SyncWord Alteration Impact on Header Detection in LR-FHSS

Wilem Lamdani
Univ Rennes, Inria, CNRS, IRISA
Rennes, France
wilem.lamdani@irisa.fr

Bruno Sericola
Inria, Univ Rennes, CNRS, IRISA
Rennes, France
bruno.sericola@inria.fr

Gerardo Rubino
Inria, Univ Rennes, CNRS, IRISA
Rennes, France
gerardo.rubino@inria.fr

Soraya Ait Chellouche
Univ Rennes, Inria, CNRS, IRISA
Rennes, France
soraya.ait-chellouche@irisa.fr

Yassine Hadjadj-Aoul
Univ Rennes, Inria, CNRS, IRISA
Rennes, France
yassine.hadjadj-aoul@irisa.fr

César Viho
Univ Rennes, Inria, CNRS, IRISA
Rennes, France
cesar.viho@irisa.fr

***Abstract*—Long-Range Frequency-Hopping Spread Spectrum (LR-FHSS) was introduced as a new physical layer for Long Range Wide Area Network (LoRaWAN) to improve network capacity and robustness in highly congested deployments. LR-FHSS leverages pseudo-random frequency hopping to distribute the impact of interference or collisions, which, combined with packet fragmentation and forward error correction mechanisms, enables successful packet recovery even when parts of the transmission are lost. While most parts of an LR-FHSS frame benefit from redundancy and channel coding, the synchronization sequence (SyncWord) remains unprotected. As the main element used by the gateways to detect and synchronize with incoming transmissions, the SyncWord is a critical component of the reception process. Its successful detection is a prerequisite for header decoding and, consequently, for packet recovery. This paper investigates the robustness of the LR-FHSS SyncWord under potential alterations. We derive an analytical model of the SyncWord detection probability as a function of the number of corrupted bits and validate it through Monte Carlo simulations. The validated model is further integrated into a discrete-event LR-FHSS network simulator to assess its impact on network performance. The results show that successful SyncWord detection remains possible despite a limited number of bit errors, contradicting the common assumption that the synchronization sequence must be received without alteration. When integrated into network-level simulations, the proposed model enables the recovery of up to 5% additional packet headers, leading to a more accurate representation of LR-FHSS reception behavior and network performance.**

***Index Terms*—LR-FHSS; SyncWord; Header Detection Probability; Analytical Model; Performance Evaluation; Monte-Carlo Simulation.**

I. Introduction

The emergence of the Internet of Things (IoT) on a large scale has led to the development of new network technologies, known as Low-Power Wide-Area Network (LPWAN), capable of providing long-range connectivity while meeting the stringent power constraints of the massive number of battery-powered devices that need to be supported. Among these, LoRaWAN is probably the most prominent standard due to its flexibility, ease of deployment, and relatively low cost [1]. LoRa's traditional modulation, while effective in terms of range, has limitations in terms of network capacity when a very large number of devices need to transmit simultaneously, particularly in the context of satellite communications or large-scale deployments [2], [3].

To address these limitations, the LoRa Alliance introduced Long-Range Frequency-Hopping Spread Spectrum (LR-FHSS) in 2020 [4], a new modulation scheme designed to significantly improve network capacity while increasing its range when used with Low Earth Orbit (LEO) satellite gateways. LR-FHSS leverages Narrowband Gaussian Minimum Shift Keying (GMSK) modulation, pseudo-random frequency hopping, and robust channel coding to improve robustness against interference and collisions [5].

At the core of this, and more generally of modern Software-Defined Radio (SDR) communication architectures, the use of a synchronization sequence, known as SyncWord [6], is crucial. This well-known binary pattern serves as a critical reference point, enabling gateways to detect incoming transmissions, achieve frame synchronization and proceed with the decoding process.

A significant lack of evidence remains in the scientific literature regarding the robustness of this mechanism, which is generally overlooked or assumed to be ideal in analytical models or network simulators [7]. This work therefore addresses this research gap by analyzing the direct impact of SyncWord corruption on the probability of frame detection. In the LR-FHSS protocol, SyncWord corruption directly affects header detection. However, since the reception of payload fragments relies on the successful detection and decoding of at least one header replica, header detection failure results in the loss of the entire packet.

To the best of our knowledge, this paper presents the first quantitative analysis of the impact of SyncWord corruption on header detection in the LR-FHSS system and demonstrates that the common assumption of an "error-free SyncWord" is overly pessimistic. More specifically, it presents an analytical reliability model for the SyncWord detector that explicitly accounts for a given number of bit errors, derives closed-

979-8-3195-0167-7/26/$31.00 ©2026 IEEE

form expressions for the detection probability, and validates them using Monte Carlo simulations. This model is then integrated into a discrete-event LR-FHSS network simulator, where it enables the recovery of up to 5% more headers under realistic traffic conditions, leading to more accurate predictions of header detection probability and packet delivery rate than existing LR-FHSS performance evaluation models, which neglect the resilience of the SyncWord.

The remainder of this paper is organized as follows. Section II reviews LR-FHSS and discusses related work on performance evaluation and SyncWord robustness. Section III details the LR-FHSS frame and header structure, with a particular focus on the design and role of the SyncWord. Section IV introduces the considered SyncWord alteration models and presents a simplified bit-level detection abstraction. Section V derives the analytical detection probability model, which is validated through simulations in Section VI, which also discusses its impact on header detection and Packet Delivery Ratio (PDR). Finally, Section VII concludes the paper and summarizes the main findings and implications of the proposed SyncWord detection model.

II. Related Works And Motivation

Since its adoption by the LoRa Alliance to overcome the scalability limitations of the legacy CSS modulation, LR-FHSS has attracted significant attention from both academia and industry. Leveraging narrowband GMSK modulation, pseudo-random frequency hopping, and robust Forward Error Correction (FEC), LR-FHSS has been shown to significantly improve network scalability while maintaining long-ranges communication capabilities [5], [8]. These characteristics make it particularly well suited for massive IoT deployments and Direct-to-Satellite (DtS) communications.

Existing research on LR-FHSS can be broadly classified into two main categories: protocol-level enhancements and performance modeling.

Several works have proposed enhancements to the LR-FHSS protocol, including: new frequency hopping strategies in [9] and [10], adaptive payload coding and header replica allocation in [11], network-coded header replication in [12] and energy consumption analysis in [13],

In parallel, considerable efforts have been devoted to the analytical modeling of LR-FHSS. One of the first analytical studies was proposed by Ullah et al. [9], who investigated the packet delivery performance of LR-FHSS for DtS communications through both analysis and simulation. Maleki et al. [14] derived a closed-form outage probability model for satellite LR-FHSS links, accounting for fading and channel impairments. At the Medium Access Control (MAC) layer, de Souza Sant'Ana et al. [15] analyzed an asynchronous contention resolution-aided ALOHA scheme to improve channel utilization in dense LR-FHSS networks. More recently, in our previous work [16], we refined LR-FHSS performance modeling by deriving an exact ALOHA-based formulation that overcomes the inaccuracies of previous approximations, particularly under low traffic conditions. Dobler et al. [7] proposed a bit-level analytical model that accounts for partial header and payload corruption, adjacent-channel interference, and header redundancy optimization, providing more accurate estimates of network capacity and packet delivery ratio.

