Analysis of In-Place Randomized Bit-Flipping Decoders for the Design of LDPC and MDPC Code-based Cryptosystems

Marco Baldi[1][0000−0002−8754−5526], Alessandro Barenghi[2][0000−0003−0840−6358], Franco Chiaraluce[1][0000−0001−6994−1448], Gerardo Pelosi[2][0000−0002−3812−5429], and Paolo Santini[1][0000−0003−0631−3668]

[1] Università Politecnica delle Marche - DII, Via Brecce Bianche 12, Ancona, Italy
{m.baldi, f.chiaraluce, p.santini}@staff.univpm.it
[2] Politecnico di Milano - DEIB, Piazza Leonardo da Vinci 32, Milano, Italy
{alessandro.barenghi, gerardo.pelosi}@polimi.it

Abstract. We present a variant of the classic in-place bit-flipping decoder, frequently used with Low- and Moderate-Density Parity Check (LDPC/MDPC) codes, which allows a statistical analysis of the achievable decoding failure rate (DFR) in worst-case conditions. Such evaluation is of paramount importance in code-based post-quantum cryptography (PQC) where the ability to achieve indistinguishability under adaptive chosen ciphertext attacks strictly depends on being able to ensure very low DFR values (e.g., in the order of 2^{-128} or lower) that, as such, are practically impossible to validate via numerical simulation. We provide theoretical evidence of the proposed approach and demonstrate its correctness through numerical examples. Moreover, we investigate the effect of changing the bit flipping decision threshold on the provided worst case analysis. Finally, we give design parameters for code-based cryptosystems employing Quasi-Cyclic LDPC/MDPC codes, able to achieve the security levels required in the NIST PQC standardization initiative which is currently in progress.

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

1 Introduction

Among the various options proposed for a new generation of asymmetric cryptosystems, able to counter the advent of quantum computers, a prominent role is played by code-based cryptosystems. McEliece [21] was the first to prove the computational hardness of decoding error affected codewords from random linear codes, in particular showing that determining the existence of such an error vector belongs to the NP-Complete class [20]. The original McEliece cryptosystem builds a trapdoor from an obfuscation of the generator matrix of a Goppa

code, an algebraic code, and encodes the messages into codewords of the obfuscated code, subsequently adding a number of errors which are guaranteed to be correctable by the secret Goppa code. In a variant proposed by Niederreiter some years later [23], the generator matrix is replaced by the parity-check matrix and the message is encoded into syndrome vectors, thus achieving a significant reduction in the number of operations for encryption, at the cost of a moderate increase in the number of operations for decryption. After 40 years from its introduction, the original McEliece cryptosystem is still unbroken and, while requiring a scaling of its parameters [1], from its original proposal, it retains its security against quantum computing adversaries. However, cryptosystems adopting Goppa codes have some drawbacks. The most important one is the large size of the public key, in the order of hundreds of kilobytes to a megabyte range depending on the target security level. Wishing to reduce the key size, a valid alternative to Goppa codes is constituted by the adoption of Quasi-Cyclic (QC) codes. These 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. Therefore, it is sufficient to store only the first row of such matrices to preserve their whole representation. The adoption of QC codes with an underlying algebraic structure is the most convenient choice from an efficiency viewpoint; nonetheless, it exposes the system to cryptanalytic attacks [15]. Such vulnerabilities disappear when using random or pseudorandom codes (i.e., without any underlying algebraic structure); indeed, the use of Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) codes [8] or Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) codes [22] allows to provide proper security guarantees from a mathematical and cryptanalytical perspective as well as satisfactory key sizes from an engineering standpoint. Currently, these kinds of codes are employed in the design of both the LEDAcrypt [7, 9, 10] and the BIKE [2] cryptosystem, which were also evaluated in the framework of the initiative promoted by the U.S.A. National Institute of Standards and Technology (NIST) for the standardization of post-quantum cryptosystems [24].

LDPC and MDPC codes are usually decoded through iterative algorithms which, in contrast with the bounded distance decoders typically used with algebraic codes, are not characterized by a deterministic decoding radius. As a consequence, there is a need to estimate the Decoding Failure Rate (DFR) performance of these codes through numerical simulations. This is not an issue whenever relatively large failure rates are satisfactory, as occurs, for example, in wireless communications, but it becomes an issue in those applications which require extremely low failure rates, as it is the case of code-based cryptographic applications.

In code-based cryptography, a non-null DFR implies that a decryption action may fail even on a valid ciphertext and this exposes the cryptosystem to Chosen Ciphertext Attacks (CCAs) such as those documented in [14, 17, 29], which exploit the availability of a decryption oracle (queried with ciphertexts properly built employing the public key) to derive information on the secret structure of the underlying QC-LDPC/QC-MDPC code. The only way to avoid these kinds

of attacks, thus attaining Indistinguishability under Adaptively Chosen Cipher-text Attack (IND-CCA2) is to apply state-of-the-art constructions such as the ones introduced in [11, 19], which in turn require the failure rate of the under-lying code to be negligible in the security parameter. As a consequence, it can be shown that a cryptosystem with security parameter λ requires a DFR $\leq 2^{-\lambda}$, with $\lambda \geq 128$. Such values are clearly impossible to simulate, even with the most powerful computer (or cluster of computers). This makes extremely important to assess models which allow to estimate the behavior of decoders for QC-LDPC and QC-MDPC codes, even though in conservative conditions. In this paper, we tackle such topic, focusing on the Bit-Flipping (BF) decoding technique [16] which is commonly used in this application as it offers an excellent trade-off between error correction capability and computational complexity.

In short, a BF algorithm performs syndrome decoding through an iterative procedure, in which the bit locations where the received message is erroneous are estimated starting from the value of the received syndrome. More precisely, the decoder starts from an estimate of the error vector with all bits set to zero and applies a sequence of *iteration function*s, each of which evaluates whether or not to flip (i.e., change) the j-th bit of the estimated error vector. The flipping action is made depending on the result of a check on whether the number of un-satisfied parity-check equations involving the estimated error vector bit exceeds a predefined threshold. If a flip is performed, the value of the syndrome is up-dated to reflec the change in the estimate. The decoder stops when the updated syndrome value is equal to zero (indicating a decoding success), or a predefined maximum number of iterations is reached.

