los fabricantes del IoT, o el nuevo protocolo ofrece funciones específicas dirigidas a aplicaciones y dispositivos integrados específicos. Sin embargo, desviarse de las especificaciones estándar a veces facilita la entrada de atacantes, ya que los fabricantes podrían introducir diseños inseguros. Además, estos protocolos suelen estar indocumentados y cerrados al público, lo que atrae a los investigadores de redes a aplicar ingeniería inversa a ellos, incluyendo RFID criptográfico [41, 42, 53], red inalámbrica para automóviles [43], protocolo ad hoc Apple Wireless Direct Link [44, 45], etc. La ingeniería inversa de estos protocolos profundiza nuestra comprensión de las redes existentes y revela fallos en su diseño. Nuestro trabajo también busca proporcionar a otros investigadores una comprensión más completa de la capa física de LoRa y proporcionar una mejor herramienta para analizar la red LoRa.

Trabajos relacionados con LoRa PHY. LoRa proporciona una modulación única que ofrece un largo alcance, lo cual resulta atractivo en muchas aplicaciones. La tecnología tradicional de retrodispersión se ve afectada por el alcance de comunicación, que se limita a un metro. Trabajos recientes sobre retrodispersión que combinan LoRa [12, 47, 52] mejoran el alcance a kilómetros. Mientras tanto, al adoptar una modulación similar a la de LoRa, Netcatter [13] admite 256 transmisiones de retrodispersión actuales. Debido a la baja velocidad de datos y las características de gran cobertura de LPWAN, la capacidad de la red es relativamente pequeña [46], lo que genera problemas cruciales de colisión. Se han realizado numerosos esfuerzos para mejorar el rendimiento de la red LoRa. Otros trabajos representativos son la decodificación de colisiones [1–4, 9, 10, 27, 49] y la decodificación de señales débiles [50, 51]. Estos se basan principalmente en la singularidad de la demodulación de LoRa PHY para

chirridos separados para paquetes de diferencia. Nuestro trabajo proporciona una comprensión más profunda de LoRa PHY y Impulsará la investigación relacionada con LoRa PHY.

8 CONCLUSIÓN

Este artículo presenta una comprensión integral de la demodulación y decodificación de LoRa y revela razones fundamentales de la brecha de rendimiento entre los trabajos existentes y el LoRa comercial.
Este trabajo es la primera implementación completa de LoRa PHY con una garantía de rendimiento demostrable.
Al chip LoRa de caja negra. Mejoramos la demodulación para lograr valores extremadamente bajos.
Decodificación SNR (−20 dB) con garantía teórica de rendimiento. Además, derivamos el orden y
parámetros de las operaciones de decodificación, incluyendo el desblanqueamiento, la corrección de errores, el desentrelazado y
Así sucesivamente, aprovechando los datos oficialmente conocidos de LoRa y la manipulación de paquetes. Además,
implementamos la primera capa física LoRa SDR en tiempo real completa en la plataforma GNU Radio. La evaluación...
muestra que nuestro método puede lograr (1) una tasa de éxito de decodificación del 100% mientras que los métodos existentes pueden
soporte máximo del 66,7%; (2) sensibilidad de −142 dBm, que es el límite del LoRa básico; y
(3) un alcance de comunicación de 3.600 m en el área urbana, de manera similar al LoRa comercial bajo el mismo
configuración.

APÉNDICE

Modelo de cálculo SER. Supongamos que la señal se transmite a través de un canal AWGN, donde
El ruido sigue la distribución gaussiana compleja, es decir, CN (0, σ2). Luego, tras la descodificación, el ruido
La distribución cambia a CN (0, Mσ2), donde M es el número de muestras dentro de la demodulación.
ventana. La altura máxima del ruido tiene una pequeña varianza y se puede aproximar como cσ, donde
c es una constante. Supongamos que la altura del pico de los datos hd sigue una distribución gaussiana N (μd σ_d^2. Tenemos

$$\mathrm{SER} = P\,(h_d < c\sigma) = Q\left(\frac{c\sigma - \mu_d}{\sigma_d}\right), \tag{18}$$

Donde Q es la función de cola de la distribución normal estándar. La relación señal-ruido (SNR) se representa $\frac{A^2}{\sigma^2}$, donde A
como Γ = la amplitud de la señal.

