A Failure Rate Model of Bit-flipping Decoders for QC-LDPC and QC-MDPC Code-based Cryptosystems

Marco Baldi[1][a], Alessandro Barenghi[2][b], Franco Chiaraluce[1][c], Gerardo Pelosi[2][d]
and Paolo Santini[1][e]

[1] *DII, Università Politecnica delle Marche, Via Brecce Bianche 12, Ancona, Italy*
[2] *DEIB, Politecnico di Milano, Piazza Leonardo da Vinci 32, Milano, Italy*

Keywords: Bit-flipping Decoding, Code-based Cryptosystems, Decoding Failure Rate, LDPC Codes, MDPC Codes, Quasi-cyclic Codes, Post-quantum Cryptosystems.

Abstract: The design of quantum-resistant cryptographic primitives has gained attraction lately, especially thanks to the U.S.A. National Institute of Standards and Technology (NIST) initiative, which is selecting a portfolio of primitives for standardization. A prime position in the set of asymmetric encryption primitives is occupied by the ones relying on decoding random linear error correction codes as their trapdoor. Among these primitives, the LEDAcrypt and BIKE cryptosystems have been admitted to the second round of the standardization initiative. They are based on the adoption of iteratively decoded Low- and Moderate-Density Parity Check (LDPC/MDPC) codes. Characterizing the decoding failure rate of such codes under iterative decoding is paramount to the security of both the LEDAcrypt and BIKE second round candidates to achieve indistinguishability under adaptive chosen ciphertext attacks (IND-CCA2). For these codes, we propose a new iterative decoder, obtained through a simple modification of the classic in-place bit-flipping decoder and, in this paper, we provide a statistical worst-case analysis of its performance. This result allows us to design parameters for LDPC/MDPC code-based cryptosystems with guaranteed extremely low failure rates (e.g., 2^{-128}), fitting the hard requirement imposed by IND-CCA2 constructions.

1 INTRODUCTION

Building asymmetric cryptosystems resistant against attacks performed with quantum computers requires a change in the underlying computationally hard problem exploited to build a trapdoor. To this end, the aim is to pick a problem either belonging, or equivalent to one belonging, to the NP-Complete class (Karp, 1972). Problems belonging to such a computational complexity class are recognized to not have a polynomial-time solution algorithm, even on quantum computers. Among these problems we find the one of decoding a syndrome obtained with a random linear block code, and the one of finding a fixed weight codeword in the said code (Berlekamp et al., 1978). Indeed, McEliece was the first to propose a cryptosystem relying on the hardness of the decoding problem (McEliece, 1978), followed by the work of Niederreiter (Niederreiter, 1986). The trapdoor function in these cryptosystems is constituted by the fact that the secret linear block code they employ is not random, but instead it is picked from a family of linear block codes for which an efficient decoding algorithm is known. To prevent an attacker from deciphering, both cryptosystems provide as a public key an obfuscated representation of the efficiently decodable code, obtained multiplying either its generator matrix, or its parity-check matrix, by a random non-singular conformant one. Code-based cryptosystems have a remarkably good security track, whenever the hidden code structure is a binary Goppa code (which is the one originally proposed by McEliece). However, such strength comes at the disadvantage of quite large public key sizes (in the hundreds of kilobytes to a megabyte range). Willing to reduce the key size, a significant amount of research work was aimed at employing code families admitting a space-efficient representation. Quasi-Cyclic (QC) codes provide a good avenue to decrease the required key sizes by two

[a] https://orcid.org/0000-0002-8754-5526
[b] https://orcid.org/0000-0003-0840-6358
[c] https://orcid.org/0000-0001-6994-1448
[d] https://orcid.org/0000-0002-3812-5429
[e] https://orcid.org/0000-0003-0631-3668

Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G. and Santini, P.
A Failure Rate Model of Bit-flipping Decoders for QC-LDPC and QC-MDPC Code-based Cryptosystems.
DOI: 10.5220/0009891702380249
In *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications (ICETE 2020) - SECRYPT*, pages 238-249
ISBN: 978-989-758-446-6

orders of magnitude or more. Such codes are characterized by generator and parity-check matrices that are quasi-cyclic, i.e., composed by circulant square blocks, where all rows are obtained cyclically rotating the first one. It is therefore sufficient to store only the first row of such matrices to preserve their entire representation. However, employing QC algebraic codes has proven to yield security concerns, as the additional structure given by the quasi-cyclicity might allow for the efficient recovery of the underlying secret code (Faugère et al., 2010), and this was reflected by the elimination of QC algebraic candidates from the NIST standardization process (Alagic et al., 2019). By contrast, families of QC codes defined by a pseudo-random sparse parity-check matrix without any underlying algebraic structure, other than their sparsity, do not suffer from the same security issues, thus leading to the well-assessed proposal of Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) codes (Baldi et al., 2007) or Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) codes (Misoczki et al., 2013) as code families to build instances of either the McEliece or the Niederreiter cryptosystem.

The downside of such code families resides in their decoding procedures, which have a non null probability of failing, known as Decoding Failure Rate (DFR). This, in turn, implies that a decryption action, which relies on the (secret) code decoder, may fail even on a valid ciphertext. Besides the possible functional hindrances caused, a non null DFR imposes the strict requirement to employ such code-based encryption/decryption algorithms as building blocks of constructions providing resistance against active attackers. Indeed, active attacks, such as the ones in (Guo et al., 2016; Fabšič et al., 2017; Santini et al., 2019b) were shown to be able to exploit the availability of a decryption oracle, which is the typical scenario of a Chosen Ciphertext Attack (CCA), to extract information on the secret QC-LDPC/QC-MDPC code. To completely avoid these kinds of attacks and attain Indistinguishability under Adaptively Chosen Ciphertext Attack (IND-CCA2) guarantees, state-of-the-art constructions, such as (Hofheinz et al., 2017; Bindel et al., 2019), require the failure rate of the scheme to be negligible in the security parameter. Essentially, this implies that codes exhibiting a probability of decoding failure equal or lower than $2^{-\lambda}$ must be employed to design a cryptosystem with security parameter equal to λ. It is clear that assessing such low values of DFR through numerical simulations is unfeasible; thus, a reliable model for the behavior of decoders for QC-LDPC and QC-MDPC codes has become a major need.

Usually, these cryptosystems employ the Bit-Flipping (BF) decoding technique, originally proposed by Gallager (Gallager, 1963), because of its good trade-off between error correction capability and computational complexity. Indeed, the use of common MDPC parameters yields a $O(n^{1.5})$ average decoding complexity, where n is the codeword length. Briefly, a BF algorithm performs syndrome decoding through an iterative procedure, in which the bit locations where the received message is in error are estimated relying on the value of the received syndrome. The syndrome is subsequently updated according to the derived estimate, and the procedure is repeated until either a null syndrome is obtained (indicating a decoding success), or a prefixed maximum number of iterations is reached. Depending on the strategy employed to update the syndrome, BF decoders are classified in two major classes: *out-of-place* algorithms and *in-place* algorithms. In the former algorithms the syndrome is updated after all bits have been evaluated, while in the latter ones the update is instantly performed each time a bit is deemed as an erroneous one, and thus flipped. The decoder operating principles, together with the rule employed to detect errors, is crucial in determining its failure rate. The impossibility of validating the DFR through numerical simulations has spurred significant efforts in modelling the behaviour of decoders for QC-LDPC and QC-MDPC codes to find reliable tools for the DFR assessment.

