A Code-specific Conservative Model for the Failure Rate of Bit-flipping Decoding of LDPC Codes with Cryptographic Applications

Paolo Santini[1,2], Alessandro Barenghi[3], Gerardo Pelosi[3], Marco Baldi[1], and Franco Chiaraluce[1]

[1] Università Politecnica delle Marche
[2] Florida Atlantic University
[3] Politecnico di Milano
p.santini@pm.univpm.it, {alessandro.barenghi, gerardo.pelosi}@polimi.it, {m.baldi, f.chiaraluce}@univpm.it

Abstract. Characterizing the decoding failure rate of iteratively decoded Low- and Moderate-Density Parity Check (LDPC/MDPC) codes is paramount to build cryptosystems based on them, able to achieve indistinguishability under adaptive chosen ciphertext attacks. In this paper, we provide a statistical worst-case analysis of our proposed iterative decoder obtained through a simple modification of the classic in-place bit-flipping decoder. This worst case analysis allows both to derive the worst-case behaviour of an LDPC/MDPC code picked among the family with the same length, rate and number of parity checks, and a code-specific bound on the decoding failure rate. The former result allows us to build a code-based cryptosystem enjoing the δ-correctness property required by IND-CCA2 constructions, while the latter result allows us to discard code instances which may have a decoding failure rate significantly different from the average one (i.e., representing weak keys), should they be picked during the key generation procedure.

Keywords: Bit-flipping decoding, cryptography, decoding failure rate, LDPC codes, MDPC codes, weak keys.

1 Introduction

Code based cryptosystems, pioneered by McEliece [16], are among the oldest public-key cryptosystems, and have survived a significant amount of cryptanalysis, remaining unbroken even for quantum-equipped adversaries [5]. This still holds true for both the original McEliece construction, and the one by Niederreiter [18], when both instantiated with Goppa codes, as they both rely on the same mathematical trapdoor, i.e., having the adversary solve the search version of the decoding problem for a general linear code, which was proven to be NP-Hard in [4].

The public-key of such schemes corresponds to an obfuscated representation of the underlying error correcting code (either the generator matrix for McEliece, or the parity-check matrix for Niederreiter), equipped with a decoding technique that can efficiently correct a non-trivial amount of errors. Since the obfuscated form of either the generator or the parity-check matrix should be indistinguishable from the one of a random code with the same length and dimension, both the original McEliece and Niederreiter proposals have public-key sizes which grow essentially quadratically in the error correction capacity of the code, on which the provided security level itself depends.

The large public-key size required in these cryptosystems has hindered their practical application in many scenarios. A concrete way of solving this problem is to employ codes described by matrices with a Quasi-Cyclic (QC) structure, which result in public-key sizes growing linearly in the code length. However, employing QC algebraic codes has proven to be a security issue, as the additional structure given by the quasi-cyclicity allows an attacker to deduce the underlying structure of the secret code [11]. By contrast, code families obtained from a random sparse parity-check matrix do not suffer from the same problem, and have lead to the successful proposal of Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) codes or Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) codes [3,17] as code families to build a secure and efficient instance of either the McEliece or the Niederreiter cryptosystem.

However, the efficient iterative algorithms used for decoding Low-Density Parity-Check (LDPC) and Moderate-Density Parity-Check (MDPC) codes are not bounded distance decoders, yielding a non-zero probability of obtaining a decoding failure, known as Decoding Failure Rate (DFR), which translates into a decryption failure rate for the corresponding code-based cryptosystems. The presence of a non-null DFR was shown to be exploitable by an active adversary, which has access to a decryption oracle (the typical scenario of a Chosen Ciphertext Attack (CCA)), to extract information on the secret QC-LDPC or QC-MDPC code [10, 13]. To reliably avoid such attacks, cryptosystem constructions providing indistinguishability under adaptive chosen ciphertext attack (IND-CCA2) guarantees, even when considering decoding failures, were analyzed in [6, 14]. In order for the IND-CCA2 security guarantees to hold, the constructions require that the average of the DFR over all the keypairs, which an adversary is able to induce crafting messages, is below a given threshold δ; a definition known as δ-*correctness* [14]. Such a threshold δ must be exponentially small in the security parameter of the scheme, in turn calling for requirements on the DFR of the underlying code that cannot be estimated via numerical simulations (e.g., DFR$\leq 2^{-128}$).

The impossibility of validating the DFR through numerical simulations has spurred significant efforts in modelling the behaviour of iterative decoders for QC-LDPC and QC-MDPC codes, with the goal of finding reliable tools to assess the DFR [2, 20, 21, 23, 25]. A subset of the aforementioned works consider a very small number of iterations of the decoder, providing code-specific exact bounds for the DFR [20, 21, 25]; however, employing such bounds to perform the code-parameter design results in impractically large public-key sizes. In [23], the authors adopt a completely different approach which extrapolates the DFR in the desired regime from numerical simulations performed with higher DFR values. This method assumes that the exponentially decreasing trend of the DFR holds as the code length is increased while keeping the rate constant. Such an assumption, however, does not rest on a theoretical basis. Finally, in [2] the authors characterize the DFR of a two-iteration out-of-place decoder, providing a closed-form method to derive an estimate of the average DFR over all the QC-LDPC codes with the same length, rate and density, under the assumption that the bit-flipping decisions taken during the first iteration are independent from each other. In the recent work [8], authors have highlighted an issue concerning possible *weak keys* of QC-LDPC and QC-MDPC code-based cryptosystems, i.e., keypairs obtained from codes having a DFR significantly lower than the average one.

Contributions. We provide an analysis of the DFR of an in-place iterative Bit Flipping (BF)-decoder for QC-LDPC and QC-MDPC codes acting on the estimated error locations in a randomized fashion for a fixed number of iterations. We provide a closed form statistical model for such a decoder, allowing us to derive a worst-case behaviour at each iteration, under clearly stated assumptions. We provide both an analysis of the DFR of the said decoder in the worst case scenario for the average QC-LDPC/QC-MDPC code, and we exploit the approach of [21] to derive a hard bound on the performance of the decoder on a given QC-LDPC/QC-MDPC code. While our analysis on the behavior of a QC-LDPC/QC-MDPC code allows us to match the requirements for a δ-correct cryptosystem [14], the hard bound we provide for the behavior of the decoder on a specific code allows us to discard *weak keys* during the key generation phase, solving any concern about the use of weak keys. We provide a confirmation of the effectiveness of our analysis by comparing its results with numerical simulations of the described in-place decoder.

2 Preliminaries

Throughout the paper, we will use uppercase (resp. lowercase) bold letters to denote matrices (resp. vectors). Given a matrix $\mathbf{A}$, its i-th row and j-th column are denoted as $\mathbf{A}_{i,:}$ and $\mathbf{A}_{:,j}$, respectively, while the entry on the i-th row, j-th column is denoted as $a_{i,j}$. Given a vector $\mathbf{a}$, its length is denoted as $|\mathbf{a}|$, while the i-th element is denoted as a_i, with $0 \leq i \leq |a|-1$; finally, the support (i.e., the set of positions of the asserted elements in a sequence) and the Hamming weight of $\mathbf{a}$ are denoted as $\mathrm{S}(\mathbf{a})$ and $\mathrm{w_H}(\mathbf{a})$, respectively. We will use $\mathcal{P}_n$, $n \geq 1$, to denote the set of $n!$ permutations of n elements, represented as a set of integers from 0 to $n-1$, while the notation $\pi \xleftarrow{\$} \mathcal{P}_n$ is employed to randomly and uniformly pick an element in $\mathcal{P}_n$, denoting the picked permutation of integers in $\{0\dots,n-1\}$ as π.

As far as the cryptoschemes are concerned, in the following we will make use of a QC-LDPC/QC-MDPC code $\mathcal{C}$, with length $n = n_0p$, dimension $k = (n_0-1)p$ and redundancy $r = n-k = p$. The private-key will coincide with the parity-check matrix $\mathbf{H} = [\mathbf{H}_0, \mathbf{H}_1, \cdots, \mathbf{H}_{n_0-1}] \in \mathbb{F}_2^{r\times n}$, where each $\mathbf{H}_i$, $0 \leq i \leq$

$n_0 - 1$ is a binary circulant matrix of size $p \times p$ and fixed Hamming weight v of each column/row. Therefore, $\mathbf{H}$ has constant column-weight v and constant row-weight $w = n_0 v$, and we say that $\mathbf{H}$ is (v, w)-*regular*.