Depending on the strategy employed by the iteration function to update the syndrome, BF decoders are classified in two main groups: in-place algorithms and out-of-place algorithms. The distinguishing point between an in-place and an out-of-place iteration function lies on when the update of the syndrome value is executed. In the in-place iteration function, the syndrome is updated just after each test establishing if the j-th bit value in the estimated error vector should be flipped or not. In the out-of-place iteration function, instead, the syndrome is updated after the tests over all the bits in the estimated error vector are executed and the corresponding bit-flips are performed.

In this paper, we consider a simple variant of the in-place BF strategy, which consists in randomizing the order in which the estimated error positions are processed. This modification permits 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 cryptosystems matching the DFR figures of merit needed to provide IND-CCA2 guarantees. A preliminary version of this paper appeared in [6], where we showed that, employing well-established assumptions in coding theory, it is possible to develop a closed-form statistical model of such a decoder, by studying the worst-case execution at each iteration for the average QC-LDPC/QC-MDPC code performance. In this work we revise and expand the theoretical tractation of the subject, adding the demonstration of some core lemmas and propositions,that were omitted in [6]. Moreover, we

provide an extended experimental evaluation in which we analyze the effect of changing the BF decoder thresholds in both a single iteration decoder and a two iterations one.

2   Notation and Background

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 an entry on the i-th row and the j-th column will be denoted as $a_{i,j}$. A 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/indexes of the asserted elements in a sequence) and the Hamming weight of $\mathbf{a}$ will be reported as $\mathrm{Supp}(\mathbf{a})$ and $w_H(\mathbf{a})$, respectively. We will use $\mathcal{P}_n$ to denote the set of permutations of n elements, represented as bijections on the set of integers $\{0, \ldots, n-1\}$. Given a permutation $\pi \in \mathcal{P}_n$ 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. Given a vector $\mathbf{a} \in \mathbb{F}_2^n$, we will use $\pi(\mathbf{a})$ to denote the vector that is obtained by permuting each of the entries of $\mathbf{a}$ according to π. We will write $\pi \xleftarrow{\$} \mathcal{P}_n$ 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 $k = (n_0 - 1)p$, $n_0 \geq 2$, 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 $w_H(\mathbf{e}) = t$ asserted bits. The encryption algorithm outputs as a ciphertext the syndrome $\mathbf{c} = \mathbf{e}\mathbf{M}^\top \in \mathbb{F}_2^{1 \times r}$. 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 $w_H(\mathbf{e}) = t$ 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 feeds a *syndrome decoding* algorithm, which in turn yield the error vector and allows to recover the original plaintext $\mathbf{m}$ employing the generation matrix and the vector $\mathbf{c} - \mathbf{e}$.

The syndrome decoding procedure analyzed in this paper originated from the BF decoding strategy that was originally proposed by Gallager in [16]. The

original BF algorithm takes as input a syndrome $\mathbf{s} = (\mathbf{c} + \mathbf{e})\mathbf{H}^\top = \mathbf{e}\mathbf{H}^\top \in \mathbb{F}_2^{1 \times r}$ computed multiplying the codeword $\mathbf{c} \in \mathbb{F}_2^{1 \times n}$ corrupted by an unknown error vector $\mathbf{e} \in \mathbb{F}_2^{1 \times n}$ by the matrix obtained transposing the parity-check matrix of the code. The algorithm yields an estimate of the unknown error vector, that we denote as $\hat{\mathbf{e}}$, which is initially assumed to be a null vector. For each bit position in the error estimate (and correspondingly also in the unknown error vector), a decision about flipping or not the bit value at hand is taken on the ground of the *number of unsatisfied parity-check* (upc) equations in which such a bit value/position participates. Considering a generic error bit in position $0 \leq j \leq n - 1$, such a quantity is computed as:

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

Indeed, note that the constant term of the 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 $\mathbf{H}$. As a consequence, the number of unsatisfied parity-check equations in which the j-th bit participates can be obtained by summing the entries of the syndrome which are indexed by the set of positions in $\mathrm{Supp}(\mathbf{H}_{:,j})$. 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 bit value in the considered position of the estimated error vector is flipped and the syndrome is coherently updated by replacing its current value with $\mathbf{s} \oplus \mathbf{H}_{:,j}$. In a single decoding iteration, all error bits are evaluated following their positional order from 0 to $n - 1$. A syndrome decoder, which applies an in-place decoding strategy, executes multiple iterations, each of which repeats the steps previously described until either a null syndrome is obtained or a prefixed maximum number of iterations is reached.

When an out-of-place strategy is employed, every error bit is assessed relying on the same syndrome value provided (as input) at the beginning of the iteration, while the updates of both the error vector estimate 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 are able to provide a closed form estimate of the DFR in a worst-case scenario.

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

---

Algorithm 1: In-place, Randomized Bit-flipping (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: imax ≥ 1: maximum number of (outer loop) iterations;
  $\mathbf{b} = [\mathbf{b}_0, \ldots, \mathbf{b}_k, \ldots, \mathbf{b}_{\text{imax}-1}]$, with $\mathbf{b}_k \in \{\lceil \frac{v}{2} \rceil, \ldots, v\}$: flip thresholds

1 iter $\leftarrow 0$
2 $\hat{\mathbf{e}} \leftarrow \mathbf{0}_n$ // estimated error initialization
3 **while** (iter < imax) $\wedge$ ($\mathrm{w_H}(\mathbf{s}) > 0$) **do**

4 $\quad \pi \xleftarrow{\$} \mathcal{P}_n$ // random permut. of size n
5 $\quad$ **for** $j \leftarrow 0$ **to** $n - 1$ **do**
6 $\quad\quad \ell \leftarrow \pi(j)$ // permuted bit index
7 $\quad\quad$ upc $\leftarrow 0$ // integer value
8 $\quad\quad$ **for** $i \in \mathrm{Supp}(\mathbf{H}_{:,\ell})$ **do**
9 $\quad\quad\quad$ upc $\leftarrow$ upc $+ s_i$
10 $\quad\quad$ **if** upc $\geq \mathbf{b}_{\text{iter}}$ **then**
11 $\quad\quad\quad \hat{e}_\ell \leftarrow \hat{e}_\ell \oplus 1$ // error update
12 $\quad\quad\quad \mathbf{s} \leftarrow \mathbf{s} \oplus \mathbf{H}_{:,\ell}$ // syndrome update
13 $\quad$ iter $\leftarrow$ iter $+ 1$ // counter update