Contributions. We provide a theoretical analysis of the DFR of an in-place iterative 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. In particular, we employ well established assumptions in coding theory to develop a closed-form statistical model of the said decoder, keeping into account the worst-case execution at each iteration for the average QC-LDPC/QC-MDPC code. We provide a numerical validation of our model, showing that it actually provides conservative estimates of the actual decoder DFR. Finally, we show practical sets of code parameters for use in QC-LDPC/QC-MDPC code-based cryptosystems exhibiting a DFR$= 2^{-\lambda}$ with $\lambda \in \{128, 192, 256\}$ and targeting a security level equivalent to breaking an instance of the AES block cipher with 128-, 192-, or 256-bit key with pre- and post-quantum resources.

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 will be denoted as $\mathbf{A}_{i,:}$ and $\mathbf{A}_{:,j}$, respectively, while the entry on the i-th row, j-th column as $a_{i,j}$. The null vector of length n will be denoted as $\mathbf{0}_n$. Given a vector $\mathbf{a}$, its length will be denoted as $|\mathbf{a}|$, while its i-th element as a_i, with $0 \leq i \leq |\mathbf{a}|-1$. Finally, the *support* (i.e., the set of positions of the asserted elements in a sequence) and the Hamming weight of $\mathbf{a}$ will be reported as $\mathrm{S}\left(\mathbf{a}\right)$ and $\mathrm{w_H}\left(\mathbf{a}\right)$, respectively. We will use $\mathcal{P}_n$ to denote the set of permutations of n elements, represented as bijections on the set of integers $\{0,\ldots,n-1\}$. For $\pi \in \mathcal{P}_n$ and an integer $i \in \{0,\ldots,n-1\}$, we will write $\pi(i) = j$ if the image of i, according to permutation π, is j. For a vector $\mathbf{a} \in \mathbb{F}_2^n$, we will use $\pi(\mathbf{a})$ to denote the vector which is obtained by permuting each of the entries of $\mathbf{a}$ according to π. We will write $\pi \overset{\$}{\leftarrow} \mathcal{P}_n$ to denote a permutation π that is randomly and uniformly picked among the elements in $\mathcal{P}_n$.

As far as the cryptosystems are concerned, in the following we will make use of a QC-LDPC/QC-MDPC code $\mathcal{C}$, with length $n = n_0 p$, dimension $k = (n_0-1)p$ and redundancy $r = n-k = p$ to correct t intentional errors. The private-key will coincide with the parity-check matrix $\mathbf{H} = \left[\mathbf{H}_0, \mathbf{H}_1, \cdots, \mathbf{H}_{n_0-1}\right] \in \mathbb{F}_2^{r \times n}$, 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.

In the case of a Niederreiter scheme, the public-key is defined as the systematic parity-check matrix of the code $\mathbf{M} \in \mathbb{F}_2^{r \times n}$ and derived from the private-key as $\mathbf{M} = \mathbf{H}_{n_0-1}^{-1}\mathbf{H}$, while the plaintext message is mapped to an error vector $\mathbf{e} \in \mathbb{F}_2^{1 \times n}$ having $\mathrm{w_H}\left(\mathbf{e}\right) = t$ asserted bits. The encryption algorithm outputs the syndrome $\mathbf{c} = \mathbf{e}\mathbf{M}^\top \in \mathbb{F}_2^{1 \times r}$, which has the meaning of ciphertext. The decryption algorithm takes as input $\mathbf{c}$ and the private-key $\mathbf{H}$ to compute a private-syndrome $\mathbf{s} \in \mathbb{F}_2^{1 \times r}$ such that $\mathbf{s} = \mathbf{c}\mathbf{H}_{n_0-1}^\top = \mathbf{e}\mathbf{M}^\top\mathbf{H}_{n_0-1}^\top = \mathbf{e}\mathbf{H}^\top(\mathbf{H}_{n_0-1}^\top)^{-1}\mathbf{H}_{n_0-1}^\top = \mathbf{e}\mathbf{H}^\top$. Subsequently, to derive the original plaintext message $\mathbf{e}$, the decryption algorithm feeds with the private-key and the computed private-syndrome a BF *syndrome decoding* algorithm.

In the case of the McEliece scheme, the public-key is chosen as the systematic generator matrix of the code: $\mathbf{G} \in \mathbb{F}_2^{k \times n}$. The ciphertext is in the form $\mathbf{c} = \mathbf{m}\mathbf{G} + \mathbf{e} \in \mathbb{F}_2^{1 \times n}$, where $\mathbf{m} \in \mathbb{F}_2^{1 \times k}$ is a plaintext message encoded with k bits, and $\mathbf{e} \in \mathbb{F}_2^{1 \times n}$ is a n-bit error vector with exactly $\mathrm{w_H}\left(\mathbf{e}\right) = t$ asserted bits. The decryption algorithm takes as input the ciphertext $\mathbf{c}$ and the private-key $\mathbf{H}$ to compute the syndrome $\mathbf{s} = \mathbf{c}\mathbf{H}^\top = \mathbf{e}\mathbf{H}^\top \in \mathbb{F}_2^{1 \times r}$ and feed a syndrome decoding algorithm, from which $\mathbf{m}$ is recovered, employing the generation matrix and the vector $\mathbf{c} - \mathbf{e}$.

2.1 Bit-flipping Decoding

The BF decoder is an iterative, syndrome decoding procedure, originally proposed by Gallager (Gallager, 1963), which takes as input a syndrome $\mathbf{s} = (\mathbf{c} + \mathbf{e})\mathbf{H}^\top = \mathbf{e}\mathbf{H}^\top \in \mathbb{F}_2^{1 \times r}$, where $\mathbf{c} \in \mathbb{F}_2^{1 \times n}$ and $\mathbf{e} \in \mathbb{F}_2^{1 \times n}$ denote the codeword and the unknown error vector, respectively. The algorithm computes an estimate of the error vector, that we denote as $\hat{\mathbf{e}}$, and which is initialized as the null vector. For each bit of the error estimate, a decision is taken on the ground of the number of unsatisfied parity-check equations in which it participates, for a generic error bit in position $0 \leq j \leq n-1$, such quantity is computed as:

$$\mathtt{upc}_j = \sum_{i \in \{\mathrm{S}(\mathbf{H}_{:,j}) \cap \{0,\ldots,r-1\}\}} s_i.$$

Indeed, note that the constant term of a generic i-th parity-check equation ($0 \leq i \leq r-1$) corresponds to the i-th entry in $\mathbf{s} = \mathbf{e}\mathbf{H}^\top$, and that the equations influenced by the j-th bit of the error vector coincide with the ones having an asserted bit at the j-th column of $\mathbf{H}$. Then, the number of unsatisfied parity-check equations in which the j-th bit participates can be simply obtained by summing the entries of the syndrome which are indexed by the set of positions in $\mathrm{S}\left(\mathbf{H}_{:,j}\right)$. When $\mathtt{upc}_j$ exceeds a given threshold (e.g., when more than a half of the parity check equations in which the j-th error bit is involved as in the original proposal by Gallager), then the error bit is flipped and the syndrome is coherently updated replacing its current value with $\mathbf{s} \oplus \mathbf{H}_{:,j}$. In a decoding iteration, all error bits are evaluated following their positional order from 0 to $n-1$; and the decoding repeats the decoder iteration until either a null syndrome is obtained or a prefixed maximum number of iterations is reached. The procedure sketched above corresponds to an in-place decoding strategy. When an out-of-place strategy is employed, every error bits is assessed relying on the syndrome value provided as input at the beginning of the iteration, while the updates of both the error vector and the syndrome are postponed after all error bits have been evaluated.