Despite these significant advances in modeling LR-FHSS, to the best of our knowledge, all existing analytical models rely on a strict SyncWord detection assumption. Indeed, in these models, a header replica is considered successfully detected only if its synchronization sequence is received without errors; otherwise, the entire header replica is assumed to be lost. the synchronization process itself was never explicitly modeled and its tolerance to bit errors is completely neglected. This assumption is particularly critical since, as discussed in the previous section, successful packet reception entirely depends on the successful detection and decoding of at least one header replica. Ignoring the resilience of the synchronization process may therefore lead to an underestimation of both header detection probability and packet delivery ratio. In this paper, we bridge this gap by deriving the first analytical model of the LR-FHSS SyncWord detection, explicitly accounting for bit errors during synchronization.

III. LR-FHSS

Long-Range Frequency-Hopping Spread Spectrum (LR-FHSS)) is a new LoRaWAN physical layer designed to significantly increase network capacity while preserving its long-range communication capabilities. Unlike conventional LoRa modulation, which transmits each packet over a single narrowband channel, LR-FHSS divides the transmission into multiple short fragments that are pseudo-randomly distributed over time and frequency across a wider bandwidth.

This frequency-hopping approach introduces both time and frequency diversity, which improves robustness to narrowband interference, fading, and packet collisions. As a result, even if some fragments are corrupted or lost, the receiver can still successfully reconstruct the original message using FEC and redundancy mechanisms. Thus, multiple transmissions coexist with limited mutual interference, making it particularly well suited for large-scale, dense IoT deployments.

To understand how LR-FHSS achieves reliable communications, it is first necessary to examine how a transmission is organized at the packet level.

A. Channel Partitioning

LR-FHSS operates within an Operating Channel Width (OCW), which is divided into multiple narrow subchannels. For the European Data Rate (DR) configurations considered in this work, namely DR8 and DR9, the OCW is 137 kHz. Within this channel, each transmitted fragment occupies an Occupied Bandwidth (OBW) of approximately 488 Hz and is sequentially transmitted over different subchannels.

At each transmission step, the transmitter occupies a single subchannel and then hops to another one according to a pseudo-random hopping sequence. This pattern ensures that packet fragments are distributed across both time and frequency domains.

This structure has two main consequences. First, it concentrates the transmitted energy into a narrow instantaneous bandwidth, which facilitates GMSK demodulation and maintains an approximately constant spectral efficiency, with one information bit per symbol. Second, by spreading fragments across multiple frequency resources, narrowband interferences or partial collisions only affect a subset of fragments rather than the entire packet.

For a given Data Rate configuration, the physical bit rate R_{phy} is determined by the symbol rate and the modulation scheme. Since binary GMSK encodes one bit per symbol, R_{phy} is approximately equal to the symbol rate of 488 symbols/s.

Consequently, the Time-on-Air (ToA) of a packet can be expressed as:

$$T_{\mathrm{ToA}} \approx \frac{N_{\mathrm{bits}}}{R_{\mathrm{phy}}}.$$

Where N_{bits} denotes the total number of encoded bits transmitted over the channel [6], [17].

B. Packet Formatting

In LR-FHSS, the transmitted information is explicitly divided: The *meta-data*, carried by the *headers*, and the *useful data* carried by the *payload*, respectively, each designed with different reliability and structural constraints.

From a transmission perspective, both header and payload undergo the following processing steps, but with different configurations:

Cyclic Redundancy Check: An error-detection mechanism is first implemented, by appending a Cyclic Redundancy Check (CRC) to both headers and the payload part. Allowing the receiver to verify the integrity of the recovered data after decoding.

Convolutional Encoding: The header and payload are then protected using convolutional encoding, a FEC technique that introduces controlled redundancy into the transmitted bitstream. Unlike block codes, convolutional codes generate parity bits based, not only on the current input bit, but also on a number of previously transmitted bits, providing the encoder with memory. Consequently, each encoded bit depends on the current input and the encoder's previous states.

This redundancy enables the receiver to recover the original information even when a portion of the transmitted bits has been corrupted or lost. At the receiver, the encoded sequence is decoded using the Viterbi algorithm [17], [18], a maximum-likelihood decoding algorithm that identifies the most probable transmitted bit sequence by exploring the trellis representation of the convolutional code.

The amount of redundancy depends on the selected Coding Rate (CR). Lower coding rates provide stronger protection at the expense of increased airtime, whereas higher coding rates improve spectral efficiency while reducing error-correction capabilities.

Interleaving: To further improve the FEC mechanisms, the processed bits are interleaved to distribute adjacent coded bits across different transmitted fragments. mitigating the impact of burst errors by converting them into single bit errors, that are more effectively handled by the convolutional decoder.

C. Payload Processing

Following CRC insertion, the payload is first whitened by XORing the payload bits with a pseudo-random sequence generated by a Linear Feedback Shift Register (LFSR). Data whitening randomizes the transmitted bitstream, preventing long sequences of identical bits and ensuring favorable spectral characteristics without providing confidentiality.

Subsequently, six zero-valued tail bits are appended to the whitened payload in order to terminate the convolutional encoder in the zero state, thereby improving the performance of the Viterbi decoder [19].

The resulting bitstream is then convolutionally encoded. Depending on the selected LR-FHSS data rate, the payload is encoded using either a CR of $1/3$ or $2/3$. The former provides stronger error protection through increased redundancy, while the latter offers higher spectral efficiency at the expense of reduced coding gain.

The payload is partitioned into 50-bit fragments, each composed of 48 information bits preceded by 2 guard bits set to zero, yielding a time on air of 103.4 ms.

Finally, the payload fragments are interleaved according to a pseudo-random permutation generated by an LFSR [17].

D. Header in-depth structure analysis

Since this work primarily focuses on the detection of LR-FHSS header, this section provides a detailed description of its internal structure and processing chain.

The header is composed of a fixed-length bit sequence that carries control and metadata information required for packet detection and decoding. It includes a CRC, the information bits, and a synchronization sequence (SyncWord).

Frame structure: The header consists of 40 bits of information, which are then encoded with a fixed CR of $1/2$.