IDEAL. Dado que la multiplicación y la FFT son lineales, el pico de datos complejos en IDEAL sigue a los complejos.
Distribución gaussiana CN (h, Mσ2), donde h = MA. La altura del pico sigue la distribución de Rice.
que puede aproximarse como distribución gaussiana N (μ1, σ2 $_1$), donde $\mu_1 = \sigma\sqrt{\frac{M\pi}{2}}L_{\frac{1}{2}}\left(-\frac{h^2}{2M\sigma^2}\right)$ y
$\sigma_1^2 = 2M\sigma^2 + h^2 - \mu_1^2$. $L_{\frac{1}{2}}(\cdot)$ es un polinomio de Laguerre. Según la referencia [47], la altura máxima
La otra altura de ruido M −1 es aproximadamente $2M\sigma^2 HM-1 = c1\sigma$, donde Hl = representa el lth $\sum_{i=1}^{\infty}\frac{1}{i}$
Número armónico. Tomando μ1, σ1 y c1 en la ecuación (18), derivamos el SER de IDEAL como sigue:

$$\mathrm{SER1} = Q\left(\frac{\sqrt{2HM-1} - \sqrt{\frac{\pi}{2}}L_{\frac{1}{2}}\left(-\frac{M\Gamma}{2}\right)}{\sqrt{2 + M\Gamma - \frac{\pi}{2}L_{\frac{1}{2}}\left(-\frac{M\Gamma}{2}\right)}}\right). \tag{19}$$

FPA. Si la compensación de fase θ = 0, entonces el dominio de frecuencia después de la suma es igual a IDEAL.
Desplazar el ruido en θ no afecta la distribución de ruido de media cero. Por lo tanto, la distribución de ruido después de la
adición aún puede considerarse como CN (0, Mσ2), que representa la altura máxima del ruido.
isc2σ, donde c2 = c1. El pico de datos en FPA sigue a CN (h1 + ejθh2, Mσ2), donde h1 (h2) es el primero.
(segundo) segmento de chirrido, y h1 + h2 = h. Sea h3 = |h1 + ejθh2 |. La altura del pico se deduce
Distribución del arroz, que puede aproximarse como N (μ2, σ2 $_2$), donde $\mu_2 = \sigma\sqrt{\frac{M\pi}{2}}L_{\frac{1}{2}}\left(-2\frac{h_3^2}{2M\sigma^2}\right)$, σ_2^2 =
$2M\sigma^2 + h_3^2 - \mu_2^2$. SER de FPA está relacionado con el símbolo transmitido, y aquí simplificamos el envío.
símbolo con h1 = h2 = $\frac{h}{2}$. El mejor caso es que θ = 0 y h3 = h. El peor caso es que |θ | = y $\frac{\pi}{k}$

h3 = h cos(el $\frac{\pi}{2k}$). Consideramos el promedio de ellos h · $\frac{1+\cos(2k\frac{\pi}{\quad})}{2}$ = h cos2 ($\frac{\pi}{4k}$) como h3 final. Por lo tanto, SER de FPA se puede aproximar como

$$\text{SER2} = Q \frac{\sqrt{2HM-1} - \frac{\pi}{2} L \frac{1}{2} - \frac{M\Gamma}{2} \cos4 (\frac{\pi}{4k})}{2 + M\Gamma \cos4 \pi \frac{}{4k} - \frac{\pi}{2} L2 \frac{1}{2} - \frac{M\Gamma}{2} \cos4 \pi \frac{}{4k}} . \quad (20)$$