3 IN-PLACE RANDOMIZED BIT-FLIPPING DECODER

In this section we describe a simple modification to the canonical in-place decoder proposed by Gallager, for which we provide a closed form estimate of the DFR in a worst-case scenario.

Algorithm 1: IR-BF decoder.

```
    Input: $\mathbf{s} \in \mathbb{F}_2^{1\times r}$: syndrome
           $\mathbf{H} \in \mathbb{F}_2^{r\times n}$: parity-check matrix with
                      column-weight $v$
    Output: $\hat{\mathbf{e}} \in \mathbb{F}_2^{1\times n}$: recovered error value
            $\mathbf{s} \in \mathbb{F}_2^{1\times r}$: syndrome; if error $\hat{\mathbf{e}} = \mathbf{e}$
                      then $\mathbf{s}$ is null
    Data: $\mathtt{imax} \geq 1$: maximum number of
                      (outer loop) iterations;
          $\mathbf{b} = [\mathtt{b}_0, \ldots, \mathtt{b}_k, \ldots, \mathtt{b}_{\mathtt{imax}-1}]$, with
          $\mathtt{b}_k \in \{\lceil \frac{v}{2} \rceil, \ldots, v\}$: flip thresholds

 1  $\mathtt{iter} \leftarrow 0$
 2  $\hat{\mathbf{e}} \leftarrow \mathbf{0}_n$ // estimated error initialization
 3  while $(\mathtt{iter} < \mathtt{imax}) \wedge (\mathrm{w_H}(\mathbf{s}) > 0)$ do
 4      $\pi \xleftarrow{\$} \mathcal{P}_n$ // random permut. of size $n$
 5      for $j \leftarrow 0$ to $n-1$ do
 6          $\ell \leftarrow \pi(j)$ // permuted bit index
 7          $\mathtt{upc} \leftarrow 0$ // integer value
 8          for $i \in \mathrm{S}(\mathbf{H}_{:,\ell})$ do
 9              $\mathtt{upc} \leftarrow \mathtt{upc} + s_i$
10          if $\mathtt{upc} \geq \mathtt{b}_{\mathtt{iter}}$ then
11              $\hat{\mathrm{e}}_\ell \leftarrow \hat{\mathrm{e}}_\ell \oplus 1$ // error update
12              $\mathbf{s} \leftarrow \mathbf{s} \oplus \mathbf{H}_{:,\ell}$ // syndrome update
13      $\mathtt{iter} \leftarrow \mathtt{iter} + 1$ // counter update
14  return $\{\mathbf{s}, \hat{\mathbf{e}}\}$
```

3.1 An In-place, Randomized Bit-flipping Decoder

Algorithm 1 reports an in-place BF decoder where the estimates on the error vector bits are computed in the order driven by a randomly picked permutation π. For this reason, we denote this decoder as In-place Randomized Bit-Flipping (IR-BF) decoder. Introducing this randomization of the bit estimate evaluation order allows us to derive an effective worst case analysis for the DFR, as we describe in Section 3.2.

The inputs to the decoding algorithm are the binary parity-check matrix $\mathbf{H}$, the syndrome $\mathbf{s}$, the maximum number of iterations $\mathtt{imax}$ and a vector $\mathbf{b}$ of length $\mathtt{imax}$, such that its k-th entry, $\mathtt{b}_k$, with $0 \leq k \leq \mathtt{imax} - 1$, is employed during the k-th iteration as a threshold on the value of the unsatisfied parity-check counters to trigger a flip of the corresponding error bit estimates or not.

For each outer loop iteration (beginning at line 3), a permutation is randomly generated (line 4) to establish the evaluation order of the bits in the estimated error vector, for the current iteration. The algorithm proceeds applying the said permutation to each value taken by the counter $j \in \{0, 1, \ldots, n-1\}$ of the inner loop iterations (line 5) to obtain the bit position $\ell = \pi(j)$ of the estimated error vector to be processed during the inner loop iteration at hand. The number of unsatisfied parity-checks ($\mathtt{upc}$) in which the ℓ-th bit of the error estimate $\hat{\mathbf{e}}$ is involved is computed by summing the syndrome bits having a position corresponding to the asserted elements of the ℓ-th column of the parity check matrix $\mathbf{H}$ (lines 7–9). If the number of unsatisfied parity-checks in which the ℓ-th bit participates exceeds the threshold, $\mathtt{b}_{\mathtt{iter}}$, chosen for the current outer loop iteration, then the value of the ℓ-th position of the estimated error vector, $\hat{e}_\ell$, is changed (i.e., flipped, hence the name of the decoding technique) and the value of the syndrome is updated to reflect this change, adding to it the ℓ-th column of $\mathbb{H}$ (lines 10–12). Once the inner loop at lines 5–12 terminates, the decoder has completed the iteration, and thus proceeds to increment the iteration counter $\mathtt{iter}$ and checks whether the syndrome is the null vector, or not, or if the maximum number of iterations is reached. We note that this classical formulation of the IR-BF decoder does not entail a constant iteration number. However, it is readily transformed into one with a constant iteration number substituting the while loop at lines 3–13 with a countable for loop executing exactly $\mathtt{imax}$ iterations. Indeed, executing extra iterations of the IR-BF algorithm when the syndrome is already the null vector does not alter the correctness of the results. Indeed, once the syndrome is the null vector, all the $\mathtt{upc}$ values will be equal to zero, and, since the least functional threshold is strictly positive, the estimate-changing if construct at lines 10–12 is never triggered.

3.2 Assessing Bit-flipping Probabilities

In the following we describe a statistical approach to model the behaviour of the IR-BF decoder. From now on, we will employ the following notation:

$\mathbf{e}$ denotes the actual error vector, with Hamming weight weight t;

$\bar{\mathbf{e}}$ denotes the error estimate at the beginning of the outer loop of Algorithm 1 (line 3), while $\hat{\mathbf{e}}$ will denote the error estimate at the beginning of the inner loop of Algorithm (line 5). In other words, $\bar{\mathbf{e}}$ is a snapshot of the error estimate made by the IR-BF 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;

$\bar{\mathbf{e}}' = \mathbf{e} \oplus \bar{\mathbf{e}}$, that is, $\bar{\mathbf{e}}'$ denotes the vector such that its asserted positions are only those corresponding to positions in which $\mathbf{e}$ and $\bar{\mathbf{e}}$ are different; the number of such mismatches is denoted as $\bar{t} = \mathrm{w_H}\left(\bar{\mathbf{e}}'\right)$. In analogous way, we define $\hat{\mathbf{e}}' = \mathbf{e} \oplus \hat{\mathbf{e}}$ and $\hat{t} = \mathrm{w_H}\left(\hat{\mathbf{e}}'\right)$.

We remark that, for the first decoding iteration, we have $\bar{\mathbf{e}} = \mathbf{0}_n$, $\bar{\mathbf{e}}' = \mathbf{e}$ and $\bar{t} = t$. To avoid cumbersome notation, we will not introduce analogous formalism for the syndrome and will always use $\mathbf{s}$ to denote it. At the beginning of the outer loop iteration in Algorithm 1, the syndrome corresponds to $(\mathbf{e} \oplus \bar{\mathbf{e}})\mathbf{H}^\top = \bar{\mathbf{e}}'\mathbf{H}^\top$ while, inside the inner loop iteration, an estimate $\hat{\mathbf{e}}$ will be associated to the syndrome $(\mathbf{e} \oplus \hat{\mathbf{e}})\mathbf{H}^\top = \hat{\mathbf{e}}'\mathbf{H}^\top$.