14 **return** $\{\hat{\mathbf{e}}, \mathbf{s}\}$

(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 imax and a vector $\mathbf{b}$ of length imax, such that its k-th entry, $\mathbf{b}_k$, with $0 \leq k \leq \text{imax} - 1$, 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 (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, $\mathbf{b}_{\text{iter}}$, chosen for the current outer loop iteration, then the value of the ℓ-th position of the estimated error vector, $\hat{\mathbf{e}}_\ell$, 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 $\mathbf{H}$ (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 `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 `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 `upc` values will be equal to zero, and, since the least functional threshold is strictly positive, the execution path controlled by the estimate-changing *if statement* at lines 10–12 is never taken.

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

We remark that, for the first outer-loop iteration of the decoding algorithm, 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 $(\mathbf{e} \oplus \hat{\mathbf{e}})\mathbf{H}^\top = \hat{\mathbf{e}}'\mathbf{H}^\top$.

Assumption 1. *The probability* $\mathrm{P}^{\mathtt{th}}_{f|1} = \Pr\left[\mathtt{upc}_j \geq \mathtt{th} \mid e_j \neq \hat{e}_j\right]$, *with $j \in \{0, 1, \ldots, n-1\}$, that the number of the unsatisfied parity checks involving the j-th bit of the error vector, i.e., $\mathtt{upc}_j$, is large enough to trigger a flip of $\hat{e}_j$, given that its current value does not match the value of the j-th bit in the unknown error vector, i.e., $\hat{e}'_j = e_j \oplus \hat{e}_j = 1$, is a function of only the total number $\hat{t} = \mathrm{w_H}(\hat{e} \oplus e)$ of positions over which the estimated error vector $\hat{e}$ and the unknown error vector e differ.*

Analogously, the probability $P_{m|0}^{\mathtt{th}} = \Pr\left[\mathtt{upc}_j < \mathtt{th} \mid e_j = \hat{e}_j\right]$ *that the number of the unsatisfied parity checks involving the j-th bit of the error vector, i.e., $\mathtt{upc}_j$, is low enough to maintain the current value $\hat{e}_j$ of the j-th estimated error vector bit, given that its current value matches the value of the j-th bit in the unknown error vector, i.e., $\hat{e}_j' = e_j \oplus \hat{e}_j = 0$, is a function of only the total number $\hat{t} = \mathrm{w_H}(\hat{e} \oplus e)$ of positions over which the estimated error vector $\hat{e}$ and the unknown error vector e differ.*

Informally, we are stating that the statistical behaviour of the single given $\mathtt{upc}_j$ does not depend on its location j, but only on the number of discrepancies between the estimated error vector and the actual one, and the fact that the j-th position of $\hat{\mathbf{e}}$ is in accordance or not with $\mathbf{e}$.

The following probabilities referred to flipping (f) or maintaining (m) the value of each bit of $\hat{e}$ will be used to characterize the iteration behaviour

$$P_{f|1}^{\mathtt{th}}(\hat{t}) = \Pr\left[\mathtt{upc}_j \geq \mathtt{th} \mid e_j \neq \hat{e}_j,\ \mathrm{w_H}(e \oplus \hat{e}) = \hat{t}\right],$$

$$P_{m|1}^{\mathtt{th}}(\hat{t}) = 1 - P_{f|1}^{\mathtt{th}}(\hat{t}) = \Pr\left[\mathtt{upc}_j < \mathtt{th} \mid e_j \neq \hat{e}_j,\ \mathrm{w_H}(e \oplus \hat{e}) = \hat{t}\right],$$

$$P_{m|0}^{\mathtt{th}}(\hat{t}) = \Pr\left[\mathtt{upc}_j < \mathtt{th} \mid e_j = \hat{e}_j,\ \mathrm{w_H}(e \oplus \hat{e}) = \hat{t}\right],$$

$$P_{f|0}^{\mathtt{th}}(\hat{t}) = 1 - P_{m|0}^{\mathtt{th}}(\hat{t}) = \Pr\left[\mathtt{upc}_j \geq \mathtt{th} \mid e_j = \hat{e}_j,\ \mathrm{w_H}(e \oplus \hat{e}) = \hat{t}\right].$$

To derive closed formulae for both $P_{f|1}$ and $P_{m|0}$, 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 $\mathtt{upc}$ at lines 8–9 is independent of the order in which they are added.

According to this, in the following we assume that each row of the parity-check matrix 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, and name a parity-check matrix (v, w)-regular if all its columns have weight v and all its rows have weight w. We share this assumption with a significant amount of literature on the prediction of the DFR of QC-LDPC decoders, ranging from the original work by Gallager on LDPCs [16, Section 4.2] to more recent ones, namely [32, Section 3] and [27, Section 4].

Formally, the following statement is assumed to hold:

Assumption 2. *Let H be a $p \times n_0 p$ quasi-cyclic block-circulant (v, w)-regular parity-check matrix and let s be the $1 \times p$ syndrome corresponding to a $1 \times n_0 p$ error vector $\hat{e}' = e \oplus \hat{e}$ that is modeled as a sample from a uniform random variable distributed over the elements in $\mathbb{F}_2^{1 \times n_0 p}$ with weight $\hat{t}$.*
We assume that each row $h_{i,:}$, $0 \leq i \leq p - 1$, of the parity-check matrix H is well modeled as a sample from a uniform random variable distributed over the elements of $\mathbb{F}_2^{1 \times n_0 p}$ with weight w.

Note that the assumption on the fact that the syndrome at hand is obtained from a vector $\hat{e}' = \hat{e} \oplus e$ of weight $\hat{t}$ is trivially true if the iteration of the decoder being considered is the first one being computed, since the error estimate $\hat{e}$ is null and the error vector e is drawn at random with weight $t = \hat{t}$. This in turn states that, when employing Assumption 2 in estimating the correction capability of the first iteration of a decoder, we are only relying on the fact that the rows of the matrix H can be considered independent, neglecting the effects of the quasi-cyclic structure.

