



LR-FHSS : j'ai perdu l'entête

Juan Andrés A Fraire, Alexandre Guitton, Oana Iova

► To cite this version:

Juan Andrés A Fraire, Alexandre Guitton, Oana Iova. LR-FHSS : j'ai perdu l'entête. CoRes 2023 - 8èmes Rencontres Francophones sur la Conception de protocoles, l'évaluation de performances et l'expérimentation de Réseaux de communication, May 2023, Cargese, France. hal-04087040

HAL Id: hal-04087040

<https://hal.science/hal-04087040>

Submitted on 2 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

LR-FHSS : j'ai perdu l'entête

Juan A. Fraire¹, Alexandre Guitton^{1,2}, Oana Iova¹

¹Univ Lyon, INSA Lyon, Inria, CITI, F-69621 Villeurbanne, France

²Université Clermont-Auvergne, CNRS, Mines de Saint-Étienne, Clermont-Auvergne-INP, LIMOS, 63000 Clermont-Fd.

LR-FHSS (*Long-Range Frequency Hopping Spread Spectrum*) est une récente modulation conçue pour que des équipements fonctionnant sur piles puissent transmettre des données à des satellites à basse orbite. Pour décoder une trame, la passerelle située sur le satellite doit recevoir au moins une copie de l'entête, et une grande proportion des fragments du *payload*. Toutefois, les entêtes LR-FHSS, plus longs que les fragments, ont de grandes chances d'être perdus lorsque le nombre de transmissions concurrentes est élevé. Dans ce papier, nous proposons une heuristique pour récupérer de telles trames LR-FHSS qui sont reçues sans entête. Nous montrons par simulation que notre heuristique a un comportement proche de l'optimal, en comparant ses résultats à ceux obtenus *via* un modèle linéaire en nombre entiers. Nous montrons aussi que notre heuristique peut récupérer beaucoup de trames reçues sans entête, tout en générant peu de faux positifs.

1 Introduction

Long-Range Frequency Hopping Spread Spectrum (LR-FHSS) is a recent modulation from Semtech defined for LoRaWAN [LoR21] and designed to enable communication from Internet of Things (IoT) devices situated on the ground to low Earth orbit (LEO) satellites. To support dense deployments with massive concurrent communications from thousands of devices, LR-FHSS uses a GMSK (Gaussian minimum-shift keying) modulation with fast intra-frame frequency hopping, which significantly increases network capacity and robustness to interference.

The header of an LR-FHSS frame is sent multiple times (typically 2 or 3 times). Each repetition is called a replica. A replica is sent on a random channel corresponding to a physical sub-carrier with a 488 Hz bandwidth. Each replica contains information about the frequency hopping sequence used for the payload. The payload is divided into short fragments, and each fragment is sent on a random channel according to the sequence. The large number of possible sequences ($2^9 = 512$, since the sequence identification is coded on 9 bits), makes it unlikely to have two or more frames sharing the same hopping sequence[†].

An LR-FHSS frame is correctly received and decoded if at least one header replica is received, as well as a large proportion of the fragments (typically one-third or two-thirds), thanks to the built-in error correction mechanism [BTPA⁺21]. One main cause for LR-FHSS frame loss is the loss of all header replicas, as, without it, the gateway cannot reconstruct a full frame from the received fragments. Even if the replicas are repeated several times, their time on air (233 ms) is more than twice the time on air of fragments (102.4 ms), making them more prone to collisions. Indeed, Ullah et al. showed through simulations that the primary reason for frame loss in an LR-FHSS network is the loss of the headers [UMA22], which is also supported by Maleki et al. [MNBB22].

In this paper, we propose for the first time a method to recover headerless LR-FHSS frames by inferring the random sequence used by the fragments. To this end, we assume that it is possible to detect whether a channel is busy at a given time slot. From this knowledge, we can cycle through all the possible frequency hopping sequences until we find one that matches the occupancy of the corresponding channels over time. Section 2 contains an integer linear program (ILP) model to find the optimal set of sequences from an observation, followed by our proposal of a low-cost heuristic to recover headerless LR-FHSS frames. We show then in Section 3 how our solution for recovering headerless LR-FHSS frames significantly boosts the frame delivery rate of LR-FHSS.

[†]. More information about the specificity of the channels and the frequency hopping for each region can be found in [LoR21].

2 Recovering LR-FHSS headerless frames

In this section, we present the model and the assumptions we made, and we describe two solutions to recover the sequences used by headerless frames : an exact ILP solution and a heuristic.