We introduce the following probabilities:

- $\mathrm{P}_{f|1} = \mathtt{Prob}\left((j\text{-th } \mathtt{upc}) \geq \mathtt{b_{iter}} \mid e_j \oplus \hat{e}_j = 1\right)$ as the probability that the computation of the j-th `upc` yields an outcome greater than or equal to the current threshold (thus, triggering a flip of $\hat{e}_j$), conditioned by the event $e_j \oplus \hat{e}_j = 1$;
- $\mathrm{P}_{m|0} = \mathtt{Prob}\left((j\text{-th } \mathtt{upc}) < \mathtt{b_{iter}} \mid e_j \oplus \hat{e}_j = 0\right)$ as 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 event $e_j \oplus \hat{e}_j = 0$.

To derive a theoretical model for the DFR of the IR-BF decoder, we assume that the above probabilities only depend on the number of mismatches between the actual error and the estimated one, that is, on the weight of $\hat{\mathbf{e}}'$; formally, the following property is assumed to hold.

Assumption 1. *Both* $\mathrm{P}_{f|1}$ *and* $\mathrm{P}_{m|0}$ *are not a function of the bit-position (i.e., j in their definitions), although both probabilities are a function of the total number* $\hat{t} = \mathrm{w_H}\left(\hat{\mathbf{e}}'\right) = \mathrm{w_H}\left(\mathbf{e} \oplus \hat{\mathbf{e}}\right)$ *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 $w = n_0 v$. We observe that Algorithm 1 uses the columns of the parity-check matrix, for each outer loop iteration, in an order chosen by the random permutation at line 4, and that the computation accumulating the syndrome bits into the `upc` at lines 8–9 is independent of the order in which they are added.

Following these observations, we consider the structure of the parity-check matrix, to be such that each row of $\mathbf{H}$ is independent of 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, the following assumption holds.

Assumption 2. *Let* $\mathbf{H}$ *be an* $r \times n$ *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* $\hat{\mathbf{e}}'$ *that is modeled as a sample from a uniform random variable distributed over the elements in* $\mathbb{F}_2^{1\times n}$ *with weight* $\hat{t}$.
We assume that each row $\mathbf{h}_{i,:}$, $0 \leq i \leq r-1$, *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.

We note that the same assumption was adopted since the inception of LDPC codes in (Gallager, 1963) and, more recently also in (Chaulet, 2017, Lemmae 2.3-2.5) and (Tillich, 2018). From Assumption 2, the probabilities $\mathrm{P}_{f|1}$ and $\mathrm{P}_{m|0}$ can be derived as stated in the following Lemma

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

$$\mathtt{Prob}\left(\langle \mathbf{h}_{i,:}, \hat{\mathbf{e}}' \rangle = 1 \mid \hat{e}'_z = 0\right) = \rho_{0,\mathtt{u}},$$

$$\mathtt{Prob}\left(\langle \mathbf{h}_{i,:}, \hat{\mathbf{e}}' \rangle = 1 \mid \hat{e}'_z = 1\right) = \rho_{1,\mathtt{u}},$$

with:

$$\rho_{0,\mathtt{u}} = \frac{\sum_{j=0,\ j \text{ odd}}^{\min\{w,\hat{t}\}} \binom{w}{j}\binom{n-w}{\hat{t}-j}}{\binom{n-1}{\hat{t}}},$$

$$\rho_{1,\mathtt{u}} = \frac{\sum_{j=0,\ j \text{ even}}^{\min\{w-1,\hat{t}-1\}} \binom{w-1}{j}\binom{n-w}{\hat{t}-1-j}}{\binom{n-1}{\hat{t}-1}}.$$

Consequently, denoting the decoding threshold as $\mathtt{b} \in \{1, \ldots, v\}$, *the probabilities* $\mathrm{P}_{f|1}$ *and* $\mathrm{P}_{m|0}$ *are obtained as*

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

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

The proof, which follows a straightforward counting argument, is omitted for the sake of brevity, and can be found in full in (Baldi et al., 2019b, Lemma 3.1.1)).

3.3 Determining a Worst-case Iteration Scenario for the IR-BF Decoder

In this section we focus on a single iteration of the outer loop of Algorithm 1 and derive a statistical model for the IR-BF decoder, employing the probabilities $\mathrm{P}_{f|1}$ and $\mathrm{P}_{m|0}$ as derived in the previous section, under Assumption 1.

In particular, we consider a *worst-case* evolution for the decoder, proving what is the computation path which ends in a decoding success with the lowest probability. To this end, we denote a decoding success the case when 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 $\mathrm{w_H}\left(\mathbf{e}\oplus\hat{\mathbf{e}}\right)=0$.

Let $\bar{\mathbf{e}}$ denote the error estimate at the beginning of the iteration, and $\bar{t}=\mathrm{w_H}\left(\mathbf{e}\oplus\bar{\mathbf{e}}\right)$ denote the corresponding number of residual mismatched bit estimations. We will study, in statistical terms, the evolution of the number of mismatches between the vectors $\mathbf{e}$ and $\hat{\mathbf{e}}$, which we denote with $\hat{t}$. From now on, we highlight the dependence of $\mathrm{P}_{f|1}$ and $\mathrm{P}_{m|0}$ on the current value of $\hat{t}$, writing them down as $\mathrm{P}_{f|1}(\hat{t})$ and $\mathrm{P}_{m|0}(\hat{t})$, respectively.

We denote as π the permutation picked in line 4 of Algorithm 1 and as $\mathcal{P}_n^*$ the set of permutations $\pi^*\in\mathcal{P}_n^*$ such that

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

Let $\mathtt{Prob}\left(\hat{\mathbf{e}}\neq\mathbf{e}|\ \pi\in\mathcal{P}_n\right)$ be the probability that the estimated error vector $\hat{\mathbf{e}}$ at the end of the current inner loop iteration is different from $\mathbf{e}$, conditional on the fact that the permutation π was applied before the inner loop execution started. Similarly, we define $\mathtt{Prob}\left(\hat{\mathbf{e}}\neq\mathbf{e}|\ \pi^*\in\mathcal{P}_n^*\right)$, by considering π^* in place of π.