In the following Lemma we establish how, upon relying on the previous assumption, the probabilities that characterize the choices on the bits of the estimated error vector, made by a either an in-place or an out-of-place iteration function, can be expressed.

Lemma 1. *Let H be a $p \times n_0 p$ quasi-cyclic block-circulant (v, w)-regular parity-check matrix; let $\hat{e}' = \hat{e} \oplus e$ be an unknown vector of length n and weight $\hat{t}$ such that $H(\hat{e}')^T = s$.*
From Assumption 1 and Assumption 2, the probabilities $\rho_{0,\mathrm{u}}(\hat{t}) = \Pr\left[s_i = 1 | \hat{e}'_z = 0\right]$ and $\rho_{1,\mathrm{u}}(\hat{t}) = \Pr\left[s_i = 1 | \hat{e}'_z = 1\right]$ that the i-th parity-check equation having $h_{i,z} = 1$, for any $0 \le z \le n-1$, is unsatisfied (i.e., $s_i = h_{i,:}(\hat{e}')^T = 1$) given the value of $\hat{e}'_z$, can be derived as follows

$$\rho_{0,\mathrm{u}}(\hat{t}) = \Pr\left[h_{i,:}(\hat{e}')^T = 1 \mid \hat{e}'_z = 0\right] = \frac{\sum_{l=1,\ l\ odd}^{\min\{w-1,\hat{t}\}} \binom{w-1}{l}\binom{n_0 p - w}{\hat{t}-l}}{\binom{n_0 p - 1}{\hat{t}}}$$

$$\rho_{1,\mathrm{u}}(\hat{t}) = \Pr\left[h_{i,:}(\hat{e}')^T = 1 \mid \hat{e}'_z = 1\right] = \frac{\sum_{l=0,\ l\ even}^{\min\{w-1,\hat{t}-1\}} \binom{w-1}{l}\binom{n_0 p - w}{\hat{t}-1-l}}{\binom{n_0 p - 1}{\hat{t}-1}}$$

Consequently, the probability $\mathrm{P}_{f|1}(\hat{t}) = \Pr\left[\mathtt{upc}_z \ge \mathtt{th} \mid \hat{e}'_z = \hat{e}_z \oplus e_z = 1\right]$ of changing (flipping) the z-th bit of the estimated error vector $\hat{e}_z$ assuming that $\hat{e}'_z = 1$, and the probability $\mathrm{P}_{m|0}(\hat{t}) = \Pr\left[\mathtt{upc}_z < \mathtt{th} \mid \hat{e}'_z = \hat{e}_z \oplus e_z = 0\right]$ of maintaining $\hat{e}_z$ assuming that $\hat{e}'_z = 0$, are computed as follows

$$\mathrm{P}_{f|1}^{\mathtt{th}}(\hat{t}) = \sum_{\sigma=\mathtt{th}}^{v} \binom{v}{\sigma}\left(\rho_{1,\mathrm{u}}(\hat{t})\right)^{\sigma}\left(1 - \rho_{1,\mathrm{u}}(\hat{t})\right)^{v-\sigma},$$

$$\mathrm{P}_{m|0}^{\mathtt{th}}(\hat{t}) = \sum_{\sigma=0}^{\mathtt{th}-1} \binom{v}{\sigma}\left(\rho_{0,\mathrm{u}}(\hat{t})\right)^{\sigma}\left(1 - \rho_{0,\mathrm{u}}(\hat{t})\right)^{v-\sigma}.$$

Proof. For the sake of brevity, we consider the case of $\hat{e}'_z = 1$ deriving the expression of $\mathrm{P}_{f|1}^{\mathtt{th}}(\hat{t})$; the proof for $\mathrm{P}_{m|0}^{\mathtt{th}}(\hat{t})$ can be carried out with similar arguments. Given a row $h_{i,:}$ of the parity-check matrix H, such that $z \in \mathrm{Supp}(h_{i,:})$, the equation $\bigoplus_{j=0}^{n_0 p-1} h_{i,j}\hat{e}'_j$ (in the unknown $\hat{e}'$) yields a non-null value for the i-th bit of the syndrome, $\hat{s}_i$, (i.e., the equation is unsatisfied) if and only if the support of $\hat{e}'$ is such that $\bigoplus_{j=0}^{n_0 p-1} h_{i,j}\hat{e}'_j = 2a + 1, a \ge 0$, including the term having $j = z$, i.e., $h_{i,z}\hat{e}'_z = 1$. This implies that the cardinality of the set obtained intersecting

the support of $h_{i,:}$ with the one of $\hat{e}'$, $\left|(\text{Supp}(h_{i,:}) \setminus \{z\}) \cap (\text{Supp}(\hat{e}') \setminus \{z\})\right|$, must be an even number, which in turn cannot be larger than the minimum between $\left|\text{Supp}(h_{i,:}) \setminus \{i\}\right| = w - 1$ and $|\text{Supp}(\hat{e}') \setminus \{i\}| = \hat{t} - 1$.

The probability $\rho_{1,\text{u}}(\hat{t})$ is obtained considering the fraction of the number of values of $\hat{e}'$ having an even number of asserted bits matching the asserted bits ones in a row of H (noting that, for the z-th bit position, both the error and the row of H are set) on the number of $\hat{e}'$ values having $\hat{t} - 1$ asserted bits over $n_0 p - 1$ positions, i.e., $\binom{n_0 p - 1}{\hat{t} - 1}$. The numerator of the said fraction is easily computed as the sum of all $\hat{e}'$ configurations having an even number $0 \le l \le \min\{w - 1, \hat{t} - 1\}$ of asserted bits. Considering a given value for l, the counting of $\hat{e}'$ values is derived as follows. Picking one vector with l asserted bits over w possible positions, i.e., one vector over $\binom{w-1}{l}$ possible ones, there are $\binom{n_0 p - w}{\hat{t} - 1 - l}$ possible values of the error vector exhibiting $\hat{t} - 1 - l$ null bits in the remaining $n_0 p - w$ positions; therefore, the total number of vectors with weigh l is $\binom{w-1}{l}\binom{n_0 p - w}{\hat{t} - 1 - l}$.
Repeating the same line of reasoning for each value of l allows to derive the numerator of the formula defining $\rho_{1,\text{u}}(\hat{t})$.