Interleaving: Unlike the payload, header interleaving is deterministic rather than pseudo-random. Given the fixed header length (80 bits with redundancy), a predefined shuffling pattern is applied.

Synchronization Sequence: A synchronization sequence (SyncWord) is inserted within the header structure. This sequence enables frame detection and timing acquisition at the receiver, and serves as a reference for correct alignment of the subsequent decoding process. More details about the SyncWord will be discussed in Section III-E.

Replication: Due to the critical role of the header in packet recovery and demodulation, in addition to all the robustness mechanisms cited above, LR-FHSS relies on header replication, by sending 2 or 3 headers, extendable to 4, in order to maximize the chances of receiving a copy.

E. Synchronization sequence

The synchronization sequence, commonly referred to as the SyncWord, is a known binary pattern. In the LR-FHSS standard, it is inserted into the middle of the header to facilitate packet synchronization at the receiver. Since its value is known a priori, the receiver can detect its presence through correlation, allowing accurate frame detection and synchronization before the header is decoded. Consequently, the SyncWord constitutes the reference sequence on which the synchronization algorithm is based.

Position within the Header: The SyncWord is inserted after the channel encoding and interleaving of the header information bits. Rather than being placed at the beginning of the frame as a conventional preamble, it is located in the middle, separating the encoded bits into two equal parts of 40 encoded bits each. The transmitted header therefore consists of 2 guard bits, 40 encoded bits, followed by the 32-bit SyncWord, and finally the remaining 40 encoded bits, resulting in a 114-bit frame, with a time on air of 233 ms.

This central placement forms a midamble, enabling synchronization to be performed after the receiver has already acquired part of the packet. Such an arrangement is particularly well suited to SDR implementations, as synchronization can be refined using samples acquired before and after the SyncWord, improving robustness against timing and frequency offsets while also being less prone to total collisions. However, unlike a conventional preamble, a midamble does not allow synchronization prior to the reception of the header, requiring the receiver to buffer the incoming signal until the synchronization sequence has been detected.

Design and Value: The SyncWord is defined as a fixed 32-bit binary sequence specified by the LR-FHSS standard. Its value `0x2C0F7995` has been selected to exhibit favorable autocorrelation properties, as seen in Fig. 1, characterized by a sharp main correlation peak and low sidelobe levels. These properties maximize the probability of correct synchronization while minimizing false detections, caused by noise or interference.

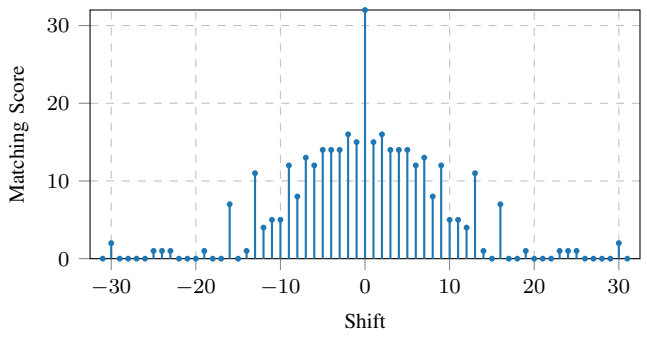


Fig. 1. Bit-matching score of the 32-bit LR-FHSS synchronization word (0x2C0F7995) as a function of the relative symbol shift. The score corresponds to the number of matching bits within the overlapping region.

F. Packet Decoding

Since the LR-FHSS gateway implementation is proprietary, unlike the transmitter-side drivers, the receiver design considered in this work is mainly based on the approach proposed in [5], which develop and evaluate an LR-FHSS receiver using real packet traces.

Channel Monitoring and Activity Detection: The LR-FHSS networks rely on an unslotted ALOHA access scheme, similar to conventional LoRaWAN networks. Consequently, the gateway continuously scans each physical channel to detect potential signal activity. The reception process does not rely on prior scheduling information, and packet detection must therefore be performed blindly based on energy and structure detection.

Channel activity is initially modeled as a sequence of energy peaks in the time–frequency domain, which is a burst of signal of approximately 50 ms. When multiple consecutive peaks (typically four to five) are observed on the same channel, the signal is flagged as a "candidate" header transmission and buffered for further processing. This stage provides a coarse estimation of the presence and temporal location of a header, and therefore a packet.

Finer Header Detection using the SyncWord: A finer synchronization stage is then performed using the known synchronization sequence (SyncWord). Unlike other parts of the packet, the SyncWord is known a priori at the receiver and is sent in clear with no encoding, which enables a correlation-based search over time and frequency offsets. This allows the gateway to refine the estimation of the header's start time, end time, and carrier frequency.

Accurate alignment with the SyncWord is a critical step, as successful header detection enables the full reconstruction of the structure. In practice, the SyncWord acts as an anchor for joint time–frequency synchronization.

Header Decoding and Packet Reconstruction: Once synchronization is achieved, the received signal is processed to reverse the transmitter operations. The header bits are first deinterleaved and then decoded using a Viterbi algorithm applied to the convolutionally encoded bitstream. The integrity of the decoded header is verified using the CRC.

If the header is successfully decoded, the receiver obtains all necessary information to reconstruct the associated payload fragments, including their time–frequency positions. The corresponding fragments are then collected and reassembled. Payload decoding is performed only if a sufficient number of non-corrupted bits are successfully received to allow correct Viterbi decoding.

A transmission is considered successful if at least one header replica is correctly decoded and sufficient payload information is available to complete the decoding process. Under these conditions, the entire packet can be reconstructed and verified using CRC.

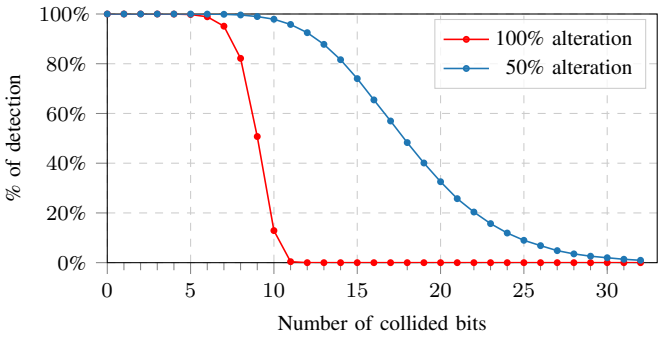


Fig. 2. Percentage of correct syncword alignments with respects to the number of collided bits, whether a collision has a 100% or 50% chance of changing the value of a bit.

IV. Syncword Alteration effect on Header Detection

The LR-FHSS synchronization sequence (SyncWord) is transmitted without additional protection mechanisms such as convolutional encoding or interleaving. As a consequence, it is directly exposed to channel impairments, including noise and inter-packet collisions, making it a particularly sensitive element of the header structure.