It can be verified that $\mathrm{P}_{f|1}(\hat{t})\geq\mathrm{P}_{f|1}(\hat{t}+1)$, and $\mathrm{P}_{m|0}(\hat{t})\geq\mathrm{P}_{m|0}(\hat{t}+1)$, $\forall\hat{t}$, as increasing the number of current mis-estimated error bits, increases the likelihood of a wrong decoder decision. By leveraging Assumptions 1 and 2 , we now prove that the decoder reaches a correct decoding at the end of the outer loop, with the lowest probability, each time a permutation $\pi^*\in\mathcal{P}_n^*$ 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. In other words, $\forall\pi\in\mathcal{P}_n$, and $\forall\pi^*\in\mathcal{P}_n^*$, the following inequality holds*

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

Proof. We consider one execution of the outer loop in Algorithm 1, and denote with $\bar{t}$ the initial number of mismatches between the actual error (that is, $\mathbf{e}$) and its estimate (that is, $\bar{\mathbf{e}}$). We can write $\mathtt{Prob}\left(\hat{\mathbf{e}}\neq\mathbf{e}|\ \pi\in\mathcal{P}_n\right)=1-\beta(\pi)$, 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

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

The decoder will hence analyze first a run of $n-\bar{t}$ 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 $\bar{t}$ positions containing only ones. Thus, we have that

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

The former expression can be derived thanks to Assumption 1 as follows. Note that, the first elements in the first $n-\bar{t}$ positions of $\pi^*(\bar{\mathbf{e}})$ and $\pi^*(\mathbf{e})$ match, therefore the decoder makes a correct evaluation if it does not change the corresponding entries in $\hat{\mathbf{e}}$. This implies that, in case a sequence of $n-\bar{t}$ correct decisions are made in the corresponding iterations of the inner loop, each iteration will have the same probability $\mathrm{P}_{m|0}(\bar{t})$ of correctly evaluating the current estimated error bit. This leads to a probability of performing the first $n-\bar{t}$ iterations taking a correct decision equal to $\left(\mathrm{P}_{m|0}(\bar{t})\right)^{n-\bar{t}}$.

Through an analogous line of reasoning, we observe that the decoder will need to change the value of the current estimated error bit during the last $\bar{t}$ 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 $\{u_0,\cdots,u_{\bar{t}-1}\}$; we have

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

We now show that we always have $\beta(\pi)\geq\beta(\pi^*)$.

Indeed, since $\mathrm{P}_{m|0}(0)=1$, due to the monotonic trends of the probabilities $\mathrm{P}_{m|0}(\hat{t})$ and $\mathrm{P}_{f|1}(\hat{t})$, the following chain of inequalities can be derived

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

□

3.4 A Worst-case DFR Estimate for the IR-BF Decoder

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. We consider one outer loop iteration, and denote with $\bar{\mathbf{e}}$ the error estimate at the beginning of the iteration; we recall that the number of mismatches between $\bar{\mathbf{e}}$ and $\mathbf{e}$ is denoted as $\bar{t} = \mathrm{w_H}\left(\mathbf{e}\oplus\bar{\mathbf{e}}\right)$. Let us define the following two sets: $E_1{=}S(\mathbf{e}\oplus\bar{\mathbf{e}})$, and $E_0{=}\{0,\ldots,n-1\}\setminus E_1$. Denote with $\hat{t}_0$ the number of places where the estimated error differs from the actual $\mathbf{e}$, in positions included in E_0. At the beginning of the outer loop iteration, we have

$$\hat{t}_0 = \left|\mathrm{S}\left(\mathbf{e}\oplus\bar{\mathbf{e}}\right)\cap E_0\right| = 0.$$

Analogously, we define $\hat{t}_1$ as the number of positions in which the estimated error and the actual one differ, in positions included in E_1; at the beginning of the outer loop iteration, we have $\hat{t}_0 = 0$ and

$$\hat{t}_1 = \left|\mathrm{S}\left(\mathbf{e}\oplus\bar{\mathbf{e}}\right)\cap E_1\right| = \bar{t}.$$

Let

i) $\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$ denote the probability that the decoder in Algorithm 1, starting from a state where $\mathrm{w_H}\left(\bar{\mathbf{e}}\oplus\mathbf{e}\right) = \omega$, and acting in the order specified by a worst case permutation $\pi^* \in \mathcal{P}_n^*$, ends in a state with $\hat{t}_0 = x$ residual errors among the bits indexed by E_0 after completing the inner loop at lines 5 – 12;

ii) $\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$ denote the probability that the decoder in Algorithm 1, starting from a state where $\mathrm{w_H}\left(\bar{\mathbf{e}}\oplus\mathbf{e}\right) = \omega$, 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 5–12;

iii) $\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow[i]{} x\right)$ be the probability that, starting from a state such that $\mathrm{w_H}\left(\bar{\mathbf{e}}\oplus\mathbf{e}\right) = \omega$, after i iterations of the outer loop at lines 5–12 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 the Appendix, and only depend on $\mathrm{P}_{f|1}(\hat{t})$ and $\mathrm{P}_{m|0}(\hat{t})$.

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

First of all, we straightforwardly have

$$\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow[1]{} x\right) =$$
$$= \sum_{\delta=\max\{0\ ;\ x-(n-\omega)\}}^{\omega} \mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x-\delta\right)\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} \delta\right).$$

From now on, we will denote with $\hat{\mathbf{e}}^{(\mathtt{iter})}$ the error vector estimate after the $\mathtt{iter}+1$ outer loop iterations; coherently, we write $\hat{t}^{(i)} = \mathrm{w_H}\left(\mathbf{e}\oplus\hat{\mathbf{e}}^{(\mathtt{iter})}\right)$, 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

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

where, to have a consistent notation, we consider $\hat{t}^{(-1)} = t$.

The above formula takes into account all possible transitions starting from an initial number of residual errors equal to t and ending in $\hat{t}^{(\mathtt{imax}-1)}$ residual errors Taking this probability into account, the DFR after $\mathtt{imax}$ iterations is

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

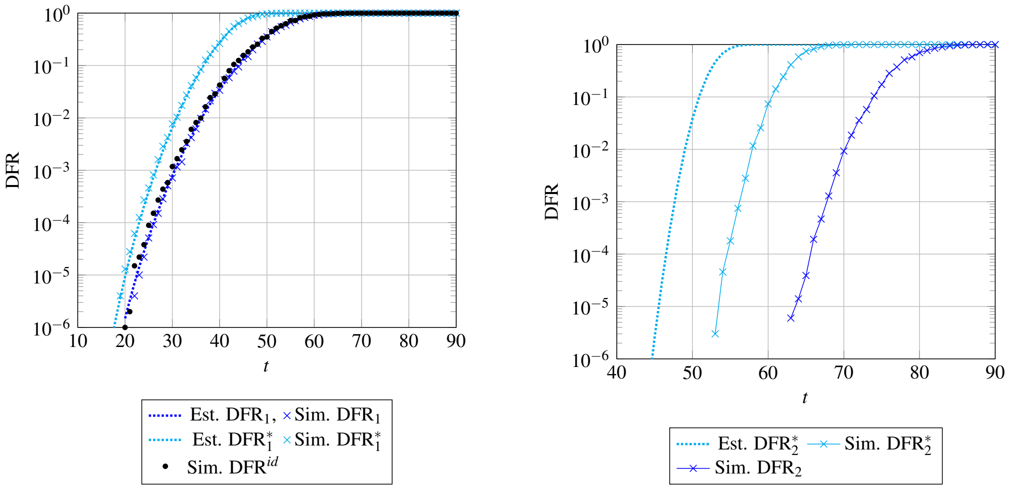
In the case of the decoder performing just one iteration, the expression of the DFR, keeping into account Lemma 2, is

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

A bonus point of the analysis we propose is that it is easy to obtain also an estimate for the average DFR of one decoding iteration, (i.e., corresponding to one outer loop iteration, using a random permutation π). Indeed, let $\pi(\mathbf{e})$ be the vector obtained by applying the permutation π on $\mathbf{e}$, with support $\mathrm{S}\left(\pi(\mathbf{e})\right)$. Let a_i, a_{i+1} be two consecutive elements of $\mathrm{S}\left(\pi(\mathbf{e})\right)$, with $0 \le i \le t-2$, and denote as d the average zero-run length in $\mathbf{e}$. It can be easily seen that, when π is randomly drawn from $\mathcal{P}_n$ and $\mathbf{e}$ is uniformly distributed over all binary n-uples, then the average zero-run length between two consecutive set entries in $\mathbf{e}$ corresponds to $d = \frac{n-t}{t+1}$.