When considering the case of the McEliece construction, the public-key may be chosen as the systematic generator matrix of the code. The plaintext is in the form $\mathbf{c} = \mathbf{mG} + \mathbf{e},\ \mathbf{c} \in \mathbb{F}_2^{1\times n}$, where $\mathbf{m} \in \mathbb{F}_2^{1\times n}$, $\mathbf{e} \in \mathbb{F}_2^{1\times n}$ and $\mathrm{w_H}(\mathbf{e}) = t$. The decryption algorithm takes as input the ciphertext $\mathbf{c}$ to compute the syndrome $\mathbf{s} = \mathbf{cH}^\top = \mathbf{eH}^\top,\ \mathbf{s} \in \mathbb{F}_2^{1\times r}$, and the private-key $\mathbf{H}$ to fed a syndrome decoding algorithm with both $\mathbf{s}$ and $\mathbf{H}$ and derive $\mathbf{e}$, from which the original message is recovered looking at the first k elements of $\mathbf{c} - \mathbf{e}$.

When the Niederreiter construction is considered, the public-key is defined as the systematic parity-check matrix of the code, obtained from the private-key as $\mathbf{M} = \mathbf{H}_0^{-1}\mathbf{H} \in \mathbb{F}_2^{r\times n}$. In this case, the message to be encrypted coincides with the error vector $\mathbf{e} \in \mathbb{F}_2^{1\times n}$, $\mathrm{w_H}(\mathbf{e}) = t$, while the encryption algorithm computes the ciphertext $\mathbf{c} = \mathbf{eM}^\top,\ \mathbf{c} \in \mathbb{F}_2^{1\times r}$ as a syndrome. The decryption algorithm takes as input the ciphertext $\mathbf{c}$ and the private-key $\mathbf{H}$ to compute a private-syndrome $\mathbf{s} = \mathbf{cH}_0^\top = \mathbf{eM}^\top\mathbf{H}_0^\top = \mathbf{eH}^\top(\mathbf{H}_0^\top)^{-1}\mathbf{H}_0^\top = \mathbf{eH}^\top$ and, subsequently, fed with it a syndrome decoding algorithm to derive the original message $\mathbf{e}$.

3 Randomized In-place Bit-flipping Decoder

In this section we describe a slightly modified version of the BF decoder originally proposed by Gallager in 1963 [12]. We focus on the *in-place* BF-decoder in which, at each bit evaluation, the decoder computes the number of unsatisfied parity-check equations in which the bit participates: when this number exceeds some threshold (which may be chosen according to different rules), then the bit is flipped and the syndrome is updated. Decoding proceeds until a null syndrome is obtained or a prefixed maximum number of iterations is reached.

The algorithm we analyze is reported in Algorithm 1. Inputs of the decoder are the binary parity-check matrix $\mathbf{H}$, the syndrome $\mathbf{s}$, the maximum number of iterations `itermax` and a vector $\mathbf{b}$ of length `itermax`, such that the i-th iteration uses b_i as threshold. The only difference with the classic in-place BF decoder is that the estimates on the error vector bits are processed in a random order, driven by a random permutation (generated at line 3). For this reason, we call this decoder Randomized In-Place Bit-Flipping (RIP-BF) decoder. Such a randomization, which is common to prevent side-channel analysis [1, 15] (and typically goes by the name of instruction shuffling in that context), is crucial in our analysis, since it allows us to derive a worst case analysis, as we describe in the following section.

3.1 Assessing Bit-flipping Probabilities

In this section we describe a statistical approach to model the behaviour of the RIP-BF decoder. We assume that the bit evaluations are independent and uncorrelated, and depend only on the number of the bits of $\hat{\mathbf{e}}$ which do not match

Algorithm 1: Randomized In-Place BF decoder

Input: $\mathbf{s} \in \mathbb{F}_2^r$: syndrome
$\mathbf{H} \in \mathbb{F}_2^{r \times n}$: private parity-check matrix
Output: $\hat{\mathbf{e}} \in \mathbb{F}_2^n$: recovered error value
$\mathbf{s} \in \mathbb{F}_2^r$: syndrome, null if error $\hat{\mathbf{e}} = \mathbf{e}$
Data: $\mathtt{itermax} \geq 1$: maximum number of (outer loop) iterations
$\mathbf{b} = [b_1, \ldots, b_{\mathtt{itermax}}], b_k \in \{\lceil \frac{v}{2} \rceil, \ldots, v\}$, $1 \leq k \leq \mathtt{itermax}$: flip thresholds

1 $\mathtt{iter} \leftarrow 0,\ \hat{\mathbf{e}} \leftarrow \mathbf{0}_n$
2 **while** $(\mathtt{iter} < \mathtt{itermax}) \wedge (\mathrm{w_H}(\mathbf{s}) > 0)$ **do**
3 $\quad \pi \xleftarrow{\$} \mathcal{P}_n$ // random permutation of size n
4 $\quad$ **foreach** $\hat{e}_j \in \pi(\hat{\mathbf{e}})$ **do**
5 $\quad\quad \mathtt{upc} \leftarrow 0$
6 $\quad\quad$ **for** $i \leftarrow 0$ **to** $r-1$ **do**
7 $\quad\quad\quad \mathtt{upc} \leftarrow \mathtt{upc} + (s_i \cdot h_{i,j})$
8 $\quad\quad$ **if** $\mathtt{upc} \geq b_{\mathtt{iter}}$ **then**
9 $\quad\quad\quad \hat{e}_j \leftarrow \hat{e}_j \oplus 1$ // estimated error vector update
10 $\quad\quad\quad$ **for** $i \leftarrow 0$ **to** $r-1$ **do**
11 $\quad\quad\quad\quad s_i \leftarrow s_i \oplus h_{i,j}$
12 $\quad \mathtt{iter} \leftarrow \mathtt{iter} + 1$ // update of the iterations counter
13 **return** $\{\mathbf{s}, \hat{\mathbf{e}}\}$

the ones of $\mathbf{e}$ at the beginning of the outer loop iterations. Such an assumption is captured by the following statement.

Consider the execution of steps in Algorithm 1 from the beginning of an outer loop iteration (line 3). For each position j, with $0 \leq j \leq n-1$, of the unknown error vector, $\mathbf{e}$, (or equivalently, for each column of the matrix $\mathbf{H}$) if the number of the unsatisfied parity-checks (upc) influenced by e_j exceeds the predefined threshold chosen for the current (outer loop) iteration, $\mathtt{b_{iter}}$, then the j-th position of the estimated error vector, $\hat{e}_j$, is flipped and the value of the syndrome is updated (lines 6-9). Denoting as

- $\mathrm{P}_{f|1} = \mathtt{Prob}\left((j\text{-}th\ \mathtt{upc}) \geq b_{\mathtt{iter}} \mid e_j = 1\right)$, the probability that the computation of the j-th upc yields an outcome greater or equal to the current threshold (thus, triggering a flip of $\hat{e}_j$) conditioned by the hyphotetical event of knowing that the actual j-th error bit is asserted, i.e., $e_j = 1$;
- $\mathrm{P}_{m|0} = \mathtt{Prob}\left((j\text{-}th\ \mathtt{upc}) < b_{\mathtt{iter}} \mid e_j = 0\right)$, the probability that the computation of the j-th upc yields an outcome less than the current threshold (thus, maintaining the bit $\hat{e}_j$ unchanged) conditioned by the hyphotetical event of knowing that the actual j-th error bit is null, i.e., $e_j = 0$.

In the following analyses, the statement below is assumed to hold.

Assumption 1 *Both* $\mathrm{P}_{f|1}$ *and* $\mathrm{P}_{m|0}$ *are not a function of the bit-position in the actual error vector (i.e., j, in the previous formulae), although both probabilities are a function of the total number* $\hat{t} = \mathrm{w_H}(\mathbf{e} \oplus \hat{\mathbf{e}})$ *of positions over which the* *unknown error* $\mathbf{e}$ *and the estimated error vector* $\hat{\mathbf{e}}$ *differ, at the beginning of the* j*-th inner loop iteration (line 5 in Algorithm 1).*

To derive closed formulae for both $\mathrm{P}_{f|1}$ and $\mathrm{P}_{m|0}$, we focus on QC-LDPC/QC-MDPC parity-check matrices as described in Section 2 with column weight v and row weight $w = n_0 v$ and observe that Algorithm 1 uses the columns of the parity-check matrix, for each outer loop iteration, in an order that is chosen with a uniformly random draw (line 3), while the computation performed at lines 6–7 is independent by the processing order of each cell of the selected column. According to this, in the following we "idelize" the structure of the parity check-matrix, assuming each row of $\mathbf{H}$ independent from the others and modeled as a sample of a uniform random variable, distributed over all possible sequences of n bits with weight w. More formally,