2.1 Problem model and assumptions

We consider a time-slotted model with \mathcal{T} time slots, C channels, S pseudo-random sequences, \mathcal{F} transmitted frames, where each frame is divided into \mathcal{P} fragments. The variable S denotes the set of S random sequences, each representing the hopping pattern for a series of fragments. We define a two-dimensional boolean matrix M , such that $M[t][c] = 1$ if and only if one or more fragment transmissions are observed on channel c at time t . We assume no fragment losses due to weak signal reception, but we consider the possibility of collisions. We assume all headers are lost. Therefore, the matrix M is composed only of a series of fragment transmissions, where each frame was fragmented using a sequence from S . As a result, each frame transmission i is uniquely defined by a sequence $s_i \in S$, a starting time slot $t_i \in [1; \mathcal{T}]$, and a number of fragments $p_i = \mathcal{P}$. We coin T as the set of 3-tuple elements $\{(s_i, t_i, p_i)\}$ for each $i \in [1; \mathcal{F}]$, each 3-tuple defining a single frame transmission. For example, in Fig. 1, we can see an observed traffic matrix M with four LR-FHSS frames transmissions. Each frame was divided into $p_i = 4$ fragments using a frequency hopping sequence $s_i \in S$ and sent at different times t_i . The gateway saves all the detected fragments into the observed traffic matrix M without knowing which fragments belong to which frame.

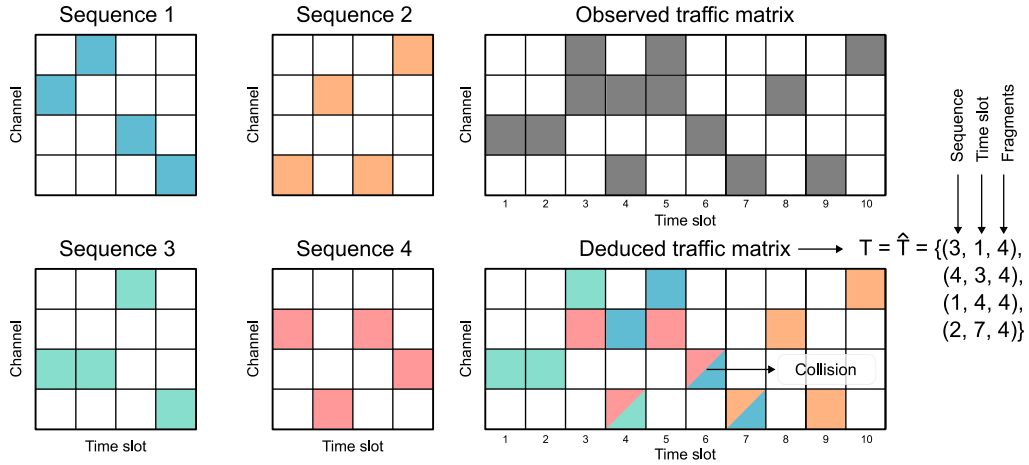


FIGURE 1 : Example of $S = 4$ sequences, an observed traffic matrix M , a deduced traffic matrix where each fragment is colored with the corresponding sequence, and a deduced traffic set $\hat{T} = \{(s_i, t_i, p_i)\}$.

The core problem of decoding headerless LR-FHSS transmissions is to correctly map the observed fragment receptions in M to the set of frame transmissions of T , that is, to deduce (s_i, t_i, p_i) for each frame i (see the deduced traffic matrix in Fig. 1). Thus, the goal is for the gateway to derive a frame transmission set \hat{T} as close as possible to the original T . In the ideal case, all frames can be recovered when $T = \hat{T}$.

To abstract the complexity of LR-FHSS headerless decoding, we make the following assumptions : a) the number of \mathcal{P} fragments per frame is known and fixed, b) fragments are synchronized in a slotted timeline, c) fragments collide if they are sent at the same time on the same channel (no capture effect considered), but they always reach the gateway (no loss due to low signal-to-noise ratio or sensitivity threshold), d) fragment collisions can be detected (but not recovered), and e) all transmissions start and end within the \mathcal{T} time slots).