CPA. Las dos alturas de pico de datos separados en CPA siguen la distribución de Rice, respectivamente. La suma de ellos tiene la distribución aproximada N (μ3, σ2 $_3$), donde μ3 = μ 3 + μ 3 , h2h σ^2_3 = 2Mσ2 + h^2_1 + h^2_2 − (μ3) ^{2} − (μ $_3$) 2, μ 3 (- 2Mσ 2), $\frac{M\pi h1}{\mu\ 3\ L\ 1}$ L $\frac{1}{2}$ $\frac{h1h}{}$ = σ $\frac{M\pi h2}{La}$ $_2$ (− 2Mσ 2). El pico de ruido altura de 2 h sigue la distribución de Rayleigh. Denote la suma del valor absoluto de dos partes del ruido como Yi (i = 1, 2,..., M − 1). Yi sigue aproximadamente N (μY , σ^2_Y), dónde

$$\mu Y = h1 + h2 \frac{}{} \frac{}{} \frac{M\pi}{2\ horas} \sigma,\ \sigma^2_Y = \frac{4-\pi}{2} M\sigma2 . \quad (21)$$

Sea S = maxi=1,2,...,M−1 Yi . Por la desigualdad de Jensen [48],

$$\begin{aligned} etE(S) \le E\ et\ S &= E_{maxi}\ etYi \\ &\le \sum_{i=1}^{M-1} E\ etYi = (M-1)e\ \mu Y\ t + \frac{1}{2}\sigma^2_Y t^2, \end{aligned} \quad (22)$$

donde la última igualdad se sigue de la definición de la función generadora del momento gaussiano. Tomando el logaritmo en ambos lados de la desigualdad (22), tenemos

$$E(S) \le \frac{\ln(M-1)}{el} + \mu Y + \frac{\sigma^2_Y\ el}{2} . \quad (23)$$

Para t > 0, según la desigualdad de medias aritméticas y geométricas, sabemos

$$E(S) \le \mu Y + 2\sqrt{\ln(M-1)\sigma Y} = c3\sigma, \quad (24)$$

donde c3 = (√ h1 + √ h2) $\frac{}{}$ $\frac{M\ \pi}{2\ horas}$ + $\sqrt{(4-\pi)M \ln(M-1)}$. Usamos el límite superior de E(S) para estimar la Altura máxima del pico de ruido en el método CPA. Considere la situación h1 = h2 = $\frac{h}{2}$; tenemos

$$\text{SER3} = Q \frac{\sqrt{\pi} + \sqrt{(4-\pi)\ln(M-1)} - \sqrt{\pi} L \frac{1}{2} (-\frac{M\Gamma}{4})}{\sqrt{2 + 2\frac{M\Gamma}{} - \frac{\pi}{2} L2 \frac{1}{2} (-\frac{M\Gamma}{4})}} . \quad (25)$$

REFERENCIAS

[1] Shuai Tong, Zhenqiang Xu y Jiliang Wang. 2020. CoLoRa: Habilitación de la recepción multipaquete en LoRa. En actas. de la Conferencia Internacional IEEE sobre Comunicaciones por Computadora (INFOCOM'20). IEEE, 2303–2311.

[2] Xianjin Xia, Yuanqing Zheng, Tao Gu. 2020. FTrack: Decodificación paralela para transmisiones LoRa. IEEE/ACM Trans. Netw. 28, 6 (2020), 2573–2586. https://doi.org/10.1109/TNET.2020.3018020

[3] Zhenqiang Xu, Shuai Tong, Pengjin Xie y Jiliang Wang. 2020. FlipLoRa: Resolviendo colisiones con arriba-abajo Cuasi-ortogonalidad. En las Actas de la Conferencia Internacional del IEEE sobre Detección, Comunicación y Redes. (SEGUNDO'20).

[4] Zhe Wang, Linghe Kong, Kangjie Xu, Liang He, Kaishun Wu y Guihai Chen. 2020. Transmisiones concurrentes en línea En la puerta de enlace LoRa. En Actas de la Conferencia Internacional IEEE sobre Comunicaciones Informáticas (INFOCOM'20). IEEE, 2331–2340.

[5] Wenju Zhao, Shengwei Lin, Jiwen Han, Rongtao Xu y Lu Hou. 2017. Diseño e implementación de riego inteligente. Sistema basado en LoRa. En Actas de los Talleres IEEE Globecom (GC Wkshps'17). IEEE, 1–6.