Assumption 2 *Let* $\mathbf{H}$ *be a* $r \times n$ *quasi-cyclic block-circulant* (v, w)*-regular parity-check matrix and let* $\mathbf{s}$ *be the* $1 \times r$ *syndrome corresponding to a* $1 \times n$ *error vector* $\mathbf{e}$ *that is modeled as a sample from a uniform random variable distributed over the elements in* $\mathbb{F}_2^{1\times n}$ *with weight* t*.*
We assume that each row $\mathbf{h}_{i,:}$*,* $0 \le i \le r-1$*, of the parity-check matrix* $\mathbf{H}$ *is well modeled as a sample from a uniform random variable distributed over the elements of* $\mathbb{F}_2^{1\times n}$ *with weight* w*.*

Lemma 1. *From Assumption 2, the probabilities that the* i*-th bit of the syndrome* $(0 \le i \le r-1)$ *is asserted knowing that the* z*-th bit of the error vector* $(0 \le z \le n-1)$ *is null or not, i.e.,* $\mathtt{Prob}\,(s_i = 1|e_z) = \mathtt{Prob}\,(\langle \mathbf{h}_{i,:}, \mathbf{e}\rangle = 1|e_z)$*,* $\langle \mathbf{h}_{i,:}, \mathbf{e}\rangle = \bigoplus_{j=0}^{n-1} h_{i,j} \cdot e_j$*, can be expressed for each bit position* z*,* $0 \le z \le n-1$*, of the error vector as follows:*

$$\rho_{0,\mathtt{u}} = \mathtt{Prob}\,(\langle \mathbf{h}_{i,:}, \mathbf{e}\rangle = 1 \mid e_z = 0) = \frac{\sum_{l=0,\ l\ odd}^{\min\{w,t\}} \binom{w}{l}\binom{n-w}{t-l}}{\binom{n-1}{t}}$$

$$\rho_{1,\mathtt{u}} = \mathtt{Prob}\,(\langle \mathbf{h}_{i,:}, \mathbf{e}\rangle = 1 \mid e_z = 1) = \frac{\sum_{l=0,\ l\ even}^{\min\{w-1,t-1\}} \binom{w-1}{l}\binom{n-w}{t-1-l}}{\binom{n-1}{t-1}}$$

Consequentially, the probability that Algorithm 1 performs a bit-flip of an element of the estimated error vector, $\hat{e}_z$*, when the corresponding bit of the actual error vector is asserted,* $e_z = 1$*, i.e.,* $\mathrm{P}_{f|1}$*, and the probability that Algorithm 1 maintains the value of the estimated error vector,* $\hat{e}_z$*, when the corresponding bit of the actual error vector is null,* $e_z = 0$*, i.e.,* $\mathrm{P}_{m|0}$*, are:*

$$\mathrm{P}_{f|1} = \sum_{\mathtt{upc}=b}^{v} \binom{v}{\mathtt{upc}} \rho_{1,\mathtt{u}}^{\mathtt{upc}} (1-\rho_{1,\mathtt{u}})^{v-\mathtt{upc}},$$

$$\mathrm{P}_{m|0} = \sum_{\mathtt{upc}=0}^{b-1} \binom{v}{\mathtt{upc}} \rho_{0,\mathtt{u}}^{\mathtt{upc}} (1-\rho_{0,\mathtt{u}})^{v-\mathtt{upc}}.$$

Proof. Provided in Appendix C.

3.2 Bounding Bit-flipping Probabilities for a Given Code

Given a QC-LDPC code $\mathcal{C}$ with its $r \times n$ (v, w)-regular parity-check matrix $\mathbf{H}$, let us consider each column of $\mathbf{H}$, $\mathbf{h}_{:,z}$, $0 \le z \le n-1$, as a Boolean vector equipped with element-wise addition and multiplication denoted as $\oplus$ and $\wedge$, respectively. Let $\mathbf{\Gamma}$ be the $n \times n$ integer matrix, where each element $\gamma_{x,y} \in \{0, \dots, v\}$, with $0 \le x, y \le n-1$, is computed as the weight of the element-wise multiplication between two different columns, and 0 otherwise, i.e.,

$$\gamma_{x,y} = \begin{cases} \mathrm{w_H}\left(\mathbf{h}_{:,x} \wedge \mathbf{h}_{:,y}\right) & x \ne y \\ 0 & x = y \end{cases}$$

The integer matrix $\mathbf{\Gamma}$ is symmetric and, when derived from a block-circulant matrix, is made of circulant blocks, as well.

An alternate way of exhibiting the probability $\mathrm{P}_{f|1}$ that Algorithm 1 performs a bit-flip of an element of the estimated error vector, $\hat{e}_z$, when the corresponding bit of the actual error vector is asserted, i.e., $e_z = 1$, consists in counting how many of the $\binom{n-1}{t-1}$ error vectors $\mathbf{e}$, with $e_z = 1$, are such that the z-th upc counter computed employing the corresponding syndrome (see lines 6–7) is above the pre-defined threshold b:

$$\mathrm{P}_{f|1} = \frac{|\{\mathbf{e} \text{ s.t. } (z\text{-th } \texttt{upc}) \ge b\}|}{\binom{n-1}{t-1}}. \tag{1}$$

Noting that the computation of z-th $\texttt{upc}$ can be derived as a function of the unknown error vector $\mathbf{e}$ as follows:

$$z\text{-th } \texttt{upc} = v - \mathrm{w_H}\left(\bigoplus_{j \in \{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}} (\mathbf{h}_{:,z} \wedge \mathbf{h}_{:,j})\right) \ge v - \sum_{j \in \{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}} \gamma_{z,j},$$

the following inequality concerning the numerator of the fraction in Eq. (1) holds:

$$|\{\mathbf{e} \text{ s.t. } (z\text{-th } \texttt{upc}) \ge b\}| \ge \left| \left\{ \mathbf{e} \text{ s.t. } \left(v - \sum_{j \in \{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}} \gamma_{z,j} \right) \ge b \right\} \right|$$

The cardinality of the set on the right-hand side of the above inequality asks for the counting of all error vectors such that the sum of the elements on the z-th row of the matrix $\mathbf{\Gamma}$ indexed by the positions in $\{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}$ (with $|\{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}| = t-1$) is less than $v - b$: i.e., $\sum_{j \in \{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}} \gamma_{z,j} \le v - b$. The answer to the said question is equivalent to counting the number of solutions of the corresponding *subset sum* problem [7], that is finding a subset of $|\{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}| = t-1$ elements out of the ones in the row $\gamma_{z,:}$ adding up to at most $v - b$. A straightforward computation of such a counting is unfeasible for cryptographic relevant values of the involved parameters, exhibiting an exponential complexity in the correction capacity of the code t.

However, observing that, for QC-LDPC codes, the number η_z of unique values on each row $\gamma_{z,:}$ of the matrix $\mathbf{\Gamma}$ is far lower than t, therefore we designed an algorithm computing the same result with a complexity exponential in η_z, reported in Appendix B. In the following, for the sake of conciseness, the outcome of the said algorithm fed with a row of the matrix $\mathbf{\Gamma}$, the cardinality $|\{\mathrm{S}(\mathbf{e}) \setminus \{z\}\}| = t-1$ (i.e., the number of terms of the summation), and the `threshold` value that the sum must honor is denoted as: $\mathcal{N}(\gamma_{z,:}, t-1, \texttt{threshold})$.

$$\mathrm{P}_{f|1} \geq \frac{\max_{0 \leq z \leq n-1} \{\mathcal{N}(\gamma_{z,:}, t-1, v-b)\}}{\binom{n-1}{t-1}}. \quad (2)$$

With similar arguments, a lower bound on $\mathrm{P}_{m|0}$ can be derived, obtaining:

$$\mathrm{P}_{m|0} \geq \frac{\max_{0 \leq z \leq n-1} \{\mathcal{N}(\gamma_{z,:}, t, b-1)\}}{\binom{n-1}{t}}. \quad (3)$$

4 Modeling the DFR of the RIP-decoder

Using the probabilities, $\mathrm{P}_{f|1}, \mathrm{P}_{m|0}$, that we have derived in the previous section, under Assumption 1 we can derive a statistical model for the RIP-BF decoder. To this end, we now focus on a single iteration of the outer loop of Algorithm 1. In particular, as we describe next, we consider a *worst-case* evolution for the decoder, by assuming that, at each iteration of the inner loop, it evolves through a path that ends in the a decoding success with the lowest probability. We obtain a decoding success if the decoder terminates the inner loop iteration in the state where the estimate of the error $\hat{\mathbf{e}}$ matches the actual error $\mathbf{e}$. Indeed, in such a case, we have $\mathrm{w_H}(\mathbf{e} \oplus \hat{\mathbf{e}}) = 0$.