2.2 ILP solution

We can formalize the headerless decoding problem for LR-FHSS as the following optimization problem. Let $y_{t,s} = 1$ if $(s, t, \mathcal{P}) \in T$, $\forall s, t$. Then :

$$\min. \sum_{t=1}^{\mathcal{T}} \sum_{s=1}^{\mathcal{S}} y_{t,s} \quad \text{s.t.} \quad \mathcal{P} \cdot y_{t,s} \geq \sum_{k=1}^{\mathcal{P}} M_{t+k-1,s[k]} \quad \forall t \in [1; \mathcal{T}], \forall s \in [1; \mathcal{S}].$$

The model aims at finding the minimum number of transmissions in \hat{T} needed to cover the matrix M . To this end, the objective function aims to minimize the sum of starting transmissions in $y_{t,s}$ (for all time slots t and sequences s), which is equal to the number of sequences in \hat{T} . However, the set of transmissions in \hat{T} must comply with a constraint that forces $M[t+k-1][s[k]] = 1$ for each k -th fragment of the frame. As observed in the equations above, a variable number of fragments \mathcal{P} would make the model non-linear. Because of this, we decided to keep \mathcal{P} as a constant for all frames in this work.

2.3 Sliding window heuristic

We introduce here a simple yet effective heuristic to find \hat{T} from the input matrix M . The algorithm iterates over each time slot t in the observed traffic matrix M . At each t , the algorithm tries to match the following \mathcal{P} fragments in the matrix M with one of the sequences $s \in \mathcal{S}$. If a match is found, the tuple (s, t, \mathcal{P}) is added to \hat{T} . This greedy heuristic has a time complexity of $O(\mathcal{T} \cdot \mathcal{S} \cdot \mathcal{P})$. Since $[1; \mathcal{T}]$ is explored in increasing time slots, this offline heuristic can be easily translated into an online heuristic with a sliding window of \mathcal{P} time slots. As a result, the heuristic presents a time and space complexity of $O(\mathcal{S} \cdot \mathcal{P})$ at each time slot, considering that $\mathcal{S} = 2^9$ and $\mathcal{P} \leq 113$ (bounded by the maximum payload size [LoR21]).

3 Results

To study the performance of both ILP and heuristic solutions, we conducted a simulation campaign using DR8 and DR9 data rates for Europe[‡]. We implemented a reference Python library that leverages an interface with Gurobi [Gur22] to solve the ILP model. The parameters used in the following analysis are the following : $C = 35$, $\mathcal{T} = 1000$, $\mathcal{S} = 512$, $\mathcal{F} \in [500; 3200]$, $\mathcal{P} \in [10; 90]$. For each configuration, we make 10 runs. We randomly generated a new set of sequences \mathcal{S} and a new set of frame transmissions T with uniform distribution for each run. We allow repeated transmissions in T (that is $(s_i, t_i, p_i) \in T$, $(s_j, t_j, p_j) \in T$ with $i \neq j$ and $s_i = s_j$, $t_i = t_j$ and $p_i = p_j$), but not identical sequences in \mathcal{S} .

For both algorithms we evaluate the following metrics : (1) True Positives (TP), which represents the number of frames from T that are found in \hat{T} , and (2) False Positives (FP), which represents the number of frames not existing in T that are found in \hat{T} . Note that none of the two algorithms produces False Negatives, which are frames from T that are not found in \hat{T} .

Figure 2(a) shows the number of TPs and FPs with a logarithmic scale, as a function of the number of frame transmissions and for a varying number of fragments. Note that both the ILP and the heuristic produce precisely the same number of TPs and FPs : thus, our heuristic has an optimal behavior for FPs. Both the ILP and the heuristic infer some transmissions that do not exist, thus creating FPs. This happens when several parts of sequences overlap such that a new sequence is formed. We note that FPs appear only when the matrix occupancy is loaded with traffic. We believe that most of these FPs can be removed through CRC checks (for instance, if there were one million FPs, a CRC of 16 bits would filter out these FPs except about 15). Finally, we can observe that the number of FPs increases with the number of fragments per frame, as a more significant number of fragments fills the matrix occupancy quicker.

Figure 2(b) shows the computation time as a function of the number of frame transmissions. As expected, the computation time of the ILP (dotted lines) snowballs with respect to the transmission count. However, the growth seems to slow down for a more significant number of fragments. This is because the matrix becomes nearly complete as the number of frame transmissions increases. Adding more transmissions past this point does not increase the computation time of the ILP. The computation time of the heuristic (solid

[‡]. DR8 has a coding rate of 1/3 and 3 header repetitions. DR9 has a coding rate of 2/3 and 2 header repetitions [LoR21].