[6] Irfan Fachrudin Priyanta, Frank Golatowski, Thorsten Schulz y Dirk Timmermann. 2019. Evaluación de la tecnología LoRa para el seguimiento de vehículos y activos en puertos inteligentes. En Actas de la 45.ª Conferencia Anual de la Sociedad de Electrónica Industrial del IEEE (IECON'19), vol. 1. IEEE, 4221–4228.

[7] Davide Magrin, Marco Centenaro y Lorenzo Vangelista. 2017. Evaluación del rendimiento de redes LoRa en un escenario de ciudad inteligente. En Actas de la Conferencia Internacional de Comunicaciones del IEEE (ICC'17). IEEE, 1–7.

[8] Umber Noreen, Ahcène Bounceur y Laurent Clavier. 2017. Estudio de la tecnología LoRa de redes de área amplia y de bajo consumo. En Actas de la Conferencia Internacional sobre Tecnologías Avanzadas para el Procesamiento de Señales e Imágenes (ATSIP'17). IEEE, 1–6.

[9] Rashad Eletreby, Diana Zhang, Swarun Kumar y Osman Yağan. 2017. Potenciación de redes de área amplia de bajo consumo en entornos urbanos. En Actas de la Conferencia ACM del Grupo de Interés Especial sobre Comunicación de Datos (SIGCOMM'17).

[10] Shuai Tong, Jiliang Wang y Yunhao Liu. 2020. Combatir las colisiones de paquetes mediante el escalado de señales no estacionarias en redes LPWAN. En las Actas de la Conferencia Internacional ACM sobre Sistemas, Aplicaciones y Servicios Móviles (MobiSys'20).

[11] Yao Peng, Longfei Shangguan, Yue Hu, Yujie Qian, Xianshang Lin, Xiaojiang Chen, Dingyi Fang y Kyle Jamieson. 2018. PLoRa: Una red pasiva de datos de largo alcance a partir de transmisiones ambientales LoRa. En Actas de la Conferencia ACM del Grupo de Interés Especial sobre Comunicación de Datos (SIGCOMM'18).

[12] Vamsi Talla, Mehrdad Hessar, Bryce Kellogg, Ali Najafi, Joshua R. Smith y Shyamnath Gollakota. 2017. Retrodispersión LoRa: Habilitando la visión de la conectividad ubicua. Proc. ACM Interact. Mobile Wear. Ubiq. Technol. 1, 3 (2017). 1–24.

[13] Mehrdad Hessar, Ali Najafi y Shyamnath Gollakota. 2019. Netscatter: Habilitación de redes de retrodispersión a gran escala. En Actas del Simposio USENIX sobre diseño e implementación de sistemas en red (NSDI'19). 271–284.

[14] Adwait Dongare, Revathy Narayanan, Akshay Gadre, Anh Luong, Artur Balanuta, Swarun Kumar, Bob Iannucci y Anthony Rowe. 2018. Charm: Explotación de la diversidad geográfica mediante la combinación coherente en redes de área amplia de baja potencia. En Actas de la 17.ª Conferencia Internacional ACM/IEEE sobre Procesamiento de la Información en Redes de Sensores (IPSN'18). IEEE, 60–71.

[15] Akshay Gadre, Revathy Narayanan, Anh Luong, Anthony Rowe, Bob Iannucci y Swarun Kumar. 2020. Configuración de frecuencia para redes de área amplia de baja potencia en un instante. En Actas del Simposio USENIX sobre Diseño e Implementación de Sistemas en Red (NSDI'20).

[16] Pieter Robyns, Peter Quax, Wim Lamotte y William Thenaers. 2018. Un decodificador de software multicanal para el esquema de modulación LoRa. En Actas de la Conferencia Internacional sobre Internet de las Cosas, Big Data y Seguridad (IoTBDS'18).

[17] Matthew Knight y Balint Seeber. 2016. Decodificación de LoRa: Implementación de una LPWAN moderna con SDR. En Actas de la Conferencia de Radio GNU.