Consequently, we can obtain a simple estimate for the average DFR after one iteration as

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



(a) One Iteration

(b) Two Iterations

Figure 1: Numerical validation of the DFR estimates (Est.) through numerical simulations (Sim.). The QC-MDPC code parameters are $n_0 = 2$, $p = 4801$ and $v = 45$. Figure (a) refers to the case of one decoding iteration (i.e., $\mathtt{imax} = 1$), figure (b) refers to a maximum number of decoding iterations equal to 2 (i.e., $\mathtt{imax} = 2$). The chosen decoding threshold, for both cases, is $\mathtt{b} = 25$. The results marked with "Est." are obtained via the computation of closed formulas as opposed to the ones marked "Sim." which are the result of a numerical simulation.

4 NUMERICAL VALIDATION

In this section we report the results of the numerical validation of the proposed analysis for the IR-BF decoder as well as practical sets of code parameters for use in QC-LDPC/QC-MDPC code-based cryptosystems. With the goal of validating our bounds on the DFR by simulations, we choose a QC-MDPC code having the parity check matrix $\mathbf{H}$ formed by $n_0 = 2$ circulant blocks of size $p = 4801$ and column weight $v = 45$. We assess the DFR varying the error weight t from 10 to 100, attempting to decode $10^6 \approx 2^{20}$ error vectors for each value of the error weight. We report the results considering a bit-flipping threshold of $\mathtt{b} = 25$, for all iterations; however, we obtained analogous results varying the bit-flipping threshold. The results with thresholds different from 25 are omitted for lack of space.

We implemented the IR-BF decoder in `C99`, compiled the code with the `GCC` 8.3.0 on Debian GNU/Linux 10.2 (stable), and run the experiments on an Intel Core i5-6500 CPU running at 3.20 GHz. The computation of the worst case DFR estimates were realized employing the NTL library.

Figure 1 reports the results of numerical simulations of the DFR of the IR-BF decoder (*Sim.* datasets in Figure 1) running for either one or two iterations, while employing either a random permutation (DFR_1 and DFR_2, respectively) or one selected among the worst case ones (DFR_1^* and DFR_2^*, respectively). For the case of one iteration, also the simulated DFR without the initial permutation, noted as DFR^{id}, is considered. Note that we are able to pick one of the worst-case permutations in practice since the actual error vector $\mathbf{e}$ is known to us, therefore allowing the computation of the discrepancies between the current error estimate and the actual error itself. The results are matched against the closed-form estimates as derived in the previous section.

As it can be seen in Figure 1 (a), our technique for the DFR estimate provides a perfect match for the case of a single iteration. Indeed, our estimated worst-case DFR (dotted cyan line) matches perfectly the simulated DFR picking a worst-case permutation π^* (cyan × symbols), and dominates the actual simulated DFR (blue × symbols). Finally, we also observed that omitting the permutation before the first iteration has no practical impact on the DFR. Such a fact can be easily justified by the random nature of the error vector, which in turn makes the initial positions where a discrepancy between the error estimate at the beginning of the decoding (which is completely null) and the error vector itself already completely random. This fact can be observed looking at the black dots in Fig. 1 (a), which report the values of DFR^{id}, and observing that they essentially match the DFR of the decoder employing a random permutation (blue × symbols). Finally, we note that our simple technique to estimate the average DFR of the IR-BF decoder (depicted as a dotted blue line) also provides

Table 1: Designed parameters, targeting the security levels of 2^{λ} and DFR$=2^{-\lambda}$, $\lambda=\{128,192,256\}$.

Security Level	v	t	2 iter.s (Santini et al., 2019) p	τ	1st iter. IR-BF + 2nd iter. (Baldi et al., 2019d) p	τ	2 iter.s IR-BF p
2^{128}	71	130	28,277	10	26,171	10	19,813
2^{192}	103	195	52,667	15	50,227	15	38,069
2^{256}	137	260	83,579	18	80,309	18	61,211

a good match for the actual DFR itself. Considering the case of a two iterations IR-BF decoder, reported in Fig. 1 (b), we note how our technique provides a conservative estimate for the worst case DFR of the IR-BF decoder. The previous comparison shows that, for the range of values that can be reached with numerical simulations, the presented theoretical analysis yields conservative estimates for the DFR of the IR-BF decoder. Thus, we have employed the presented DFR analysis to design parameters for code-based cryptosystems employing QC-LDPC/QC-MDPC codes, targeting a security level equivalent to breaking an instance of the AES block cipher with 128-, 192-, or 256-bit keys. Focusing on the case of $n_0 = 2$, we provide the resulting size and column weight of the circulant blocks in the parity-check matrix $\mathbf{H}$, which we respectively denote with p and v, in Table 1; the number of errors which need to be corrected (denoted with t in the table) has been computed to guarantee security levels of 2^{λ} (Baldi et al., 2019a). We recall that, in the example of a Niederreiter based key encapsulation mechanism, employing a quasi-cyclic parity-check matrix with two circulant blocks, as it is the case in the reported parameters, the public key of the cryptosystem will be p bit long and the encapsulated session key will also be p bits long.

Table 1 compares the parameter sets for the two-iterations out of place decoder proposed in (Santini et al., 2019), a decoder obtained with an iteration of the IR-BF decoder, followed by one iteration of the out-of-place LEDAcrypt decoder (Baldi et al., 2019d), computing the resulting DFR on the base of the number of residual error distribution after the first iteration provided by our technique, and a two-iteration IR-BF decoder. In the table, the values of τ refers to the number of errors that can be corrected with certainty by an iteration of an out-of-place BF decoder. As it can be seen from the reported results, even employing a hybrid approach, where the first decoder iteration is performed in-place by the IR-BF decoder, and the second one is performed out-of-place, allows a small reduction of the key size. Moving to our in-place decoding strategy allows to reduce the public key and ciphertext size by $\approx 25\%$ with respect to the approach described in (Santini et al., 2019).

5 RELATED WORK AND DISCUSSION

A code-specific analysis to establish the total error correction capability (i.e., null DFR) for a single BF iteration has first appeared in (Tillich, 2018) and has then been improved in (Santini et al., 2019). With similar arguments, an assumption-free, conservative upper bound for the DFR of a single decoder iteration was derived in (Santini et al., 2019a). However, employing such approaches to design the secret code parameters results in impractically large public-key sizes. To obtain keys with smaller size, in (Tillich, 2018; Santini et al., 2019; Baldi et al., 2019c; Baldi et al., 2018) the DFR of a two-iteration out-of-place decoder is analyzed, providing a closed-form method to derive an upper bound on the average DFR over all the QC-LDPC/QC-MDPC codes with the same length, rate and density, under reasonable assumptions. However, the second and final decoding iteration is analyzed in a conservative way, thus designing code parameters which may be further improved.