From Assumption 2, the observation of any unsatisfied parity check involving the z-th bit of the error vector $\hat{e}'_z$ (i.e., $h_{i,:}(\hat{e}')^T = 1$) given that $\hat{e}'_z = \hat{e}_z \oplus e_z = 1$, is modeled as a random variable with a Bernoulli distribution having parameter (or expected value) $\rho_{1,\text{u}}(\hat{t})$, and each of these random variables is independent of the others. Consequentially, the probability that the decoder performs a bit-flip of an element of the estimated error vector when the corresponding bit of the error vector is asserted and the counter of the unsatisfied parity checks (`upc`) is above or equal to a given threshold `th`, is derived as the binomial probability obtained adding the outcomes of v (column-weight of H) i.i.d. Bernoulli trials.

$\square$

Worst-case Iteration Scenario for the IR-BF Decoder. In the following 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 $\text{P}_{f|1}$ and $\text{P}_{m|0}$ as previously derived.

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 $w_H(\mathbf{e} \oplus \hat{\mathbf{e}}) = 0$.

Considering the error estimate at the beginning of the outer-loop iteration $\bar{\mathbf{e}}$ and the corresponding number of residual mismatched bit estimations $\bar{t} = w_H(\mathbf{e} \oplus \bar{\mathbf{e}})$, 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 $\hat{t}$.

We denote as π the permutation picked in line 4 of Algorithm 1 and as π^* an element of the subset $\mathcal{P}_n^*$ of permutations ($\mathcal{P}_n^* \subset \mathcal{P}_n$) such that

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

Let $\texttt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} \mid \pi \in \mathcal{P}_n\right)$ 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 $\texttt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} \mid \pi^* \in \mathcal{P}_n^*\right)$, 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 how the decoder reaches a correct decoding at the end of the outer loop, with the lowest probability.

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*

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

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 $\texttt{Prob}\left(\hat{\mathbf{e}} \neq \mathbf{e} \mid \pi \in \mathcal{P}_n\right) = 1 - \beta(\pi)$, 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

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

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.

Considering a generic permutation π, such that the support of $\pi(\mathbf{e})$ is $\{u_0, \cdots, u_{\bar{t}-1}\}$; we have

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

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

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

$\square$

A Worst-case DFR Estimate for the IR-BF Decoder.   We consider one outer loop iteration with $\bar{\mathbf{e}}$ being the error vector estimate before the beginning of the loop, exhibiting $\bar{t} = \mathrm{w_H}(\mathbf{e} \oplus \bar{\mathbf{e}})$ mismatches with the unknown error vector $\mathbf{e}$. From now on we will assume that, in each inner-loop 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 consider the following two sets bit-positions/indexes in the error vector estimate: $E_1 = S(\mathbf{e} \oplus \bar{\mathbf{e}})$ and $E_0 = \{0, \ldots, n-1\} \setminus E_1$. Denote with $\hat{t}_0$ the number of places/positions where the estimated error differs from the actual unknown $\mathbf{e}$ and the positions are also included in E_0. At the beginning of the outer loop iteration, we have

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

Analogously, denoting with $\hat{t}_1$ the number of positions in which the estimated error and the actual one differ and the positions are also included in E_1; at the beginning of the outer loop iteration, we have

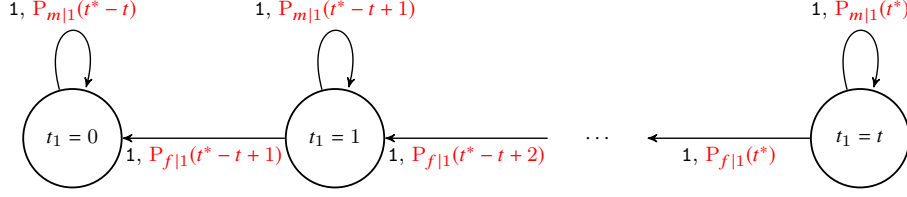$$\hat{t}_1 = |\mathrm{Supp}(\mathbf{e} \oplus \bar{\mathbf{e}}) \cap E_1| = \bar{t}.$$
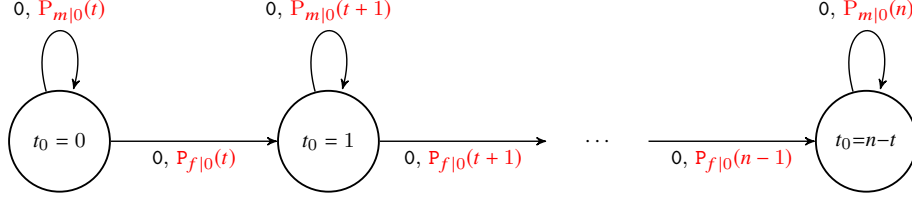
Fig. 1. Structure of the probabilistic FSA modeling the evolution of the distribution of the $\hat{t}_0$ variable. Read characters are reported in black, transition probabilities in red.

i) Let $\mathrm{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$ denote the probability that the decoder in Algorithm 1, starting from a state where $\mathrm{w_H}(\bar{\mathbf{e}} \oplus \mathbf{e}) = \omega$, 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) Let $\mathrm{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$ denote the probability that the decoder in Algorithm 1, starting from a state where $\mathrm{w_H}(\bar{\mathbf{e}} \oplus \mathbf{e}) = \omega$, 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) Let $\mathrm{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{i} x\right)$ be the probability that, starting from a state such that $\mathrm{w_H}(\bar{\mathbf{e}} \oplus \mathbf{e}) = \omega$, 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 $\mathrm{w_H}(\hat{\mathbf{e}} \oplus \mathbf{e}) = x$.