[18] Joachim Tapparel, Orion Afisiadis, Paul Mayoraz, Alexios Balatsoukas-Stimming y Andreas Burg. 2020. Un prototipo de capa física LoRa de código abierto en radio GNU. En Actas del 21.º Taller Internacional del IEEE sobre Avances en el Procesamiento de Señales en las Comunicaciones Inalámbricas (SPAWC'20). 1–5. DOI: http://dx.doi.org/10.1109/SPAWC48557.2020.9154273

[19] Josh Blum. 2016. Módem LoRa con LimeSDR. Recuperado de https://myriadrf.org/news/lora-modem-limesdr/.

[20] RevSpace. DecodingLoRa. Recuperado de https://revspace.nl/DecodingLora.

[21] Alexandre Marquet, Nicolas Montavont y Georgios Z. Papadopoulos. 2019. Investigación del rendimiento teórico y técnicas de demodulación para LoRa. En Actas del Simposio Internacional IEEE sobre un Mundo de Redes Inalámbricas, Móviles y Multimedia (WoWMoM'19). IEEE, 1–6.

[22] Semtech. 2020. Hoja de datos SX1276/77/78/79, Rev. 7.

[23] Especificación LoRaWAN v1.1. Recuperado de https://lora-alliance.org/resource_hub/lorawan-specification-v1-1/.

[24] Symphony Link. Recuperado de https://www.link-labs.com/symphony.

[25] Semtech. 2019. Hoja de datos SX1268, Rev. 1.1.

[26] Olivier Bernard, André Seller y Nicolas Sornin. 2014. Transmisor de largo alcance y baja potencia. Patente EP 2 763 321 A1.

[27] Xiong Wang, Linghe Kong, Liang He y Guihai Chen. 2019. mLoRa: Un protocolo de recepción de múltiples paquetes en redes LoRa. En Actas de la Conferencia Internacional IEEE sobre Protocolos de Red (ICNP'19). IEEE, 1–11.

[28] Rajalakshmi Nandakumar, Vikram Iyer y Shyamnath Gollakota. 2018. Localización 3D para dispositivos subcentimétricos. En Actas de la Conferencia ACM sobre Sistemas de Sensores Integrados en Red (SenSys'18).

[29] Yinghui Li, Jing Yang y Jiliang Wang. 2020. DyLoRa: Hacia un control de transmisión LoRa dinámico y energéticamente eficiente. En Actas de la Conferencia Internacional IEEE sobre Comunicaciones Informáticas (INFOCOM'20). IEEE, 2312–2320.

[30] Nodo LoRaMac. Recuperado de https://github.com/Lora-net/LoRaMac-node.

[31] Proyecto SX1302 LoRa Gateway. Recuperado de https://github.com/Lora-net/sx1302_hal.

[32] ChirpStack. Recuperado de https://www.chirpstack.io/.

[33] Servidor LoRaWAN. Recuperado de https://github.com/gotthardp/lorawan-server.

[34] Descodificador de encantos. Recuperado de https://github.com/WiseLabCMU/charm-decoder.
[35] Mehrdad Hessar, Ali Najafi, Vikram Iyer y Shyamnath Gollakota. 2020. TinySDR: Plataforma SDR de bajo consumo para bancos de pruebas IoT programables por aire. En Actas del Simposio USENIX sobre Diseño e Implementación de Sistemas en Red (NSDI'20). 1031–1046.