Let $\bar{\mathbf{e}}$ be the error estimate at the beginning of the outer loop of Algorithm 1 (line 3), and $\hat{\mathbf{e}}$ be the error estimate at the beginning of the inner loop of the same algorithm (line 5). In other words, $\bar{\mathbf{e}}$ is a snapshot of the error estimate made by the RIP decoder before a sweep of n estimated error bit evaluations is made, while $\hat{\mathbf{e}}$ is the value of the estimated error vector before each estimated error bit is evaluated.

Let $\hat{t}$ denote the number of residual erroneous bit estimations at the beginning of the inner loop iteration, that is $\hat{t} = \mathrm{w_H}(\mathbf{e} \oplus \hat{\mathbf{e}})$. From now on, we highlight the dependency of $\mathrm{P}_{f|1}$ and $\mathrm{P}_{m|0}$ from the current value of $\hat{t}$, writing them down as $\mathrm{P}_{f|1}(\hat{t})$ and $\mathrm{P}_{m|0}(\hat{t})$.

We denote as π the permutation picked in line 3 of Algorithm 1. Let $\mathcal{P}_n^*$ be the set of all permutations $\pi^* \in \mathcal{P}_n^*$ such that

$$\mathrm{S}(\pi^*(\mathbf{e}) \oplus \pi^*(\bar{\mathbf{e}})) = \{n-\hat{t}, n-\hat{t}+1, \cdots, n-1\}, \ \ \forall \pi^* \in \mathcal{P}_n^*.$$

Let $\texttt{Prob}(\hat{\mathbf{e}} \neq \mathbf{e} |\ \pi \in \mathcal{P}_n)$ be the probability that the estimated error vector $\hat{\mathbf{e}}$ at the end of the current inner loop iteration is different from $\mathbf{e}$, conditioned by the fact that the permutation π was applied at the beginning of the outer loop. Similarly, we define $\texttt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} \middle|\ \pi^* \in \mathcal{P}_n^*\right)$. Note that it can be verified that $\mathrm{P}_{f|1}(\hat{t}) \geq \mathrm{P}_{f|1}(\hat{t}+1), \mathrm{P}_{m|0}(\hat{t}) \geq \mathrm{P}_{m|0}(\hat{t}+1),\ \forall \hat{t}$, as increasing the number of current mis-estimated error bits, increases the likelihood of a wrong decoder decision. By leveraging the assumption made in the previous section, we now prove that the decoder reaches a correct decoding at the end of the outer loop with the least probability each time a $\pi^* \in \mathcal{P}_n^*$ is applied at the beginning of the outer loop.

Lemma 2. *The execution path of the inner loop in Algorithm 1 yielding the worst possible decoder success rate is the one taking place when $\pi^* \in \mathcal{P}_n^*$ is applied at the beginning of the outer loop, that is:*

$$\forall \pi \in \mathcal{P}_n, \forall \pi^* \in \mathcal{P}_n^*,\quad \texttt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} \middle|\ \pi \in \mathcal{P}_n\right) \leq \texttt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} \middle|\ \pi^* \in \mathcal{P}_n^*\right).$$

Proof. See Appendix D.

From now on we will assume that, in each iteration, a permutation from the set $\mathcal{P}_n^*$ is picked; in other words, we are assuming that the decoder is always constrained to reach a decoding success through the worst possible execution path. Let us define the following two sets: $E_1 = S(\mathbf{e})$, and $E_0 = \{0, \ldots, n-1\} \setminus S(\mathbf{e})$. Denote with $\hat{t}_0 = |\{S(\mathbf{e} \oplus \bar{\mathbf{e}}) \cap E_0\}|$, that is the number of places where the estimated error at the beginning of the outer loop iteration $\bar{\mathbf{e}}$ differs from the actual $\mathbf{e}$, in positions included in E_0. Analogously, define $\hat{t}_1 = |\{S(\mathbf{e} \oplus \bar{\mathbf{e}}) \cap E_1\}|$. Furthermore, let

i) $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$ denote the probability that the decoder in Algorithm 1, starting from a state where $\mathrm{w_H}\left(\hat{\mathbf{e}} \oplus \mathbf{e}\right) = \omega$, and acting in the order specified by a worst case permutation $\pi^* \in \mathcal{P}_n^*$ ends in a state with $\hat{t}_0 = x$ after completing the inner loop at lines $4-11$;
ii) $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$ denote the probability that the decoder in Algorithm 1, starting from a state where $\mathrm{w_H}\left(\hat{\mathbf{e}} \oplus \mathbf{e}\right) = \omega$, and acting in the order specified by a worst case permutation $\pi^* \in \mathcal{P}_n^*$ ends in a state with $\hat{t}_1 = x$ residual errors among the bits indexed by E_1 after completing the loop at lines 4–11;
iii) $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \underset{i}{\rightarrow} x\right)$ as the probability that, starting from a state such that $\mathrm{w_H}\left(\hat{\mathbf{e}} \oplus \mathbf{e}\right) = \omega$, after i iterations the outer loop at lines 2–12 of Algorithm 1, each one operating with a worst case permutation, ends in a state where $\mathrm{w_H}\left(\hat{\mathbf{e}} \oplus \mathbf{e}\right) = x$.

The expressions of the probabilities *i)* and *ii)* are derived in Appendix A, and only depend on the probabilities $\mathrm{P}_{f|1}(\hat{t})$ and $\mathrm{P}_{m|0}(\hat{t})$.

We now describe how the aforementioned probabilities can be used to express the worst case DFR after $\texttt{itermax}$ iterations, which we denote as $\mathrm{DFR}^*_{\texttt{itermax}}$. First of all, we straightforwardly have

$$\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \underset{1}{\rightarrow} x\right) = \sum_{\delta=\max\{0\ ;\ x-(n-\omega)\}}^{t} \texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x-\delta\right) \texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} \delta\right). \tag{4}$$

We can denote as $\hat{t}^{(i)} = \mathrm{w_H}\left(\mathbf{e} \oplus \hat{\mathbf{e}}^{(\mathtt{iter})}\right)$, that is: $\hat{t}^{(i)}$ corresponds to the number of residual errors after the i-th outer loop iteration. Then, by considering all possible configurations of such values, and taking into account that the first iteration begins with t residual errors, we have

$$\mathtt{Prob}_{\mathcal{P}_n^*}\left(t \xrightarrow[\mathtt{itermax}-1]{} \hat{t}^{(\mathtt{itermax}-1)}\right) = \sum_{\hat{t}^{(0)}=0}^{n} \cdots \sum_{\hat{t}^{(\mathtt{itermax}-2)}=0}^{n} \mathtt{Prob}_{\mathcal{P}_n^*}\left(\hat{t}^{(\mathtt{i_{max}}-2)} \xrightarrow[1]{} \hat{t}^{(\mathtt{itermax}-1)}\right) \prod_{j=0}^{\mathtt{itermax}-2} \mathtt{Prob}_{\mathcal{P}_n^*}\left(\hat{t}^{(j-1)} \xrightarrow[1]{} \hat{t}^{(j)}\right), \quad (5)$$

where, to have a consistent notation, we consider $\hat{t}^{(-1)} = t$. The above formula is very simple and, essentially, takes into account all possible transitions starting from an initial number of residual errors equal to t and ending in x residual errors. Taking this probability into account, the DFR after `itermax` iterations is straightforwardly obtained as

$$\mathrm{DFR}^*_{\mathtt{itermax}} = 1 - \sum_{\hat{t}^{(\mathtt{itermax}-1)}=0}^{n} \mathtt{Prob}_{\mathcal{P}_n^*}\left(t \xrightarrow[\mathtt{itermax}-1]{} \hat{t}^{(\mathtt{itermax}-1)}\right) \mathtt{Prob}_{\mathcal{P}_n^*}\left(\hat{t}^{(\mathtt{itermax}-1)} \xrightarrow[1]{} 0\right). \quad (6)$$

4.1 Analyzing a Single-iteration Decoder

For the case of the decoder performing just one iteration, the simple expression of the DFR has been derived in the proof of Lemma 2, that is

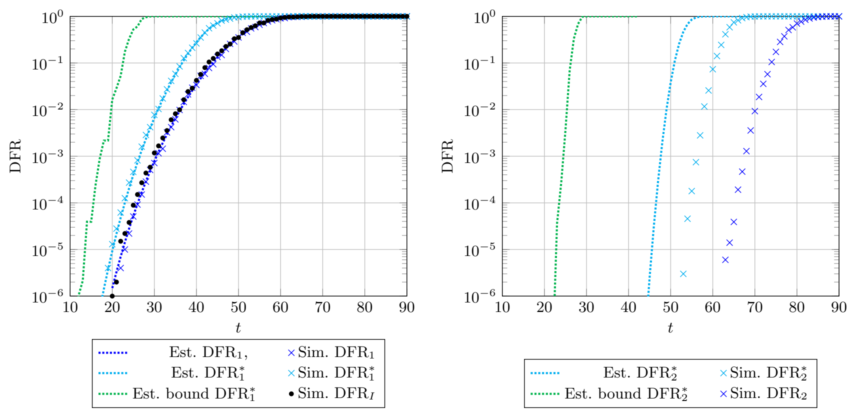
$$\mathrm{DFR}^*_1 = 1 - \mathtt{Prob}_{\mathcal{P}_n^*}\left(t \xrightarrow[1]{} 0\right) = \left(\mathrm{P}_{m|0}(t)\right)^{n-t} \prod_{j=1}^{t} \mathrm{P}_{f|1}(j).$$