The expressions of the probabilities i) and ii) describe the statistical distributions of $\hat{t}_0$ and $\hat{t}_1$ after that all n iterations of the inner loop of Algorithm 1 have been executed and all the bits in the estimated error vector have been processed in the order pointed out by the permutation $\pi^* \in \mathcal{P}_n^*$, selected at the beginning of the outer loop. To show how the said distributions are computed and that they only depend on $\mathrm{P}_{f|1}(\hat{t})$ and $\mathrm{P}_{m|0}(\hat{t})$, respectively, we employ the framework of Probabilistic Finite State Automata (PFSA) [26].

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. A transition from a PFSA state to its subsequent one in the computation is computed taking, for each automaton state for which an admissible transition is present (i.e. the read symbol matches the one on the input tape), the probability mass related to the automaton state itself and, adding the product of the probability mass multiplied by the transition probability to the destination automaton state probability mass.

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 Fig. 1.

Fig. 2. 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.

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 $\hat{e}$ so that the $(n - t$, at the first iteration) positions with no discrepancy between $\bar{e}$ and 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 $\hat{e}$ equal to zero, and matching the one in the same position of e, causing a discrepancy. Because of Assumption 1, the probability of such a transition is $P_{f|0}(t+i)$, while the probability of the self-loop transition from $\hat{t}_0 = i$ to $\hat{t}_0 = i$ itself is $P_{m|0}(t+i)$.

Note that, during the inner loop iterations of the IR-BF decoder acting on positions of $\hat{e}$ which have no discrepancies, it is not possible to decrease the value $\hat{t}_0$, as no reduction on the number of discrepancies between $\hat{e}$ and e can be done changing values of $\hat{e}$ which are already equal to the ones in 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 efficiently 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 0) as the $(n - t + 1) \times (n - t + 1)$ matrix

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

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 $\mathbf{y} = [\texttt{Prob}\,(\hat{t}_0 = 0) = 1, \texttt{Prob}\,(\hat{t}_0 = 1) = 0, \texttt{Prob}\,(\hat{t}_0 = 2) = 0, \cdots, \texttt{Prob}\,(\hat{t}_0 = n - t) = 0]$, we compute $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$ as the $(x+1)$-th element of $\mathbf{yK}_0^{n-t}$.

We now model the distribution of $\hat{t}_1$, during the last $\hat{t}$ 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, $\tilde{t} = w_H\,(\hat{\mathbf{e}}')$, 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 of this second phase of the outer loop 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 $w_H\,(\mathbf{e} + \hat{\mathbf{e}})$.

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. 2. The initial distribution of the values of $\hat{t}_1$, constituting the initial state of the PFSA in Fig. 2, is such that $\texttt{Prob}\,(\hat{t}_1 = t) = 1$, corresponding to the $\hat{t}_1$-element vector $\mathbf{z} = [0, 0, \ldots, 0, 1]$. The transition matrix of the PFSA is

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

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

Having shown how to compute $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_0} x\right)$, and $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow{E_1} x\right)$, we proceed to show how they can be used together to derive $\texttt{Prob}_{\mathcal{P}_n^*}\left(\omega \xrightarrow[i]{} x\right)$ and express the worst case DFR after $\texttt{imax}$ iterations of the outer-loop of Algorithm 1, which we denote as $\mathrm{DFR}^*_{\texttt{itermax}}$.

First of all, it is easy to observe have

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

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

is: $\hat{t}^{(i)}$ corresponds to the number of residual errors after the i-th outer loop iteration.

As a consequence, by considering all possible configurations of such values, and taking into account that the first iteration begins with t residual errors, we have

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

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

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 $\hat{t}^{(\text{imax}-1)}$ residual errors. Taking this probability into account, the DFR after imax iterations is

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

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

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

A bonus point of the analysis we propose is that it is easy to obtain also an estimate for the average DFR of a single-iteration decoder, (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 $\mathrm{Supp}(\pi(\mathbf{e}))$. Let a_i, a_{i+1} be two consecutive elements of $\mathrm{Supp}(\pi(\mathbf{e}))$, with $0 \leq i \leq t-2$, 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

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

4   Validation of DFR Models and Cryptosystem Design

In this section we perform a numerical validation of the proposed analysis of IR-BF decoders and employ the our model to design practical sets of code parameters for QC-LDPC/QC-MDPC code-based cryptosystems employing a two-iteration IR-BF decoder. First, we consider the accuracy of the new model for a

single decoding iteration showing that it is fully accurate regardless of both the choice of the thresholds employed to perform the bit-flipping evaluations and the code parameters. Subsequently, we consider more than one decoding iteration and comment on the differences between the modeled and the experimental DFRs of Algorithm 1. In particular, we observe that differences between the proposed model and the experimental behavior arise only when the threshold values selected to perform the bit-flipping evaluations in Algorithm 1 are significantly larger than $v/2$, and provide qualitative justifications for such a phenomenon. Furthermore, we provide experimental evidence that when the values of the bit-flipping thresholds adopted by Algorithm 1 are close to $v/2$, the worst case model we developed for the IR-BF decoder well fits the behavior observed through numerical simulations.

4.1   Experimental validation of DFR models

We consider a QC-MDPC code having the-parity check matrix $\mathbf{H}$ formed by a row of $n_0 = 2$ circulant blocks of size $p = 4,801$ and column weight $v = 45$. The error correction capability of this code via IR-BF decoding has been assessed through Monte Carlo simulations. For this purpose, the IR-BF decoder has been implemented in `C99` while the whole simulation software has been compiled with the `GCC` 8.3.0 on a Debian GNU/Linux 10.2 (stable) operating system. Numerical simulations have been performed on a machine equipped with an Intel Core i5-6500 CPU running at 3.20 GHz.