In practice, synchronization is achieved through a correlation process between the received signal and the known SyncWord [5], due to the nature of the SyncWord, the peak correlation not being aligned is very unlikely. Any alteration of the SyncWord due to channel noise reduces the correlation peak and may lead to incorrect alignment. Since reliable header detection and decoding relies on successful SyncWord alignment, its corruption directly impacts the overall packet reception probability.

A. Simplified Bit-Level Detection Model

To analyze this effect, a simplified bit-level Matching score is adopted in both the simulator and the model.

Rather than relying on continuous-time signal processing and inner product computations to find the peak correlation, SyncWord detection is approximated by direct bit comparisons over all 32-bit sliding windows extracted from the received header bitstream and the reference SyncWord. This includes all the header bits excluding the first two guard bits (forced to 0), resulting to a total of 112 bits.

Under this assumption, the correlation metric is equivalent to the number of matching bits between the received sequence and the reference SyncWord. The maximum correlation corresponds to the case where all 32 bits match (no alterations), while any bit alteration reduces the detection metric proportionally.

B. Statistical Assumption for Header Bits

For the purpose of a simplified analysis, we omit the 2 guard bits at the beginning of the frame, and the encoded header information (excluding the SyncWord) is modeled as a random bit stream. This assumption is justified by the combined effect of CRC appending, convolutional coding at rate $1/2$, and interleaving, which jointly decorrelate the transmitted bits.

Under this hypothesis, the header behaves as a white-noise-like process from the perspective of SyncWord detection, simplifying analytical modeling and simulation.

C. Impact Evaluation

To quantify the robustness of the synchronization process under increasing SyncWord degradation, we evaluate the impact of bit corruption through a simple Python simulation. Two impairment models are considered: (i) controlled bit-flip errors affecting a predefined number of bits within the 32-bit synchronization sequence, and (ii) a simple collision-like model in which each SyncWord bit has an independent probability of 50% of being flipped.

The resulting detection probabilities are presented in Fig. 2. As expected, the detection probability decreases as the number of altered bits increases. However, the results also show that successful detection remains possible even when the SyncWord is partially corrupted, provided that a sufficient number of bits still match the reference sequence.

This observation challenges the common assumption adopted in the literature that an error-free synchronization sequence must be received for successful detection. Instead, it suggests that the synchronization process exhibits an inherent tolerance to bit errors. These results therefore motivate the development of a probabilistic detection model capable of characterizing the likelihood of successful synchronization as a function of the SyncWord corruption level.

V. Analytical Detection Probability Model

A. Definitions

In this model, the SyncWord is composed of N bits (in the protocol, $N = 32$), and is denoted $V = (V_1, \ldots, V_N)$.

The candidate header roughly detected by the receiver $\mathcal{R}$ is represented as three parts' frame, in the order $[U|V|W]$, where U and W have M bits each, and $M > N$ (in the protocol, $M = 40$). So, the frame has $2M+N$ bits (the guard bits are omitted in our analysis). U and V parts contain the encoded header information's bits, that can then be seen as white noise.

Because of possible errors in the transmission, $\mathcal{R}$ receives a possibly modified frame denoted here $[U'|V'|W']$.

For convenience, the bits of $[U'|V'|W']$ are also denoted $B_1, B_2, \ldots, B_{2M+N}$. This means, for instance, that $V'_1 = B_{M+1}$, etc.

Our previous description of the frame means that at every position i corresponding to U' or W', B_i can be considered as a Binary random variable with parameter $1/2$, that is,

$$\Pr(B_i = 1) = \Pr(B_i = 0) = \frac{1}{2}.$$

Moreover, these B_i are independent of each other, and of anything else in the model.

The procedure is as follows: $\mathcal{R}$ has a copy of SyncWord V, and it continuously applies a sliding-window detector over

the received bit stream. It then compares it to the first N consecutive bits of the received frame $(U'_1, U'_2, \ldots, U'_N)$, then with word $(U'_2, U'_3, \ldots, U'_{N+1})$, etc. When comparing V with $(B_i, B_{i+1}, \ldots, B_{i+N-1})$, $\mathcal{R}$ counts how many bits are identical in both words, and stores this value in the variable S_i. Formally,

$$S_i = \sum_{h=1}^{N} 1(V_h = B_{i-1+h}).$$

Here, $1(P)$ represents the indicator function, it is equal to 1 if predicate P is true, to 0 if not.

Then, it selects $i^* = \text{argmax}_i S_i$ as the beginning of V'.

B. System Model

The described procedure defines $2M+1$ random variables, $S_1, S_2, \ldots, S_{2M+1}$. We class them in different types: we have variables such as S_1, described above, and variables having a part of white noise (including the case of none at all) and a part of the central V' word; for instance S_M (see below).

If $i^* = M+1$ the procedure works, and it fails otherwise. In the performance analysis of the procedure, our target is to evaluate the success probability.

In more detail, let us introduce the different types I_1 to I_5 of variables first.

- I_1) $S_1, S_2, \ldots, S_{M-N+1}$: only white noise in the frame (so, $M-N+1$ variables);
- I_2) $S_{M-N+2}, S_{M-N+3}, \ldots, S_M$: first part of the sliding window is white-noise, the rest is composed of the first bits of V' ($N-1$ variables);
- I_3) S_{M+1}: the only variable with no white noise part in the comparison;
- I_4) $S_{M+2}, S_{M+3}, \ldots, S_{M+N}$: as in I_2, but first the comparison is with last bits of V', then with white noise ($N-1$ variables);
- I_5) $S_{M+N+1}, S_{M+N+2}, \ldots, S_{2M+1}$: as type I_1.

So, we have $2M+1$ variables to consider. Let $\mathcal{I}$ be the set of indexes $\mathcal{I} = \{1, 2, \ldots 2M+1\}$.

The target is the evaluation of the probability of the following event E, that we can denote $R = \Pr(E)$, and call the "reliability" of the procedure:

$$E = \text{“for all } i \in \mathcal{I},\ i \neq M+1, \quad S_i < S_{M+1}\text{”}.$$

Event E can be called "the success of the procedure". We use $S_i < S_{M+1}$ and not $S_i \leq S_{M+1}$ because the first inequality is the standard good situation and because more than one maximum is very unlikely (as we will see).

The goal of this modeling process is to obtain a good approximation of R.

C. Evaluation

a) Analysis of the S_is: Let us look now at the different S_is, for all $i \in \mathcal{I}$.