Actually, for just one iteration, the average DFR (corresponding to the use of a random permutation π) can be approximated in a very simple way, as follows. Let a_i, a_{i+1}, with $i \in [0; t-2]$, be two consecutive elements of $\mathrm{S}\left(\pi(\mathbf{e})\right)$. Then denote with d the average zero-run lenght in $\mathbf{e}$, $d = \mathbb{E}\left[a_{i+1} - a_i\right] = \frac{n-t}{t+1}, \ \forall i \in [0; t-2]$ where $\mathbb{E}[\cdot]$ denotes the expected value. Consequently, we can write

$$\mathrm{DFR}_1 \approx 1 - \left(\prod_{j=1}^{t} \left(\mathrm{P}_{m|0}(j)\right)^{d}\right) \prod_{l=1}^{t} \mathrm{P}_{f|1}(l). \quad (7)$$

4.2 Simulation Results

In this section we report the results of an experimental validation of the proposed analysis of the behavior of the RIP decoder. As a case study we chose a QC-LDPC code having the parity check matrix $\mathbf{H}$ formed by $n_0 = 2$ circulant blocks of size $p = 4801$, column weight $v = 45$ and we assessed the DFR varying the error weight t from 10 to 100, attempting to decode 10^6 error vectors for



(a) One Iteration (b) Two Iterations

Fig. 1. Experimental validation of the DFR estimates (Est.) through numerical simulations (Sim.). The QC-LDPC code parameters are $n_0 = 2$, $p = 4,801$ and $v = 45$. The decoding threshold is $b_0 = 25$.

each value of the error weight. To this end, we implemented the RIP decoder in `C99`, and run the experiments on an Intel Core i5-6500 CPU running at 3.20 GHz, compiling the code with the `GCC` 8.3.0 and running the built executables on Debian GNU/Linux 10.2 (stable). The computation of the worst case DFR estimates and bounds in Eq. (2) and Eq. (3) were realized employing the NTL library [24], while the solver for the counting version of the subset sum problem was implemented in plain `C++`. Computing the entire DFR upper bound relying on the counting subset sum problem takes significantly less than a second, for the selected parameters. We report the results considering a bit flipping threshold of $b = 25$, for all the iterations; however, we obtained analogous results varying the bit flipping threshold. The results with thresholds different from 25 are omitted for lack of space. Figure 1 reports the results of numerical simulations of the DFR of the RIP decoder running for either one or two iterations, while employing a random permutation (DFR_1 and DFR_2) or artificially computing the error estimates according to the worst-case permutation (DFR_1^* and DFR_2^*). As it can be seen, our technique for the DFR estimation provides a perfect match for the case of a single iteration, while our assumptions turn out to provide a conservative estimate for the worst-case DFR in the case of a 2-iteration RIP decoder. In both cases, the actual behavior of the decoder with a random permutation matches our expectations of having the DFR bounded by both the worst-case one and the closed form code-specific bound reported in green in Fig. 1. Finally, it is interesting to note, from an implementation viewpoint, that skipping the permutation in the case of a single-iteration RIP decoder appears to have no effect on the simulated DFR (black dots, marked DFR_I in Fig. 1). This can be explained observing that the first iteration of the RIP decoder is actually applying the random permutation to the positions of the error estimates which have randomly-placed discrepancies with the actual error itself.

5 Conclusions

We provided a statistical analysis of the behavior of a randomized in place bit flipping decoder, derived from the canonical one by randomizing the order in which the estimated error positions are processed. This modification to the decoder allows us to provide a statistical worst-case analysis of the DFR of the decoder at hand, both considering the average behavior among all the codes with the same length, dimension and number of parity checks, and a code-specific bound for a given QC-LDPC/QC-MDPC. The former analysis can be fruitfully exploited to design code parameters allowing to obtain DFR values such as the ones needed to employ QC-LDPC/QC-MDPC codes in constructions providing IND-CCA2 guarantees under the assumption that the underlying scheme is δ-correct [14]. The latter result allows us to analyze a given QC-LDPC/QC-MDPC code to assess whether the DFR it exhibits is above the maximum tolerable one for an IND-CCA2 construction, thus allowing us to discard weak keypairs upon generation. We note that our analysis relies on the RIP decoder performing a finite number of iterations, as opposed to the one provided in [23], in turn allowing a constant-time implementation of the RIP decoder itself. This fact is of significant practical relevance since the timing information leaked from decoders performing a variable number of iterations was shown to be as valuable as the one leaked by decryption failures to a CCA attacker [9, 22], leading to concrete violations of the IND-CCA2 property.

References

1. G. Agosta, A. Barenghi, G. Pelosi, and M. Scandale. Trace-based schedulability analysis to enhance passive side-channel attack resilience of embedded software. *Inf. Process. Lett.*, 115(2):292–297, 2015.
2. M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini. LEDAcrypt: QC-LDPC Code-Based Cryptosystems with Bounded Decryption Failure Rate. In M. Baldi, E. Persichetti, and P. Santini, editors, *Code-Based Cryptography - 7th International Workshop, CBC 2019, Darmstadt, Germany, May 18-19, 2019, Revised Selected Papers*, volume 11666 of *Lecture Notes in Computer Science*, pages 11–43. Springer, 2019.
3. M. Baldi, F. Chiaraluce, R. Garello, and F. Mininni. Quasi-Cyclic Low-Density Parity-Check Codes in the McEliece Cryptosystem. In *Proceedings International Conference on Communications (ICC 2007)*, pages 951–956, Glasgow, Scotland, Jun. 2007.
4. E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. On the inherent intractability of certain coding problems (Corresp.). *IEEE Trans. Information Theory*, 24(3):384–386, 1978.
5. D. J. Bernstein. Grover vs. McEliece. In *Proceedings Post-Quantum Cryptography: Third International Workshop (PQCrypto 2010)*, pages 73–80, Darmstadt, Germany, May 2010. Springer Berlin Heidelberg.

6. N. Bindel, M. Hamburg, K. Hövelmanns, A. Hülsing, and E. Persichetti. Tighter proofs of CCA security in the quantum random oracle model. Cryptology ePrint Archive, Report 2019/590, 2019. https://eprint.iacr.org/2019/590.
7. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.
8. N. Drucker and S. Gueron. A toolbox for software optimization of QC-MDPC code-based cryptosystems. Cryptology ePrint Archive, Report 2017/1251, 2017. https://eprint.iacr.org/2017/1251.
9. E. Eaton, M. Lequesne, A. Parent, and N. Sendrier. QC-MDPC: A timing attack and a CCA2 KEM. In T. Lange and R. Steinwandt, editors, *PQCrypto*, pages 47–76, Fort Lauderdale, FL, USA, Apr. 2018. Springer International Publishing.
10. T. Fabšič, V. Hromada, P. Stankovski, P. Zajac, Q. Guo, and T. Johansson. A Reaction Attack on the QC-LDPC McEliece Cryptosystem. In T. Lange and T. Takagi, editors, *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017*, pages 51–68. Springer, Utrecht, The Netherlands, June 2017.
11. J.-C. Faugère, A. Otmani, L. Perret, and J.-P. Tillich. Algebraic Cryptanalysis of McEliece Variants with Compact Keys. In H. Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 279–298. Springer, 2010.
12. R. G. Gallager. *Low-Density Parity-Check Codes*. PhD thesis, M.I.T., 1963.
13. Q. Guo, T. Johansson, and P. Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In J. H. Cheon and T. Takagi, editors, *ASIACRYPT 2016*, volume 10031 of *LNCS*, pages 789–815. Springer Berlin Heidelberg, 2016.
14. D. Hofheinz, K. Hövelmanns, and E. Kiltz. A Modular Analysis of the Fujisaki-Okamoto Transformation. In Y. Kalai and L. Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 341–371. Springer, 2017.
15. S. Mangard, E. Oswald, and T. Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007.
16. R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. *Deep Space Network Progress Report*, 44:114–116, Jan. 1978.
17. R. Misoczki, J.-P. Tillich, N. Sendrier, and P. L. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *Proceedings IEEE International Symposium on Information Theory (ISIT 2013)*, pages 2069–2073, Istambul, Turkey, Jul. 2013.
18. H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
19. A. Salomaa. Chapter II - Finite Non-deterministic and Probabilistic Automata. In A. Salomaa, editor, *Theory of Automata*, volume 100 of *International Series of Monographs on Pure and Applied Mathematics*, pages 71 – 113. Pergamon, 1969.
20. P. Santini, M. Battaglioni, M. Baldi, and F. Chiaraluce. Hard-Decision Iterative Decoding of LDPC Codes with Bounded Error Rate. In *Proceedings IEEE International Conference on Communications (ICC 2019)*, pages 1–6, Shanghai, China, May 2019.
21. P. Santini, M. Battaglioni, M. Baldi, and F. Chiaraluce. A theoretical analysis of the error correction capability of LDPC and MDPC codes under parallel bit-flipping decoding, 2019.
22. P. Santini, M. Battaglioni, F. Chiaraluce, and M. Baldi. Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes. In *CBC*, 2019.