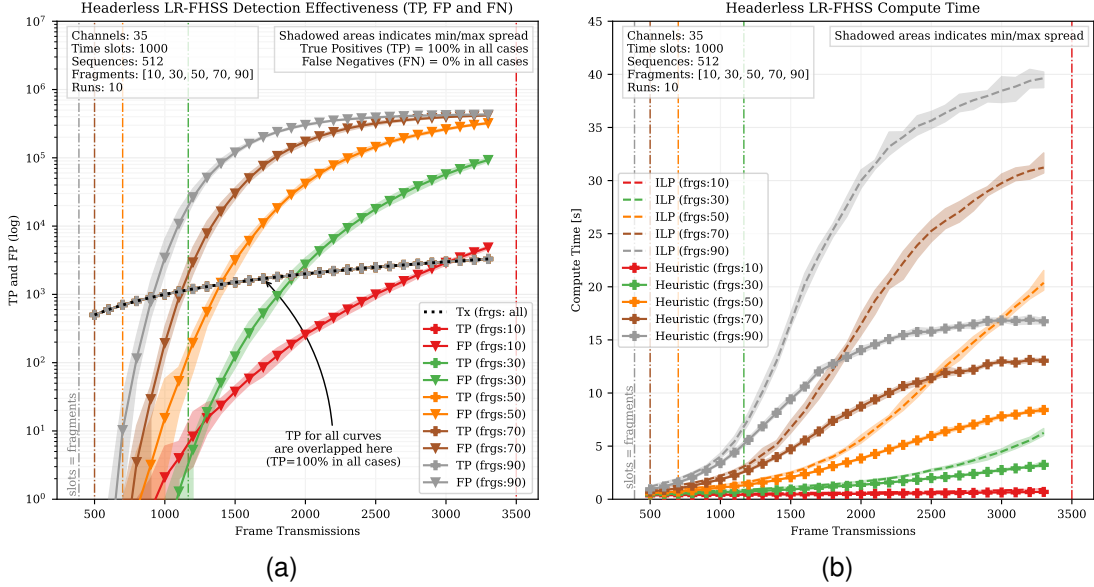


FIGURE 2 : (a) True Positives (TPs) and False Positives (FPs) for varying frame transmissions (both the ILP and the heuristic have identical performance). (b) Compute time for ILP model and the heuristic algorithm for varying frame transmissions. To give an intuition for channel occupation, the vertical lines represent the point when the total number of fragments exceeds the number of slots in M . Still, as some transmission slots will overlap, the matrix M is not completely full on the right side of the vertical line.

lines) is much shorter and does not exceed 20 seconds, even for a large \mathcal{P} . Note that the computation time presented for the heuristic is the time required for the whole simulation of $\mathcal{T} = 1000$ time slots. As described previously, this time can be reduced by considering an online approach.

4 Conclusion

LR-FHSS is a long-range low-power modulation gaining momentum due to its high scalability. However, the headers of LR-FHSS frames are not very robust, and the loss of a frame header causes the loss of the whole frame. In this paper, we proposed two algorithms to recover headerless LR-FHSS frames : the first solution is obtained through an ILP, and the second is an online heuristic. We show through simulations that both algorithms can successfully retrieve a significant number of frames. Moreover, they achieve the same performance, thus showing the advantage of the heuristic. In our future work we will weaken some of the assumptions that we made, and we will consider a more realistic simulation setup.

Références

- [BTPA⁺21] G. Boquet, P. Tuset-Peiró, F. Adelantado, T. Watteyne, and X. Vilajosana. LR-FHSS : Overview and performance analysis. *IEEE Communications Magazine*, 59(3) :30–36, 2021.
- [Gur22] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2022.
- [LoR21] LoRa Alliance Technical Committee Regional Parameters Workgroup. LoRawan regional parameters. Technical Report RP002-1.0.3, LoRa Alliance, 5 2021.
- [MNBB22] A. Maleki, H. H. Nguyen, E. Bedeer, and R. Barton. D2D-aided LoRaWAN LR-FHSS in direct-to-satellite IoT networks. *arXiv preprint arXiv :2212.04331*, 2022.
- [UMA22] M. A. Ullah, K. Mikhaylov, and H. Alves. Analysis and simulation of LoRaWAN LR-FHSS for direct-to-satellite scenario. *IEEE Wireless Communications Letters*, 11(3) :548–552, 2022.