Consider first S_1. $\mathcal{R}$ counts how many of the bits in its perfect copy of V are identical to those in the beginning of the received frame. In symbols,

$$S_1 = \sum_{j=1}^{N} 1\big(B_j = V_j\big).$$

Since we consider B_is as pure random bits, the r.v.s $1\big(B_j = V_j\big)$, $j = 1, \ldots, N$, are also random bits. So, S_1 is Binomial with values in $\{0, 1, \ldots, N\}$ and probability parameter $1/2$, which we denote $S_1 \sim \text{Binomial}(N, 1/2)$. What we saw for S_1 happens for any S_i in classes I_1 or I_5: S_i is Binomial$(N, 1/2)$. So,

$$\Pr(S_i = n) = \binom{N}{n} 2^{-N},$$

and $E(S_i) = N/2$.

The cases of classes I_2, I_3 and I_4 are more involved. Except for the single variable S_{M+1} in I_3, in all S_is in I_2 or I_4 there is a random noise part, followed by a comparison between the first bits of V' and the last bits of V in I_2, or between the last bits of V' and the first bits of V in I_4.

Consider type I_2. The $N-1$ variables in this class are the counters S_i where $i = M-N+1+h$, for $h = 1, \ldots, N-1$. They are sums of two terms: $S_{i,wn}$, a white noise area in the frame segment considered (the first $M-i+1 = N-h$ bits), and $S_{i,sync}$, a term coming from comparing the first bits of V' (this concerns $N-M+i-1 = h$ bits) to the last bits of V. So, $S_i = S_{i,wn} + S_{i,sync}$. $S_{i,wn}$'s law is just Binomial$(N-h, 1/2)$, and $S_{i,sync}$ is a word having h bits, that will be analyzed next. By definition of white noise, these two terms are independent of each other.

Let us consider now class I_4. Its $N-1$ variables are the counters S_i where $i = M+1+h$ for $h = 1, \ldots, N-1$. As for class I_2, they can be written $S_i = S_{i,wn} + S_{i,sync}$: the first term is a Binomial$(N-h, 1/2)$ r.v., and the second term is the number of identical bits between the first $N-h$ bits of V and the last $N-h$ bits of V'.

Let us resume here the situation. For classes I_1 and I_5, we have $2(M-N+1)$ r.v.s $\sim$ Binomial$(N, 1/2)$. To these, we add $2(N-1)$ more Binomial variables, but decomposed as 2 variables $\sim$ Binomial$(h, 1/2)$, for $h = 1, \ldots, N-1$. All these r.v.s are independent of anything else in the model. The remaining variables are the $S_{i,sync}$ elements for i in classes I_2 and I_4, plus S_{M+1}, that we analyze next.

b) Conditioning on the number of bits flipped in the sync word V': Before moving to the analysis of the sync elements, the $S_{i,sync}$s variables, plus the S_{M+1} counter, let us introduce here the possible errors in the transmission.

First of all, the white noise elements don't change their probabilistic nature (if bits are flipped in U or W, the result is again white noise). So, we must only take care of the sync elements in the model.

Let K be the number of bits arrived modified in V'. We have $0 \leq K \leq N$. We consider here the simplest error model, a Bernoulli process with parameter $p \in [0, 1]$. That is, if X_i

is the probability that bit i arrives modified, the X_is are i.i.d., Binary, with $\Pr(X_i = 1) = p$. Then, $K \sim$ Binomial(N, p). Assume that we know that $K = k$ (that is, in the sequel we will always condition on this event). Let Z be a subset of V' having size z. The probability that $\ell \leq z$ errors (flips) occur inside Z is then $\binom{z}{\ell}\binom{N-z}{k-\ell}/\binom{K}{k}$ (a Hypergeometric law).

A preliminary notation now. Let t be a positive integer satisfying $1 \leq t \leq N$. We are going to compare the first t bits of V (vector $(V_1, \ldots, V_t)$ to its last t bits (vector $(V_{N-t+1}, \ldots, V_N)$. Denote by $e(t)$ (resp. $d(t)$) the number of bits that are equal (resp. different) in both vectors, and let $E(t)$ and $D(t)$ be the respective areas in V'. Looking at the positions in the frame of the two classes I_2 and I_4, we can write the index in I_2 in the form $M - N + 1 + t$, $t = 1, 2, \ldots, N-1$, and for I_4, we use the representation $i = M + 1 + t$, again for $t = 1, 2, \ldots, N-1$.

From this point, and to underline the fact that we are conditioning on $K = k$, we will use notation $V^{(k)}$ instead of V'.

We calculate now the laws of the $S_{i,sync}$s r.v.s in the classes I_2 and I_4, conditioned to the event $K = k$, for $k = 0, 1, \ldots, N$, plus the trivial case of S_{M+1}. Concerning this last r.v., since there are k bit changes in $V^{(k)}$ with respect to V, we obviously have $S_{M+1} = N - k$.

Now, take some $t \in \{1, 2, \ldots, N-1\}$. Call "overlap" region, in the case of I_2, the first t bits of $V^{(k)}$. The score S_i for $i = M - N + 1 + t \in I_2$, can be written

$$S_i = X^{(t)} + O^{(t)},$$

where $X^{(t)}$ is a Binomial$(N - t, 1/2)$ r.v., and $O^{(t)}$ counts the number of equal bits between the last t bits of V and the first t ones in $V^{(k)}$. In other words, this is a new notation for $S_{i,sync}$ ($O^{(t)}$) and $S_{i,wn}$ ($S_{i,sync}$), whose aim is to underline the dependency on t: if $i = M - N + 1 + t \in I_4$, then we have a similar representation of the score S_i:

$$S_{i,sync} = X^{(N-t)} + O^{(N-t)}.$$

Here, $O^{(N-t)}$ is another instance of the previously defined variable, and $X^{(N-t)}$ is a Binomial$(t, 1/2)$ r.v.

So, the last element in the analysis is that of the law of $O^{(t)}$. Consider first the case of $O^{(t)} = 0$. To be able to observe this value, we need to have $e(t)$ flips in $E(t)$, no flip in $D(t)$, and then, the rest of the errors in the remaining part of $V^{(k)}$. So, in the same way of the Hypergeometric example above, we can write

$$\Pr(O^{(t)} = 0) = \binom{e(t)}{e(t)}\binom{d(t)}{0}\binom{N-t}{k-e(t)}/\binom{K}{k}.$$

Consider now the case of exactly 1 bit correct in the overlap region. Here we have two possibilities: either all bits in $E(t)$ are flipped but also one in $D(t)$, or $e(t) - 1$ bits in $E(t)$ are flipped and none in $D(t)$. We can then write

$$\Pr(O^{(t)} = 1) = \frac{\sum_{z=0}^{1}\binom{e(t)}{e(t)+z-1}\binom{d(t)}{z}\binom{N-t}{k-e(t)-2z+1}}{\binom{K}{k}}.$$

The same reasoning leads to the general expression

$$\Pr(O^{(t)} = x) = \frac{\sum_{z=0}^{x}\binom{e(t)}{e(t)+z-x}\binom{d(t)}{z}\binom{N-t}{k-e(t)-2z+x}}{\binom{K}{k}},$$

for $k = 0, 1, \ldots, N$, $1 \leq t \leq N-1$, $x = 0, 1, \ldots, t$. In the numerator of the law of $O^{(t)}$ we have Polyhypergeometric expressions. In our case, this means an extension to the Hypergeometric law introduced above, where instead of having a subset Z of V' we have two subsets, say Z_1 (size z_1) and Z_2 (size z_2), and where we want the probability of having ℓ_1 errors in Z_1 and ℓ_2 errors in Z_2, which is equal to $\binom{z_1}{\ell_1}\binom{z_2}{\ell_2}\binom{N-z_1-z_2}{k-\ell_1-\ell_2}/\binom{N}{k}$.