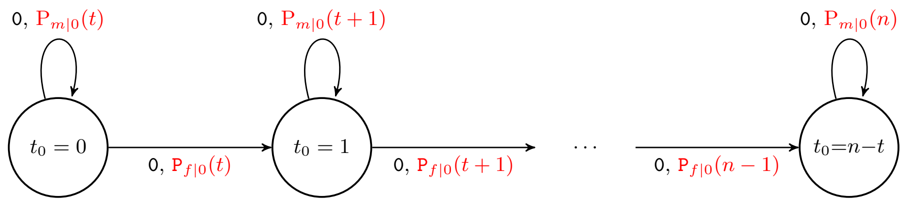


Fig. 2. Structure of the probabilistic FSA modeling the evolution of the distribution of the $\hat{t}_0$ variable. Read characters are reported in black, transition probabilities in red.

23. N. Sendrier and V. Vasseur. On the Decoding Failure Rate of QC-MDPC Bit-Flipping Decoders. In J. Ding and R. Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019, Chongqing, China, May 8-10, 2019 Revised Selected Papers*, volume 11505 of *Lecture Notes in Computer Science*, pages 404–416. Springer, 2019.
24. V. Shoup. NTL: A Library for doing Number Theory. http://shoup.net/ntl/, Version 11.4.1, 2019.
25. J. Tillich. The Decoding Failure Probability of MDPC Codes. In *2018 IEEE International Symposium on Information Theory, ISIT 2018, Vail, CO, USA, June 17-22, 2018*, pages 941–945. IEEE, 2018.

A Deriving the Bit-flipping Probabilities for the RIP Decoder

Denote with $\hat{t}_0 = |\{S(\mathbf{e} \oplus \bar{\mathbf{e}}) \cap E_0\}|$, that is the number of places where the estimated error at the beginning of the outer loop iteration $\bar{\mathbf{e}}$ differs from the actual $\mathbf{e}$, in positions included in E_0. Analogously, define $\hat{t}_1 = |\{S(\mathbf{e} \oplus \bar{\mathbf{e}}) \cap E_1\}|$.

We now characterize the statistical distribution of $\hat{t}_0$ and $\hat{t}_1$ after n iterations of the inner loop of the RIP-BF decoder are run, processing the estimated error bit positions in the order pointed out by $\pi^* \in \mathcal{P}_n^*$, i.e., the permutation which places at the end all the positions j where $\hat{e}_j \neq e_j$. We point out that, at the first iteration of the outer loop of the decoder, this coincides with placing all the positions where $e_j = 1$ at the end, since $\bar{\mathbf{e}}$ is initialized to the n-binary elements zero vector, hence $\bar{\mathbf{e}} \oplus \mathbf{e} = \mathbf{e}$.

In characterizing the distribution of $\hat{t}_0$, because of Assumption 1, we rely only on the probabilities $P_{f|0}(t)$ and $P_{m|0}(t)$, i.e. the probability that an error estimate bit will be flipped or maintained. In the following, for the sake of simplicity, we will consider $\hat{t}_1 = t$, which is the case of the RIP decoder performing the first outer loop iteration. To model the statistical distribution of $\hat{t}_0$ we employ the framework of Probabilistic Finite State Automata (PFSA) [19]. Informally, a PFSA is a Finite State Automaton (FSA) characterized by transition probabilities for each of the transitions of the FSA. The state of a PFSA is a discrete probability distribution over the set of FSA states and the probabilities of the transitions starting from the same FSA state, reading a the same symbol, must add up to one.

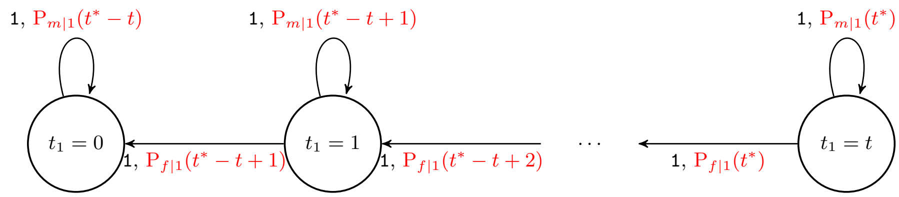


Fig. 3. Structure of the probabilistic FSA modeling the evolution of the distribution of the $\hat{t}_1$ variable. Read characters are reported in black, transition probabilites in red-

We model the statistical distribution of $\hat{t}_0$ as the state of a PFSA having $n-t$ FSA states, each one mapped onto a specific value for $\hat{t}_0$, as depicted in Figure 2. We consider the underlying FSA to be accepting the input language constituted by binary strings obtained as the sequences of $\hat{e}_j \neq e_j$ values, where j is the error estimate position being processed by the RIP decoder at a given inner loop iteration. We therefore have that, for the PFSA modeling the evolution of $\hat{t}_0$ while the RIP decoder acts on the first $n-t$ positions specified by π^*, all the read bits will be equal to 0, as π^* sorts the positions of $\hat{\mathbf{e}}$ so that the ($n-t$ at the first iteration) positions with no discrepancy between $\bar{\mathbf{e}}$ and $\mathbf{e}$ come first.

The transition probability for the PFSA transition from a state $\hat{t}_0 = i$ to $\hat{t}_0 = i+1$ requires the RIP decoder to flip a bit of $\hat{\mathbf{e}}$ equal to zero, and matching the one in the same position of $\mathbf{e}$, causing a discrepancy. Because of Assumption 1, the probability of such a transition is $\mathtt{P}_{f|0}(t+i)$. , while the probability of the self-loop transition from $\hat{t}_0 = i$ to $\hat{t}_0 = i$ itself is $\mathrm{P}_{m|0}(t+i)$.

Note that, during the inner loop iterations of the RIP decoder acting on positions of $\hat{\mathbf{e}}$ which have no discrepancies it is not possible to decrease the value $\hat{t}_0$, as no reduction on the number of discrepancies between $\hat{\mathbf{e}}$ and $\mathbf{e}$ can be done changing values of $\hat{\mathbf{e}}$ which are already equal to the ones in $\mathbf{e}$. Hence, we have that the probability of transitioning from $\hat{t}_0 = i$ to $\hat{t}_0 = i-1$ is zero.

The evolution of a PFSA can be computed simply taking the current state, represented as the vector $\mathbf{y}$ of probabilities for each FSA state and multiplying it by an appropriate matrix which characterizes the transitions in the PFSA. Such a matrix is derived as the adjacency matrix of the PFSA graph representation, keeping only the edges for which the read character matches the edge label, and substituting the one-values in the adjacency matrix with the probability labelling the corresponding edge. We obtain the transition matrix modeling an iteration of the RIP decoder acting on an $\hat{e}_j = e_j$ (i.e. reading a 0) as the $(n-t+1)\times(n-t+1)$ matrix:

$$\mathbf{K}_0 = \begin{bmatrix} \mathrm{P}_{m|0}(t) & \mathtt{P}_{f|0}(t) & 0 & 0 & 0 & 0 \\ 0 & \mathrm{P}_{m|0}(t+1) & \mathtt{P}_{f|0}(t+1) & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \mathrm{P}_{m|0}(n-1) & \mathtt{P}_{f|0}(n-1) \\ 0 & 0 & 0 & 0 & 0 & \mathrm{P}_{m|0}(n) \end{bmatrix}$$

Since we want to compute the effect on the distribution of $\hat{t}_0$ after $n-t$ iterations of the RIP decoder acting on positions j such that $\hat{e}_j = e_j$, we can obtain it simply as $\mathbf{y}\mathbf{K}_0^{n-t}$. Note that the subsequent t iterations of the RIP decoder will not alter the value of $\hat{t}_0$ as they act on positions j such that $e_j = 1$. Since we know that, at the beginning of the first iteration $\mathbf{y} = [\texttt{Prob}\left(\hat{t}_0 = 0\right) = 1, \texttt{Prob}\left(\hat{t}_0 = 1\right) = 0, \texttt{Prob}\left(\hat{t}_0 = 2\right) = 0, \cdots, \texttt{Prob}\left(\hat{t}_0 = n-t\right) = 0]$, we are able to compute $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$ as the $(x+1)$-th element of $\mathbf{y}\mathbf{K}_0^{n-t}$.