In (Sendrier and Vasseur, 2019b), the authors propose a characterization of a variant of the out-of-place decoder based on the extrapolation of the DFR curve in the desired regime of low DFR values, starting from higher DFR values estimated through numerical simulations. This method assumes that the exponentially decreasing trend of the DFR curve is steady as the code length increases, while all the other parameters are kept constant. This assumption is made in the scenario where numerical simulations allow to examine a DFR in the range of 2^{-27}, assuming that the trend is still the same for DFR values of 2^{-128} and lower. A qualitative justification is provided for this assumption in the appendix of (Sendrier and Vasseur, 2019a). In the said appendix, the authors rely on the so-called *concavity assumption*, according to which the DFR curve remains concave for all values of practical interest. Such an assumption in turn implies that the so-called *error floor* region of the DFR curve, where the said curve changes concavity, and that is present in all LDPC/MDPC codes, after the so-called *waterfall* region, does not occur for DFR values of practical interest.

We find this assumption to be difficult to maintain, since predicting the beginning of the error floor region is an extremely challenging task, which has currently no satisfactory closed form solution. Indeed, phenomena such as the existence of the so-called *trapping sets* (particular sets of error patterns which cause an iterative decoder to fail), which are deemed to have a negligible impact in the assumption made in (Sendrier and Vasseur, 2019a), are one of the prime objects of study to determine the location of the error floor region (Richardson, 2003; Hashemi and Banihashemi, 2019).

We note that if either a concavity change, or simply the change in the rate of the exponential decrease of the DFR curve before the concavity change, takes place before the region of practical interest, the extrapolations made in (Sendrier and Vasseur, 2019a) will provide cryptosystem parameters which are not matching the DFR needed in IND-CCA2 constructions. We therefore believe that relying on DFR curve extrapolations may provide overly optimistic cryptosystem parameter designs (Drucker and Gueron, 2017; Drucker et al., 2019).

In (Sendrier and Vasseur, 2019a), the authors also analyze an in-place decoding algorithm, called *Step-by-step* decoder modeling its DFR. The proposed analysis however, obtains a DFR estimate which is lower than the actual DFR obtained via numerical simulation, and thus cannot be employed when an upper bound of the DFR value is desired. Furthermore, the proposed analysis considers the asymptotic behaviour of the *Step-by-step* decoder when an infinite number of iterations is performed. Such an approach provides a practical hindrance in principle to the implementation of the decoding procedure as a constant time one, as there is no fixed upper bound to the number of iterations a-priori.

In this work, we obtain a characterization of a simple in-place decoder with a finite number of iterations, allowing its constant-time implementation in practice. Our characterization provides a statistical model which, by considering the worst case evaluation of the decoder, provides a conservative estimate of the decoder evolution. As a result, we do not rely on any specific *a-priori* assumption on the behaviour of the DFR curve but, on the contrary, completely derive it as a function of the scheme parameters and the decoder setting.

6 CONCLUSION

We have presented a statistical analysis of the behaviour of an in-place randomized bit-flipping (IR-BF) decoder, derived from the classic in-place bit-flipping decoder by randomizing the order in which the estimated error positions are processed. The said modification allows us to derive a worst-case analysis for the DFR of syndrome-decoding based systems, which is employed to design code parameters for QC-LDPC/QC-MDPC based cryptoystems matching the DFR figures of merit needed to provide IND-CCA2 guarantees. Differently from other solutions available at the state-of-the-art, the proposed analysis allows to fix the number of iterations of the IR-BF decoder a-priori (e.g., $\texttt{imax} = 2$), allowing an easy constant-time implementation, preventing timing-based side channel attacks.

REFERENCES

Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Smith-Tone, D., and Liu, Y.-K. (2019). Status report on the first round of the NIST post-quantum cryptography standardization process. Technical Report NISTIR 8240.

Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., and Santini, P. (2018). LEDAkem: A post-quantum key encapsulation mechanism based on QC-LDPC codes. In Lange, T. and Steinwandt, R., editors, *Post-Quantum Cryptography - 9th Int.'l Conference, PQCrypto 2018, Fort Lauderdale, April 9-11, 2018*, volume 10786 of *LNCS*, pages 3–24. Springer.

Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., and Santini, P. (2019a). A finite regime analysis of information set decoding algorithms. *Algorithms*, 12:209.

Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., and Santini, P. (2019b). LEDAcrypt 2.5 specification. https://www.ledacrypt.org/documents/LEDAcrypt_spec_2_5.pdf.

Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., and Santini, P. (2019c). LEDAcrypt: QC-LDPC code-based cryptosystems with bounded decryption failure rate. In Baldi, M., Persichetti, E., and Santini, P., editors, *Code-Based Cryptography. CBC 2019*, volume 11666 of *LNCS*, pages 11–43. Springer, Cham.

Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., and Santini, P. (2019d). LEDAcrypt website. https://www.ledacrypt.org/.

Baldi, M., Chiaraluce, F., Garello, R., and Mininni, F. (2007). Quasi-cyclic low-density parity-check codes in the McEliece cryptosystem. In *ICC 2007*.

Barenghi, A., Fornaciari, W., Galimberti, A., Pelosi, G., and Zoni, D. (2019). Evaluating the Trade-offs in the Hardware Design of the LEDAcrypt Encryption Functions. In *26th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2019, Genoa, Italy, November 27-29, 2019*, pages 739–742. IEEE.

Berlekamp, E. R., McEliece, R. J., and van Tilborg, H. C. A. (1978). On the inherent intractability of certain coding problems. *IEEE Trans. Information Theory*, 24(3):384–386.

Bindel, N., Hamburg, M., Hövelmanns, K., Hülsing, A., and Persichetti, E. (2019). Tighter proofs of CCA security in the quantum random oracle model. https://eprint.iacr.org/2019/590.

Chaulet, J. (2017). *Etude de cryptosystèmes à clé publique basés sur les codes MDPC quasi-cycliques*. PhD thesis, Université Pierre et Marie Curie - Paris VI.

Drucker, N. and Gueron, S. (2017). A toolbox for software optimization of QC-MDPC code-based cryptosystems. Cryptology ePrint Archive, Report 2017/1251. https://eprint.iacr.org/2017/1251.

Drucker, N., Gueron, S., and Kostic, D. (2019). QC-MDPC decoders with several shades of gray. eprint 2019/1423. https://eprint.iacr.org/2019/1423.

Fabšič, T., Hromada, V., Stankovski, P., Zajac, P., Guo, Q., and Johansson, T. (2017). A reaction attack on the QC-LDPC McEliece cryptosystem. In *PQCrypto 2017*. Springer.

Faugère, J.-C., Otmani, A., Perret, L., and Tillich, J.-P. (2010). Algebraic cryptanalysis of McEliece variants with compact keys. In *EUROCRYPT*, volume 6110 of *LNCS*. Springer.

Gallager, R. G. (1963). *Low-Density Parity-Check Codes*. PhD thesis, M.I.T.

Guo, Q., Johansson, T., and Stankovski, P. (2016). A key recovery attack on MDPC with CCA security using decoding errors. In *ASIACRYPT 2016*, volume 10031 of *LNCS*, pages 789–815. Springer.

Hashemi, Y. and Banihashemi, A. H. (2019). Characterization and efficient search of non-elementary trapping sets of LDPC codes with applications to stopping sets. *IEEE Trans. Inf. Theory*, 65(2):1017–1033.

Hofheinz, D., Hövelmanns, K., and Kiltz, E. (2017). A modular analysis of the Fujisaki-Okamoto transformation. In *Theory of Cryptography. TCC 2017*, volume 10677 of *LNCS*, pages 341–371. Springer.