Now, a carefully analysis of the combinatorial formula giving $\Pr(O^{(t)} = x)$ shows that whenever the values of the different variables $(t, e(t), d(t), k, x$ and $z)$ make their combination impossible, the corresponding term in the sum at the numerator is equal to 0. For instance, it is the case if $x > k - e(t)$, or if $x > N - k - e(t)$, or if $z > d(t) - x$, etc.

c) The system's reliability: Recall that the system's reliability, that is, the probability that the procedure detects correctly position $M + 1$ in the received frame, is

$$R = \Pr\big(\text{ for } i \in \mathcal{I},\ i \neq M+1,\ \text{we have } S_i < N - k\ \big).$$

Now, it is clear that, even in our simple Bernoulli loss model, we have dependencies between the S_is, because of the $S_{i,sync}$ counters. In spite of that, we will show that assuming independence, we obtain a very accurate approximation $\widetilde{R}$ of R. That is, we will evaluate

$$\widetilde{R} = \Pi_{i \in \mathcal{I},\ i \neq M+1} \Pr(S_i < N - k).$$

Let us write $\widetilde{R} = R_a R_b$, with $R_a = \Pi_{i \in I_1 \cup I_5} \Pr(S_i < N-k)$ and $R_b = \Pi_{i \in I_2 \cup I_4} \Pr(S_i < N - k)$.

First, since we have $M - N + 1$ scores in each of the classes I_1 and I_5, and each score follows the law Binomial$(N, 1/2)$, we have

$$R_a = \left(2^{-N}\sum_{j=0}^{N-k-1}\binom{N}{j}\right)^{2(M-N+1)}.$$

In R_b, we have twice the sums $O^{(t)} + X^{(t)} = Y^{(t)}$, where $X^{(t)}$ is a Binomial$(N-t, 1/2)$, and $t = 1, 2, \ldots, N-1$. This means that we have to compute the convolution of the Binomial and the law of $O^{(t)}$ that we obtained explicitly, thus obtaining the distribution of $Y^{(t)}$. Then, we evaluate the probability

$$R_b = \Big(\Pi_{t=1}^{N-1} \Pr\big(Y^{(t)} < N - k\big)\Big)^2.$$

This completes our approximation of the reliability R.

VI. Results & Discussions

A. Experimental setup

1) Header Detection Probability: To quantify the impact of SyncWord corruption on header detection, we developped a simple python Monte Carlo simulator

To estimate the probability of correctly locating the synchronization sequence as a function of the number of altered SyncWord bits, 10^4 independent trials were performed for each corruption level.

The corrupted bits were modeled in two ways :

- The corrupted bit changes its value (100% alteration).
- The corrupted bit has a 50% chance to either preserve or change its value (50% alteration).

The resulting synchronization probabilities were subsequently incorporated into the LR-FHSS packet-level simulator to evaluate the impact of partial SyncWord corruption on the overall packet delivery ratio (see below).

2) Packet Delivery Ratio & Header Salvaging: A second simulation framework was developed to evaluate LR-FHSS performance under more realistic network dynamics.

The simulator extends LR-FHSS-Sim [20], a discrete-event Python Monte Carlo simulator based on the SimPy library.

The original model relies on a classical fragment-level collision abstraction, where any degree of corruption results in the loss of the entire fragment. However, to accurately capture the impact of FEC in LR-FHSS, partial corruption effects must be considered. Consequently, collisions are modeled at the bit level based on the temporal overlap between simultaneously transmitted fragments.

When a new fragment enters a channel, it is compared against all currently active fragments. For each pairwise interaction, the temporal overlap is computed from the intersection of their transmission intervals. Using the LR-FHSS time-on-air formula, this mapping enables conversion of overlap duration into the number of corrupted bits. The resulting bit-level collision information is then used as input to the convolutional decoding process, enabling a more accurate estimation of the packet delivery ratio.

For the header decoding process, two successive conditions are considered:

- **Header detection:** the gateway must correctly align with the SyncWord. Two detection strategies are considered: a *strict* criterion, where any corruption of the SyncWord leads to detection failure, and a *tolerant* criterion, where detection is performed probabilistically according to the SyncWord alignment model developed in this work.
- **Header decoding:** once detected, the header is considered successfully decoded if at least half of the information bits are correctly recovered (CR $1/2$).

Note that this simulator operates at the MAC layer abstraction level, with a single static LR-FHSS gateway. The physical layer is not explicitly modeled; the only channel impairments are represented through inter-packet collisions.

Note that for all Monte-Carlo experiments, statistical confidence intervals were computed in order to assess the reliability of the estimated metrics. In particular, confidence intervals up to the 99.99% level were evaluated. However, due to the large number of simulation runs and extensive averaging performed, the resulting confidence bounds were found to be extremely tight (negligibly small compared to the plotted values). Consequently, they are omitted from the figures for readability purposes.

B. Header Detection Probability

To validate the proposed SyncWord alignment model, we compare the analytical detection probability with the Monte Carlo simulation results under different corruption levels and assumptions (100% and 50% bit alteration).

The comparison is shown in Fig. 3, where the detection rates and probabilities are plotted against the number of corrupted SyncWord bits. The results show a strong agreement between the analytical model and the simulation, confirming the validity of the proposed formulation.

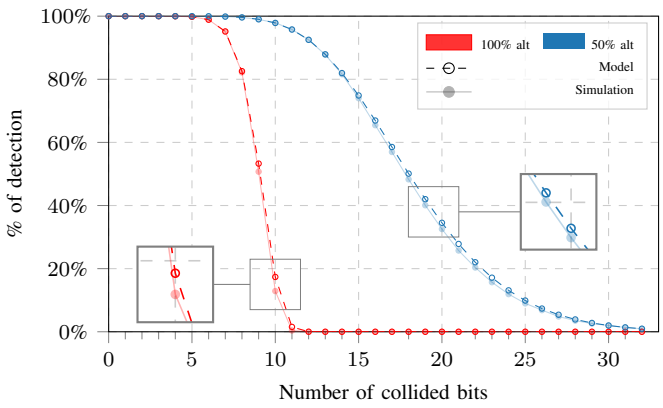


Fig. 3. Probabilty vs. Percentage of correct syncword alignments with respects to the number of collided bits, whether a collision has a 100% or 50% chance of changing the value of a bit.