We now model the distribution of $\hat{t}_1$, during the last t iterations of the inner loop of the RIP decoder performed during an iteration of the outer loop. Note that, to this end, the first $n-t$ iterations of the inner loop have no effect on $\hat{t}_1$. Denote with t^* the incorrectly estimated bits $w_H(\mathbf{e}+\hat{\mathbf{e}})$ at the beginning of the inner loop iterations acting on positions j where $\hat{e}_j \neq e_j$. Note that, at the first iteration of the outer loop of the RIP decoder, $t^* = \hat{t}_0 + t$, when the RIP decoder is about to analyze the first position for which $w_H(\mathbf{e}+\bar{\mathbf{e}})$. Arguments analogous to the ones employed to model the PFSA describing the evolution for $\hat{t}_0$ allow us to obtain the one modeling the evolution for $\hat{t}_1$, reported in Figure 3.

We are thus able to obtain the $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$ PFSA reported in Figure 3 for $\hat{t}_1$ is $\mathbf{z} = [\texttt{Prob}\left(\hat{t}_1 = 0\right) = 0, \texttt{Prob}\left(\hat{t}_1 = 0\right) = 0, \ldots, \texttt{Prob}\left(\hat{t}_1 = t\right) = 1]$ and employing the $(t+1) \times (t+1)$ transition matrix $\mathbf{K}_1$ of the PFSA to compute $\mathbf{z}\mathbf{K}_1^t$. The value of $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$ corresponds to the $(x+1)$-th element of $\mathbf{z}\mathbf{K}_1^t$.

B Solving the Counting Subset Sum Problem

In the following, we describe the algorithm computing $\mathcal{N}(\mathbf{y}, \eta, \texttt{thr})$, i.e., the number of subsets of the elements of $\mathbf{y}$, which have cardinality equal to η, and which have the sum of their elements lesser than or equal to $\texttt{thr}$.

In doing this, we leverage the fact that $\mathbf{y}$ has only a small number of distinct elements, $z \ll n = |\mathbf{y}|$. To this end, we represent $\mathbf{y}$ as the sequence of its z distinct elements $[\epsilon_0, \epsilon_1, \ldots, \epsilon_{z-1}]$ in increasing order of their value, i.e., $\forall i < j, e_i < e_j$. Such a sequence is paired with the sequence of the number of times that each ϵ_i appears in $\mathbf{y}$, $[\lambda_0, \lambda_1, \ldots, \lambda_{z-1}]$.

First of all, we note that the sets which are counted in $\mathcal{N}(\mathbf{y}, \eta, \texttt{thr})$, can be partitioned according to the number of distinct elements contained in them. Denote with $\mathcal{N}_i(\mathbf{y}, \eta, \texttt{thr})$ the number of the number of subsets of the elements of $\mathbf{y}$, with cardinality equal to η, sum lesser or equal to $\texttt{thr}$, and exactly i distinct elements. The value of $\mathcal{N}(\mathbf{y}, \eta, \texttt{thr})$ is obtained as the sum over all $i \in 1, \ldots, z$ of the values of $\mathcal{N}_i(\mathbf{y}, \eta, \texttt{thr})$. The computation of $\mathcal{N}_i(\mathbf{y}, \eta, \texttt{thr})$ is described in Algorithm 2.

C Proof of Lemma 1

Lemma 3. *From Assumption 2, the probabilities that the i-th bit of the syndrome* $(0 \leq i \leq r-1)$ *is asserted knowing that the z-th bit of the error vector*

Algorithm 2: Computation of $\mathcal{N}_i(\mathbf{y}, \eta, \mathtt{thr})$

Input: $\mathbf{y}$: an integer sequence, with elements in $\{0, \ldots v\}$, $|\mathbf{y}| = n$. The collection admits repeated items
η: the number of elements of the sought subsets of $\mathbf{y}$
$\mathtt{thr}$: the maximum allowed value of the sum of the η-wide integer subsets of $\mathbf{y}$
i: the number of distinct elements admitted in the subsets
Output: $\mathcal{N}_i(\mathbf{y}, \eta, \mathtt{thr})$: the number of subsets of $\mathbf{y}$, of η integers picked with sum $\leq \mathtt{thr}$
Data: z: the number of distinct elements in $\mathbf{y}$
ϵ_i: the i-th distinct integer in $\mathbf{y}$, $i \in \{0, \ldots, z-1\}$, $i < j \Rightarrow \epsilon_i < \epsilon_j$
λ_i: the number of occurrences (multiplicity) of ϵ_i in $\mathbf{y}$

$\mathtt{sum} \leftarrow 0$
if $i = 1$ **then**
  for $j \leftarrow 0$ **to** $z - 1$ **do**
    // Pick η terms equal to ϵ_j: their sum should be $\leq \mathtt{thr}$
    if $(\epsilon_j \cdot \eta \leq \mathtt{thr}) \wedge (\lambda_j \geq \eta)$ **then**
      $\mathtt{sum} \leftarrow \mathtt{sum} + \binom{\lambda_i}{\eta}$
  return sum
else
  for $j \leftarrow 0$ **to** $z - 1$ **do**
    $m \leftarrow \min\{\lambda_j, \lfloor \frac{\mathtt{thr}}{\epsilon_j} \rfloor, \eta - (i-1)\}$
    // $i - 1$ *distinct* terms must still be placed: place at most $\eta - (i-1)$
    for $k \leftarrow 1$ *to* m **do**
      $\mathtt{sum} \leftarrow \mathtt{sum} + \binom{\lambda_j}{k} \mathcal{N}_{(i-1)}\left(\mathbf{y} \setminus \{\epsilon_0 \ldots \epsilon_j\}, \eta - k, \mathtt{thr} - (k \cdot \epsilon_j)\right)$
return sum

$(0 \leq z \leq n-1)$ *is null or not, i.e.,* $\mathtt{Prob}\left(s_i = 1 | e_z\right) = \mathtt{Prob}\left(\langle \mathbf{h}_{i,:}, \mathbf{e} \rangle = 1 | e_z\right)$, $\langle \mathbf{h}_{i,:}, \mathbf{e} \rangle = \bigoplus_{j=0}^{n-1} h_{i,j} \cdot e_j$, *can be expressed for each bit position* z, $0 \leq z \leq n-1$, *of the error vector as follows:*

$$\rho_{0,\mathtt{u}} = \mathtt{Prob}\left(\langle \mathbf{h}_{i,:}, \mathbf{e} \rangle = 1 \mid e_z = 0\right) = \frac{\sum_{l=0,\ l\ odd}^{\min\{w,t\}} \binom{w}{l}\binom{n-w}{t-l}}{\binom{n-1}{t}}$$

$$\rho_{1,\mathtt{u}} = \mathtt{Prob}\left(\langle \mathbf{h}_{i,:}, \mathbf{e} \rangle = 1 \mid e_z = 1\right) = \frac{\sum_{l=0,\ l\ even}^{\min\{w-1,t-1\}} \binom{w-1}{l}\binom{n-w}{t-1-l}}{\binom{n-1}{t-1}}$$

Consequentially, the probability that Algorithm 1 performs a bit-flip of an element of the estimated error vector, $\hat{e}_z$, *when the corresponding bit of the actual error vector is asserted,* $e_z = 1$, *i.e.,* $\mathrm{P}_{f|1}$, *and the probability that Algorithm 1 maintains the value of the estimated error vector,* $\hat{e}_z$, *when the corresponding bit of the actual error vector is null,* $e_z = 0$, *i.e.,* $\mathrm{P}_{m|0}$, *are:*

$$\mathrm{P}_{f|1} = \sum_{\mathtt{upc}=b}^{v} \binom{v}{\mathtt{upc}} \rho_{1,\mathtt{u}}^{\mathtt{upc}} (1 - \rho_{1,\mathtt{u}})^{v-\mathtt{upc}},$$

$$\mathrm{P}_{m|0} = \sum_{\mathtt{upc}=0}^{b-1} \binom{v}{\mathtt{upc}} \rho_{0,\mathtt{u}}^{\mathtt{upc}} (1 - \rho_{0,\mathtt{u}})^{v-\mathtt{upc}}.$$

Proof. For the sake of brevity, we consider the case of $e_z = 1$ deriving the expression of $\mathrm{P}_{f|1}$; the proof for $\mathrm{P}_{m|0}$ can be carried out with similar arguments.