[36] TinySDR. Recuperado de https://github.com/uw-x/tinysdr.
[37] Semtech. 2017. Hoja de datos SX1301, V2.4.
[38] Nadia Ben Atti, Gema M. Díaz-Toca y Henri Lombardi. 2006. Revisión del algoritmo berlekamp-massey. Aplicar. Álgebra. Ing. Comunitario. Computadora. 17, 1 (2006), 75–82.
[39] Wikipedia. LFSR. https://en.wikipedia.org/wiki/Linear-feedback_shift_register.
[40] Ross Williams et al. 1993. Una guía sencilla para los algoritmos de detección de errores CRC. https://zlib.net/crc_v3.txt.
[41] P. Fraga-Lamas y TM Fernández-Caramés. 2017. Ingeniería inversa del protocolo de comunicaciones de una tarjeta RFID de transporte público. En Actas de la Conferencia Internacional del IEEE sobre RFID (RFID'17). 30–35.
[42] Karsten Nohl, David Evans, Starbug Starbug y Henryk Plötz. 2008. Ingeniería inversa de una etiqueta RFID criptográfica. En Actas del Simposio de Seguridad USENIX, Vol. 28.
[43] Ishtiaq Rouf, Robert D. Miller, Hossen A. Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe e Ivan Seskar. 2010. Vulnerabilidades de seguridad y privacidad en redes inalámbricas para automóviles: Estudio de caso de un sistema de monitorización de la presión de los neumáticos. En Actas del Simposio de Seguridad USENIX, vol. 10.
[44] Milan Stute, David Kreitschmann y Matthias Hollick. 2018. La fórmula mágica de mil millones de manzanas: Receta para el protocolo ad hoc de enlace directo inalámbrico de Apple. En Actas de la 24.ª Conferencia Internacional Anual sobre Computación Móvil y Redes. 529–543.

[45] Milan Stute, Sashank Narain, Alex Mariotto, Alexander Heinrich, David Kreitschmann, Guevara Noubir y Matthias Hollick. 2019. Mil millones de interfaces abiertas para Eve y Mallory: MitM, DoS y ataques de rastreo en iOS y macOS mediante el enlace directo inalámbrico de Apple. En Actas del Simposio de Seguridad de USENIX. 37–54.
[46] Branden Ghena, Joshua Adkins, Longfei Shangguan, Kyle Jamieson, Philip Levis y Prabal Dutta. 2019. Desafío: Las LPWAN sin licencia aún no son el camino hacia la conectividad ubicua. En Actas de la Conferencia Internacional Anual de la ACM sobre Computación Móvil y Redes (MobiCom'19). 1–12.
[47] Tallal Elshabrawy y Joerg Robert. 2018. Aproximación de forma cerrada del rendimiento de la BER de la modulación LoRa. IEEE Carta Comun. (2018).
[48] Pascal Massart. 2007. Desigualdades de concentración y selección de modelos, Vol. 6. Springer.
[49] Qian Chen y Jiliang Wang. 2021. AlignTrack: Ampliando los límites de la decodificación de colisiones LoRa. IEEE 29.ª Conferencia Internacional sobre Protocolos de Red (ICNP'21), 1–11. DOI:10.1109/ICNP52444.2021.9651985 [50] Chenning Li,
Hanqing Guo, Shuai Tong, Xiao Zeng, Zhichao Cao, Mi Zhang, Qiben Yan, Li Xiao, Jiliang Wang y Yunhao Liu. 2021. NELoRa: Hacia una comunicación LoRa con SNR ultrabaja con demodulación mejorada neuronal. En
Actas de la 19ª Conferencia ACM sobre sistemas de sensores en red integrados, 56–68.
[51] Shuai Tong, Zilin Shen, Yunhao Liu y Jiliang Wang. 2021. Combatiendo la dinámica de enlaces para una conexión Lora confiable en entornos urbanos. En Actas de la 27.ª Conferencia Internacional Anual sobre Computación Móvil y Redes, 642–655.

[52] Jinyan Jiang, Zhenqiang Xu, Fan Dang y Jiliang Wang. 2021. Retrodispersión ambiental de LoRa de largo alcance con decodificación paralela. En Actas de la 27.ª Conferencia Internacional Anual sobre Computación Móvil y Redes, 684–696.
[53] Qianwen Miao, Fu Xiao, Haiping Huang, Lijuan Sun y Ruchuan Wang. 2019. Sistema de asistencia inteligente basado en un algoritmo de distribución de frecuencia con etiquetas RFID pasivas. Tsinghua Science and Technology 25, 2 (2019), 217–226.

Recibido el 24 de febrero de 2022; revisado el 19 de mayo de 2022; aceptado el 5 de junio de 2022