The DFR has been estimated by varying the error vector weight t from 10 to 100, and generating and decoding $10^6 \approx 2^{20}$ randomly generated error vectors for each value of the weight. The bit-flipping threshold has been considered equal for all iterations. The DFR curves obtained through the said experimental simulations are compared with one obtained applying the theoretical worst case DFR estimate computed according to Section 3.2, which has been implemented employing the NTL library.[3]

Let us first consider a bit-flipping threshold $\mathtt{b} = 25$ ($\geq \lceil \frac{v}{2} \rceil = 23$). In Fig. 3 we report the results of numerical simulations of the DFR of the IR-BF decoder (*Sim.* datasets in Fig. 3) running for either one or two iterations (see DFR_1 and DFR_2, respectively). The DFR curves corresponding to a permutation selected among the worst case ones are also reported, for either one or two decoding iterations (see 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, thus 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 Fig. 3 (a), our technique for the DFR estimate provides a perfect match for the case of a single iteration. Indeed, our estimated worst-case

---

[3] NTL: A Library for doing Number Theory. https://www.shoup.net/ntl/
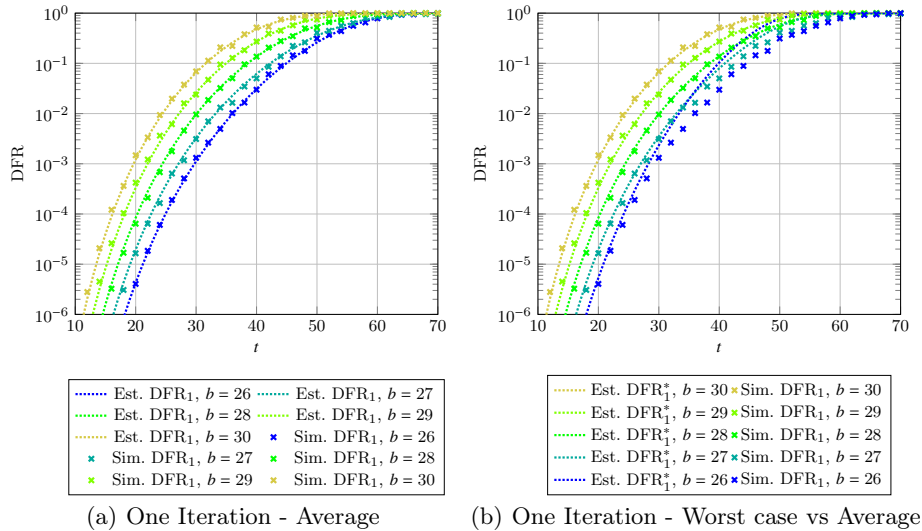
(a) One Iteration   (b) Two Iterations
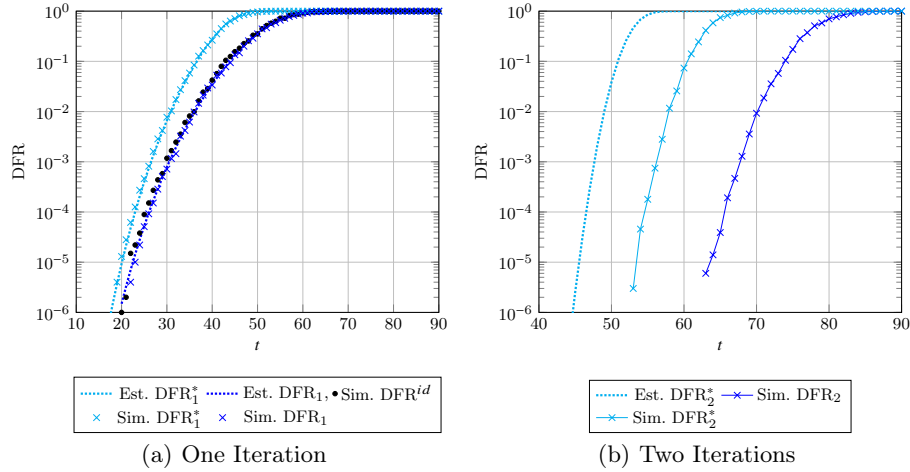
Fig. 3. Numerical validation of the DFR estimates (Est.) through numerical simulations (Sim.). The QC-MDPC code parameters are $n_0 = 2$, $p = 4801$ and $v = 45$. Figure (a) refers to the case of one decoding iteration (i.e., $\mathtt{imax} = 1$), figure (b) refers to a maximum number of decoding iterations 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.

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 observe that omitting the permutation before the first iteration has no practical impact on the DFR (black • symbols). Such a fact can be easily justified by the random nature of the error vector. Indeed, the discrepancies between the error estimate at the beginning of the decoding (when it is completely null) and the unknown error vector itself are already completely random. This fact can be observed looking at the black dots in Fig. 3 (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 a good match for the actual DFR itself. Considering the case of a two iterations IR-BF decoder, reported in Fig. 3 (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.

Let us now consider different choices for the bit-flipping threshold, namely, $\mathtt{b} \in \{26, 27, 28, 29, 30\}$. We first focus on a single decoding iteration and, again, compare the DFR values derived from the theoretical model with the ones ob-

(a) One Iteration - Average   (b) One Iteration - Worst case vs Average

Fig. 4. Numerical validation of the DFR estimates (Est.) through numerical simulations (Sim.), for one decoding iteration. The QC-MDPC code parameters are $n_0 = 2$, $p = 4801$ and $v = 45$. 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.

tained via numerical simulations; the corresponding results are reported in Fig. 4. For every threshold value, there is a tight match between the pair of DFR curves reporting theoretically estimated values and simulated values, respectively, for each value of the bit-flipping threshold. A noteworthy point is that, as the value of the employed threshold increases, the DFR values provided by our worst case modeling tends to coincide with those provided by an average case modeling.