Given a row $\mathbf{h}_{i,:}$ of the parity-check matrix $\mathbf{H}$, such that $z \in \mathrm{S}(\mathbf{h}_{i,:})$, the equation $\bigoplus_{j=0}^{n-1} h_{i,j} \cdot e_j$ (in the unknown $\mathbf{e}$) yields a non-null value for the i-th bit of the syndrome, s_i, (i.e., the eq. is unsatisfied) if and only if the support of the error vector $\mathbf{e}$ is such that $\bigoplus_{j=0}^{n-1} h_{i,j} \cdot e_j = 2a+1, a \geq 0$, including the term having $j = z$, i.e., $\mathbf{h}_{i,z} \cdot e_z = 1$. This implies that the cardinality of the set obtained intersecting the support $\mathbf{h}_{i,:}$ with the one of $\mathbf{e}$, $|(\mathrm{S}(\mathbf{h}_{i,:}) \setminus \{z\}) \cap (\mathrm{S}(\mathbf{e}) \setminus \{z\})|$, must be an even number, which in turn cannot be larger than the minimum between $|\mathrm{S}(\mathbf{h}_{i,:}) \setminus \{i\}| = w-1$ and $|\mathrm{S}(\mathbf{e}) \setminus \{i\}| = t-1$.

The probability $\rho_{1,\mathtt{u}}$ is obtained considering the fraction of the number of error vector values having an even number of asserted bits matching the asserted bits ones in a row of $\mathbf{H}$ (noting that, for the z-th bit position, both the error and the row of $\mathbf{H}$ are set) on the number of error vector values having $t-1$ asserted bits over $n-1$ positions, i.e., $\binom{n-1}{t-1}$. The numerator of the said fraction is easily computed as the sum of all error vector configurations having an even number $0 \leq l \leq \min\{w-1, t-1\}$ of asserted bits. Considering a given value for l, the counting of the error vector values is derived as follows. Picking one of vector with l asserted bits over w possible positions, i.e., one vector over $\binom{w-1}{l}$ possible ones, there are $\binom{n-w}{t-1-l}$ possible values of the error vector exhibiting $t-1-l$ null bits in the remaining $n-w$ positions; therefore, the total number of vectors with weigh l is $\binom{w-1}{l} \cdot \binom{n-w}{t-1-l}$. Repeating the same line of reasoning for each value of l allows to derive the numerator of the formula defining $\rho_{1,\mathtt{u}}$.

From Assumption 2, the value of any row $\mathbf{h}_{i,:}$ is modeled as a random variable with a Bernoulli distribution having parameter (or expected value) $\rho_{1,\mathtt{u}}$, and each of these random variables is independent from the others. Consequentially, the probability that Algorithm 1 performs a bit-flip of an element of the estimated error vector when the corresponding bit of the actual error vector is asserted and the counter of the unsatisfied parity checks (upc) is above or equal to a given threshold b, is derived as the binomial probability obtained adding the outcomes of v (column-weight of $\mathbf{H}$) i.i.d. Bernoulli trials. □

D Proof of Lemma 2

Lemma 2. The execution path of the inner loop in Algorithm 1 yielding the worst possible decoder success rate is the one taking place when $\pi^* \in \mathcal{P}_n^*$ is applied at the beginning of the outer loop, that is:

$$\forall \pi \in \mathcal{P}_n, \forall \pi^* \in \mathcal{P}_n^*, \;\; \mathtt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} | \;\pi \in \mathcal{P}_n\right) \leq \mathtt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} | \;\pi^* \in \mathcal{P}_n^*\right).$$

Proof. First of all, we can write $\mathtt{Prob}\left(\mathbf{e}' \neq \mathbf{e} | \;\pi \in \mathcal{P}_n\right) = 1 - \beta(\pi)$, where $\beta(\pi)$ is the probability that all bits, evaluated in the order specified by π, are correctly processed. To visualize the effect of a permutation $\pi^* \in \mathcal{P}_n$, we can consider the following representation

$$\pi^*(\mathbf{e}) \oplus \pi^*(\bar{\mathbf{e}}) = [\underbrace{0, 0, \cdots, 0}_{\text{length } n-\hat{t}}, \underbrace{1, 1, \cdots, 1}_{\text{length } \hat{t}}], \;\; \forall \pi^* \in \mathcal{P}_n^*.$$

The decoder will hence analyze first a run of $n-\hat{t}$ positions where the differences between the permuted error $\pi^*(\mathbf{e})$ vector and $\pi^*(\bar{\mathbf{e}})$ contain only zeroes, followed by a run of $\hat{t}$ positions containing only ones. Thus, we have that

$$\beta(\pi^*) = \left(\mathrm{P}_{m|0}(\hat{t})\right)^{n-\hat{t}} \cdot \mathrm{P}_{f|1}(\hat{t}) \cdot \mathrm{P}_{f|1}(\hat{t}-1) \cdots \mathrm{P}_{f|1}(1)$$

The former expression can be derived thanks to Assumption 1 as follows. Note that, the first elements in the first $n-\hat{t}$ positions of $\pi^*(\hat{\mathbf{e}})$ and $\pi^*(\mathbf{e})$ match, therefore the decoder makes a correct evaluation if it does not change the value of $\pi^*(\hat{\mathbf{e}})$. This in turn implies that, in case a sequence of $n-\hat{t}$ correct decisions are made in the corresponding iterations of the inner loop, each iteration will have the same probability $\mathrm{P}_{m|0}(\hat{t})$ correctly evaluating the current estimated error bit. This leads to a probability of performing the first $n-\hat{t}$ iterations taking a correct decision equal to $\left(\mathrm{P}_{m|0}(\hat{t})\right)^{n-\hat{t}}$ Through an analogous line of reasoning, observe that the decoder will need to change the value of the current estimated error bit during the last $\hat{t}$ iterations of the inner loop. As a consequence, if all correct decisions are made, the number of residual errors will decrease by one at each inner loop iteration, yielding the remaining part of the expression.

Consider now a generic permutation π, such that the resulting $\pi(\mathbf{e})$ has support $\{u_0, \cdots, u_{\hat{t}-1}\}$; we have

$$\begin{aligned}\beta(\pi) &= \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_0} \mathrm{P}_{f|1}(t) \left[\mathrm{P}_{m|0}(\hat{t}-1)\right]^{u_1-u_0-1} \mathrm{P}_{f|1}(\hat{t}-1) \cdots \mathrm{P}_{f|1}(1) \left[\mathrm{P}_{m|0}(0)\right]^{n-1-u_{\hat{t}-1}} \\ &= \left[\mathrm{P}_{m|0}(t)\right]^{u_0} \left[\mathrm{P}_{m|0}(0)\right]^{n-1-u_{\hat{t}-1}} \prod_{j=1}^{\hat{t}-1} \left[\mathrm{P}_{m|0}(\hat{t}-j)\right]^{u_j-u_{j-1}-1} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l).\end{aligned}$$

We now show that we always have $\beta(\pi) \geq \beta(\pi^*)$. Indeed, since $\mathrm{P}_{m|0}(0) = 1$ and due to the monotonic trends of P_u and P_f, the following chain of inequalities can be derived

$$\begin{aligned}\beta(\pi) &= \left[\mathrm{P}_{m|0}(0)\right]^{n-1-u_{\hat{t}-1}} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_0} \prod_{j=1}^{\hat{t}-1} \left[\mathrm{P}_{m|0}(\hat{t}-j)\right]^{u_j-u_{j-1}-1} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l) \\ &\geq \left[\mathrm{P}_{m|0}(0)\right]^{n-1-u_{\hat{t}-1}} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_0} \prod_{j=1}^{\hat{t}-1} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_j-u_{j-1}-1} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l) \\ &= \left[\mathrm{P}_{m|0}(0)\right]^{n-1-u_{\hat{t}-1}} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_0} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_{\hat{t}-1}-u_0-(\hat{t}-1)} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l)) \\ &= \left[\mathrm{P}_{m|0}(0)\right]^{n-1-u_{\hat{t}-1}} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_{\hat{t}-1}-(\hat{t}-1)} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l) \\ &\geq \left[\mathrm{P}_{m|0}(\hat{t})\right]^{n-1-u_{\hat{t}-1}} \left[\mathrm{P}_{m|0}(\hat{t})\right]^{u_{\hat{t}-1}-(\hat{t}-1)} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l) \\ &= \left[\mathrm{P}_{m|0}(\hat{t})\right]^{n-\hat{t}} \prod_{l=0}^{\hat{t}-1} \mathrm{P}_{f|1}(\hat{t}-l) = \beta(\pi^*).\end{aligned}$$

□