C. Effect on PDR

The impact of the proposed syncword alignment model on system-level performance is evaluated in terms of PDR and salvaged headers. In this experiment, we compare the strict and tolerant header detection criteria introduced in Section V under varying traffic loads. Two LR-FHSS standard data rates (DR8 and DR9) are considered in order to assess the robustness of the proposed approach across different transmission configurations.

- **DR8:** The most robust European data rate, characterized by a coding rate of $1/3$ and three header replicas.
- **DR9:** A more lightweight configuration, with a coding rate of $2/3$ and two header replicas.

The first results are presented in Fig. 5, where the packet delivery ratio is plotted as a function of the packet generation rate. The results highlight a significant performance gap between the strict and tolerant detection criteria, particularly in the medium traffic regime where collisions and partial SyncWord corruption are both frequent.

Furthermore, the impact of the detection criterion is consistent across both data rates, although DR9 exhibits a steeper degradation due to its higher sensitivity to channel occupancy and collision density.

In addition to PDR, Fig. 4 reports the proportion of salvaged headers, defined as headers that would have been discarded under the strict criterion but are successfully recovered under the tolerant detection scheme. The results show that up to approximately 5% of headers can be recovered in moderate traffic conditions. This confirms the critical role of the SyncWord detection mechanism, since even a small improvement at the header level translates into a noticeable gain in overall network performance.

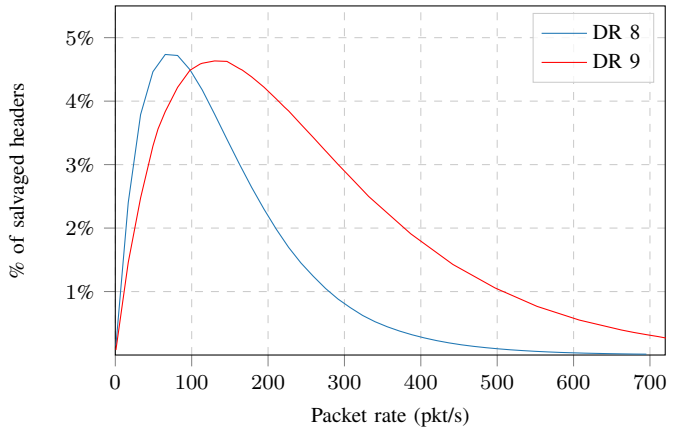


Fig. 4. Salvaged LR-FHSS header fragments (% of total transmitted headers) versus packet rate (packets/s) for different LR-FHSS configurations, with a payload size of 20 octets.

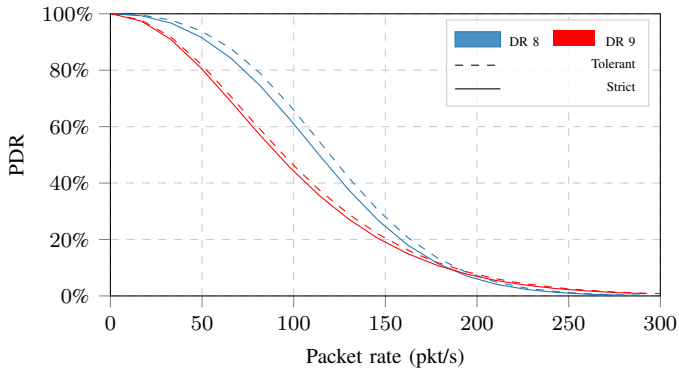


Fig. 5. Packet Delivery Ratio (PDR) obtained with a strict and a tolerant syncword alignment criterion based on the Header Detection Model proposed, under two different LR-FHSS Data Rates.

D. Single Header Configuration

To further assess the impact of the approach, we consider a more critical configuration; the case of a single header transmission combined with the most robust coding rate (CR $= 1/3$). In this setting, the header becomes a critical point of failure: any successful recovery of the header directly determines whether the entire packet can be decoded or not.

The results are presented in Fig. 6, which jointly reports the PDR and the proportion of salvaged headers under strict and tolerant detection criteria. As expected, both curves exhibit a performance degradation with increasing traffic load due to the rising collision probability.

However, a clear gap is observed between the strict and tolerant criteria. In the strict case, any corruption of the SyncWord immediately leads to header rejection, which results in a faster collapse of the PDR as soon as the network enters medium-load conditions. In contrast, the tolerant criterion partially mitigates this effect by allowing successful synchronization even under partial SyncWord corruption, thereby preserving a non-negligible fraction of transmissions.

This effect is further highlighted by the salvaged header metric. Although the absolute proportion of recovered headers remains limited (on the order of a few percent at peak), its impact on system performance is significant. Indeed, in the single-header configuration, recovering a header is equivalent to recovering the entire packet transmission, since no redundancy exists at the header level. Consequently, each salvaged header directly translates into a potential successfully decoded packet that would otherwise have been lost.

These results emphasize that, under highly constrained redundancy conditions, even marginal improvements in synchronization robustness can yield measurable gains in end-to-end performance.

VII. Conclusion

In this paper, we present the first in-depth quantitative analysis of the impact of Synchronization Sequence (SyncWord) corruption on header detection in LR-FHSS networks. While the existing literature often relies on the simplifying and pessimistic assumption that the SyncWord must be received completely free of errors to achieve synchronization, our work demonstrates tthat correct alignment can still be achieved in the presence of corrupted SyncWord bits. To support this observation, we developed a probabilistic analytical model that estimates the reliability of the SyncWord detection mechanism as a function of the number of corrupted bits. The analytical results were validated through bit-level simulations, showing close agreement with the proposed approximation. The integration of this detection abstraction into a discrete-event LR-FHSS network simulator further demonstrated that accounting for partially corrupted SyncWords can recover headers that would otherwise be discarded under the conventional error-free detection criterion, thereby providing a more refined evaluation of header detection and packet delivery performance.