Karp, R. M. (1972). Reducibility among combinatorial problems. In Miller, R. E., Thatcher, J. W., and Bohlinger, J. D., editors, *Complexity of Computer Computations*, pages 85–103. Springer, Boston, MA.

McEliece, R. J. (1978). A public-key cryptosystem based on algebraic coding theory. *Deep Space Network Progress Report*, 44.

Misoczki, R., Tillich, J.-P., Sendrier, N., and Barreto, P. L. (2013). MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *ISIT 2013*, Istambul, Turkey.

Niederreiter, H. (1986). Knapsack-type cryptosystems and algebraic coding theory. *Prob. Con. and Inf. Theory*, 15.

Richardson, T. (2003). Error floors of LDPC codes. In *Proc. 41st Annu. Allerton Conf. Commun. Control Comput.*, pages 1426–1435, Monticello, IL, USA.

Salomaa, A. (1969). Chapter II - Finite Non-deterministic and Probabilistic Automata. In *Theory of Automata*, volume 100 of *of Monographs on Pure and Applied Mathematics*. Pergamon.

Santini, P., Battaglioni, M., Baldi, M., and Chiaraluce, F. (2019). Hard-decision iterative decoding of LDPC codes with bounded error rate. In *Proc. IEEE Conf. on Communications (ICC 2019)*, Shanghai, China.

Santini, P., Battaglioni, M., Baldi, M., and Chiaraluce, F. (2019a). A theoretical analysis of the error correction capability of LDPC and MDPC codes under parallel bit-flipping decoding. https://arxiv.org/abs/1910.00472.

Santini, P., Battaglioni, M., Chiaraluce, F., and Baldi, M. (2019b). Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes. In *Code-Based Cryptography. CBC 2019*. Springer , Cham.

Sendrier, N. and Vasseur, V. (2019a). About low DFR for QC-MDPC decoding. Cryptology ePrint Archive, Report 2019/1434. https://eprint.iacr.org/2019/1434.

Sendrier, N. and Vasseur, V. (2019b). On the decoding failure rate of QC-MDPC bit-flipping decoders. In *PQCrypto 2019*, volume 11505 of *LNCS*, pages 404–416. Springer, Cham.

Tillich, J. (2018). The decoding failure probability of MDPC codes. In *ISIT 2018*, Vail, CO, USA.

APPENDIX

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 IR-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 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 $\mathrm{P}_{f|0}(\hat{t})$ and $\mathrm{P}_{m|0}(\hat{t})$, 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 IR-BF 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) (Salomaa, 1969). 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 the same symbol, must add up to one.

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

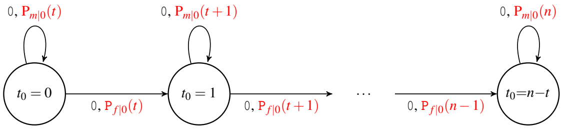


Figure 2: Structure of the probabilistic FSA modeling the evolution of the distribution of the $\hat{t}_0$ variable. Read characters are reported in black, transition probabilities in red.

Fig. 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 IR-BF decoder at a given inner loop iteration. We therefore have that, for the PFSA modeling the evolution of $\hat{t}_0$ while the IR-BF 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 IR-BF 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 IR-BF 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 IR-BF decoder acting on an $\hat{e}_j = e_j$ (i.e., reading a $\mathtt{0}$) 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 IR-BF decoder acting on positions j such that $\hat{e}_j = e_j$, we can obtain it simply as $\mathbf{yK}_0^{n-t}$. Note that the subsequent t iterations of the IR-BF 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 $\mathbf{y} = [\mathtt{Prob}\left(\hat{t}_0 = 0\right) = 1, \mathtt{Prob}\left(\hat{t}_0 = 1\right) = 0, \mathtt{Prob}\left(\hat{t}_0 = 2\right) = 0, \cdots, \mathtt{Prob}\left(\hat{t}_0 = n-t\right) = 0]$, we compute $\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$ as the $(x+1)$-th element of $\mathbf{yK}_0^{n-t}$. We now model the distribution of $\hat{t}_1$, during the last t iterations of the inner loop of the IR-BF 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 $\mathrm{w_H}\left(\mathbf{e}+\hat{\mathbf{e}}\right)$ 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 IR-BF decoder, $t^* = \hat{t}_0 + t$, when the IR-BF decoder is about to analyze the first position for which $\mathrm{w_H}\left(\mathbf{e} \oplus \bar{\mathbf{e}}\right)$. We now model the distribution of $\hat{t}_1$, during the last $\hat{t}$ rounds of the loop in the randomized in-place iteration function. Note that, to this end, the first $n-t$ iterations of the inner loop have no effect on $\hat{t}_1$. Denote with $\tilde{t}$ the incorrectly estimated bits in $\hat{\mathbf{e}}' = \mathbf{e} \oplus \hat{\mathbf{e}}$, that is, $\tilde{t} = \mathrm{w_H}\left(\hat{\mathbf{e}}'\right)$, when the iteration function is about to evaluate the first of the positions j where $\hat{e}_j \neq e_j$. Note that, at the beginning we have $\tilde{t} = \tilde{t}_0 + \hat{t}_1$, where $\tilde{t}_0$ is the number of discrepancies in the first $n - \hat{t}$ positions when the iteration is about to analyze the first position of $\hat{e}$ for which $\mathrm{w_H}\left(\mathbf{e}+\hat{\mathbf{e}}\right)$. Analogously to the PFSA describing the evolution for $\hat{t}_0$, we obtain the PFSA modeling the evolution of $\hat{t}_1$, reported in Fig. 3. The initial distribution of the values of $\hat{t}_1$, constituting the initial state of the PFSA in Fig. 3, is such that $\mathtt{Prob}\left(\hat{t}_1 = t\right) = 1$, corresponding to the $\hat{t}_1$-element vector $\mathbf{z} = [0, 0, \ldots, 0, 1]$. The transition matrix of the PFSA is

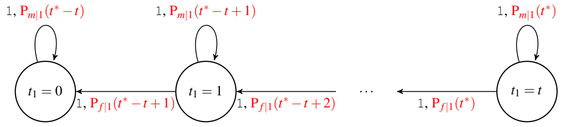

1, $\mathrm{P}_{m|1}(t^*-t)$
1, $\mathrm{P}_{m|1}(t^*-t+1)$
1, $\mathrm{P}_{m|1}(t^*)$
$t_1 = 0$
$t_1 = 1$
$t_1 = t$
1, $\mathrm{P}_{f|1}(t^*-t+1)$
1, $\mathrm{P}_{f|1}(t^*-t+2)$
$\cdots$
1, $\mathrm{P}_{f|1}(t^*)$

Figure 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 probabilities in red.

$$\mathbf{K}_1 = \begin{bmatrix} \mathrm{P}_{m|1}(\tilde{t}-t) & 0 & 0 & 0 & 0 & 0 \\ \mathrm{P}_{f|1}(\tilde{t}-t+1) & \mathrm{P}_{m|1}(\tilde{t}-t+1) & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \mathrm{P}_{f|1}(\tilde{t}-1) & \mathrm{P}_{m|1}(\tilde{t}-1) & 0 \\ 0 & 0 & 0 & 0 & \mathrm{P}_{f|1}(\tilde{t}) & \mathrm{P}_{m|1}(\tilde{t}) \end{bmatrix}.$$

We are thus able to obtain the $\mathtt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$ computing $\mathbf{zK}_1^t$ and taking its $(x+1)$-th element.