The proposed approach intentionally relies on a bit-level abstraction of the synchronization process and does not explicitly model waveform-level effects such as noise, fading, capture effects, timing and frequency offsets, or receiver-specific detection thresholds. Future work will therefore focus on incorporating these physical-layer effects and investigating more realistic and correlated bit-error patterns as well as integrating the synchronization model into large-scale LR-

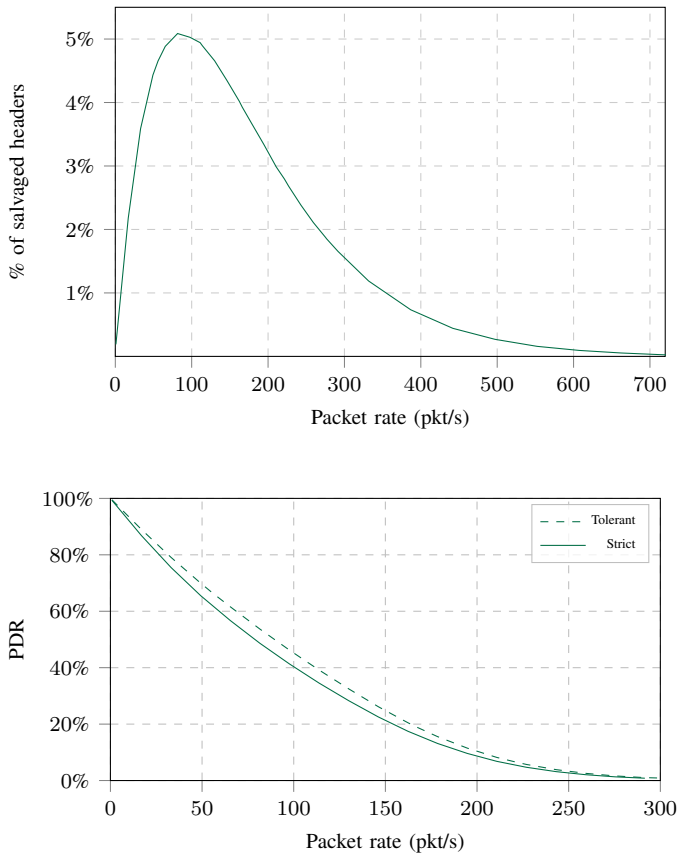


Fig. 6. Salvaged LR-FHSS header fragments (% of total transmitted headers) and Packet Delivery Ratio versus packet rate (packets/s) for a coding rate of 1/3 with only one header and a 20-octet payload.

FHSS network models, and exploring the impact of alternative synchronization sequences on detection robustness.

REFERENCES

[1] J. P. Shanmuga Sundaram, W. Du, and Z. Zhao, "A survey on lora networking: Research problems, current solutions, and open issues," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 371–388, 2020.

[2] D. Bankov, E. Khorov, and A. Lyakhov, "On the limits of lorawan channel access," in *2016 International Conference on Engineering and Telecommunication (EnT)*, pp. 10–14, 2016.

[3] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the limits of lorawan," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34–40, 2017.

[4] G. Boquet, P. Tuset-Peiró, F. Adelantado, T. Watteyne, and X. Vilajosana, "Lr-fhss: Overview and performance analysis," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 30–36, 2021.

[5] J. Bukhari and Z. Zhang, "Understanding long range-frequency hopping spread spectrum (lr-fhss) with real-world packet traces," *ACM Trans. Sen. Netw.*, vol. 20, Oct. 2024.

[6] LoRa Alliance, "Rp002-1.0.5 lorawan regional parameters," Oct. 2025. LoRaWAN Regional Parameters for different regulatory regions worldwide.

[7] F. Dobler, E. Bérat, A. Duda, and M. Heusse, "LR-FHSS: Analytical model of channel capacity and header redundancy optimization," *IEEE Internet of Things Journal*, vol. 12, no. 23, pp. 50156–50167, 2025.

[8] G. Boquet, P. Tuset-Peiró, F. Adelantado, T. Watteyne, and X. Vilajosana, "LR-FHSS: Overview and performance analysis," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 30–36, 2021-03.

[9] M. A. Ullah, K. Mikhaylov, and H. Alves, "Analysis and simulation of LoRaWAN LR-FHSS for direct-to-satellite scenario," *IEEE Wireless Communications Letters*, vol. 11, no. 3, pp. 548–552, 2022-03.

[10] F. Zhang, F. Yu, X. Zheng, L. Liu, and H. Ma, "DFH: Improving the reliability of LR-FHSS via dynamic frequency hopping," in *2023 IEEE 31st International Conference on Network Protocols (ICNP)*, pp. 1–12, 2023-10. ISSN: 2643-3303.

[11] J. d. A. Farhat, J. Michel de Souza Sant'Ana, J. L. Rebelatto, N. H. Mahmood, G. Pasolini, and R. D. Souza, "Probabilistic allocation of payload code rate and header copies in LR-FHSS networks," *IEEE Access*, vol. 13, pp. 30010–30020, 2025.

[12] D. N. Knop, J. L. Rebelatto, and R. D. Souza, "LR-FHSS with network-coded header replication," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 6, pp. 9066–9070, 2024-06.

[13] R. Sanchez-Vital, L. Casals, B. Heer-Salva, R. Vidal, C. Gomez, and E. Garcia-Villegas, "Energy performance of LR-FHSS: Analysis and evaluation," *Sensors*, vol. 24, no. 17, p. 5770, 2024-01. Number: 17 Publisher: Multidisciplinary Digital Publishing Institute.

[14] A. Maleki, H. H. Nguyen, and R. Barton, "Outage probability analysis of LR-FHSS in satellite IoT networks," *IEEE Communications Letters*, vol. 27, no. 3, pp. 946–950, 2023-03.

[15] J. M. de Souza Sant'Ana, O. d. S. Neto, A. Hoeller, J. L. Rebelatto, R. D. Souza, and H. Alves, "Asynchronous contention resolution-aided ALOHA in LR-FHSS networks," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 16684–16692, 2024-05.

[16] W. Lamdani, G. Rubino, S. A. Chellouche, Y. Hadjadj-Aoul, and C. Viho, "A finer aloha-based analytical model for lr-fhss performance evaluation," in *2025 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 01–07, 2025.

[17] Lora-net, "Swdm001: Long-range frequency hopping spread spectrum (lr-fhss) transmission demo." https://github.com/Lora-net/SWDM001.

[18] D. Maldonado, L. S. Cardoso, J. A. A. Fraire, A. Guitton, O. Iova, M. Kaneko, and H. Rivano, "Enhanced LR-FHSS receiver for headerless frame recovery in space–terrestrial integrated IoT networks," *Computer Networks*, vol. 257, p. 111018, 2025.

[19] EPFL Telecom Communications Lab, "Reverse engineering report," tech. rep., École Polytechnique Fédérale de Lausanne (EPFL), 2020.

[20] J. M. d. S. S. Ana, A. Hoeller Jr., H. Alves, and R. D. Souza, "LR-FHSS-sim: A discrete-event simulator for LR-FHSS networks," 2024-04-15.