A justification of this phenomenon can be provided analyzing the role of the threshold value b when the bit-flipping evaluations are performed. In the IR-BF decoder with a single iteration, a decoding failure can be caused either by *i)* flipping error estimate values where no discrepancy with the actual error is present or by *ii)* not flipping error estimate values which are in discrepancy with the actual error. The probability that the IR-BF outer-loop iteration flips (either wrongly or correctly) a bit value rapidly decreases as the value of the bit-flipping threshold b increases. As a consequence, in the decoder behaviour it is extremely unlikely that a non discrepant error estimate bit value is flipped (decoding failure cause i)), thus leading to the failures being caused preeminently by missed flips on discrepant error estimate values (decoding failure cause ii)). Since the non flipping actions on non discrepant error estimate values are taken with increasing probability as the decoding threshold is risen, the inner loop iteration of the decoder in which the said values are processed becomes less and less important. Indeed, as more and more of such non-flip decisions are
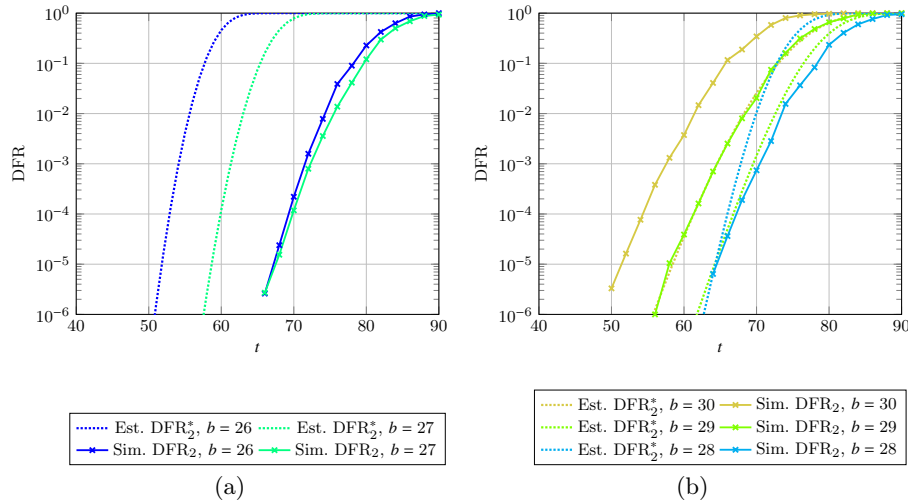
Fig. 5. Comparison between the worst case theoretical model for the DFR of a two iterations decoder, and the average simulated one. The QC-MDPC code parameters are $n_0 = 2$, $p = 4801$ and $v = 45$. The results marked with "Est." are obtained via the computation of closed formulas as opposed to the ones marked "Sim." which are the result of a numerical simulation.

correctly made, their effect on the number of residual discrepancies to be dealt by the decoder vanishes. Such a phenomenon is well represented by both our model and the simulation results reported in Figure 4, where it can be seen that the worst-case and average-case behavior of the decoder for higher values of b becomes more and more similar, becoming substantially the same for $b = 30$.

We also note that, whenever the threshold is risen, the likelihood of the BF decoding missing a flip on discrepant location of the error estimate raises, therefore leading to an increased number of decoding failures (assuming all the remaining code parameters and the weight of the error are unchanged). This is also evident in Figure 4. A phenomenon analogous to the one observed in the single iteration case takes place in the case of a multi-iteration decoder. Indeed, our proposed model yields a worst case estimate for the DFR, as reported in Figure 5(a) up to a threshold of $b = 27$. When considering higher threshold values, as the ones depicted in Figure 5(b) where larger threshold values starting at $\mathtt{b} = 28$ are employed, we have that the effect of decoding failures being determined by a larger amount of missed flipping actions on discrepant error estimate locations is amplified in a two iteration decoder. This in turn leads, for larger threshold values to a predicted worst-case DFR value which is lower than the actual simulated one. We note that this phenomenon can be practically counteracted picking low thresholds for the IR-BF decoder iterations, thus having it work in the regime where our DFR analysis provides conservative results.

Table 1. Cryptosystem parameters; DFR=$2^{-\lambda}$, λ={128, 192, 256}

Security Level	v	t	Two out-of-place iterations [27]		1st iter. IR-BF + 2nd out-of-place iter. [7]		Two iter.s IR-BF
			p	τ	p	τ	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

4.2   Design of Code-based Cryptosystems

We can employ the presented DFR model 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 a key size equal to 128, 192, or 256 bits. 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 **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 [4]. In the case of a Niederreiter-based key encapsulation mechanism (KEM), 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-bit long.

   Table 1 compares the parameter sets for the two-iterations out-of-place decoder proposed in [27], a decoder obtained with one iteration of the IR-BF decoder, followed by one iteration of the out-of-place LEDAcrypt decoder [7], 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-iterations IR-BF decoder. In the table, the values of τ refer 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 [27].

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 [32] and has then been improved in [27]. With similar arguments, an assumption-free, conservative upper bound for the DFR of a single decoder iteration was derived in [28]. However, employing such approaches to design the secret code parameters results in impractically large public-key sizes. To obtain keys with smaller size, in [3, 5, 27, 32] the DFR of a two-iterations 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 [31], 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 [30]. 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 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 [30], are one of the prime objects of study to determine the location of the error floor region [18, 25].

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 [30] will provide cryptosystem parameters which are not matching the DFR needed in IND-CCA2 constructions. Thus, we believe that relying on DFR curve extrapolations may provide overly optimistic cryptosystem parameter designs [12, 13].

In [30], 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 on 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 have obtained 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

In this work, we presented a closed form analysis of the error correction capability of a randomized variant of the classic in-place bit flipping decoder. Considering this randomized variant allowed us to provide closed form worst case DFR estimates, for the said bit-flipping decoder, allowing its evaluation even in regimes where the actual DFR cannot be derived by numerical simulations. This in turn allowed us to provide sound parameters for code-based cryptosystems relying on QC-LDPC/QC-MDPC codes, where providing extremely low DFRs is a requirement to achieve IND-CCA2 security guarantees. Our worst case analysis provides a perfect match of the IR-BF decoder behaviour when considering single iteration decoders, and a conservative bounding of the DFR in case a two-iteration IR-BF decoder is employed with a threshold close to the one of a canonical majority decoder. We foresee, as a interesting future research direction, the analysis of the emerging phenomena in two-iteration IR-BF decoders, whenever they are operating with thresholds far from the majority one.

References

1. Albrecht, M.R., Bernstein, D.J., Cid, C., Gilcher, J., Lange, T., Maram, V., von Maurich, I., Misoczki, R., Niederhagen, R., Paterson, K.G., Persichetti, E., Peters, C., Schwabe, P., Sendrier, N., Szefer, J., Tjhai, C.J., Tomlinson, M., Wang, W.: Classic McEliece website. https://classic.mceliece.org (2020)
2. Aragon, N., Barreto, P.S.L.M., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Ghosh, S., Gueron, S., Güneysu, T., Melchor, C.A., Misoczki, R., Persichetti, E., Sendrier, N., Tillich, J.P., Vasseur, V., Zémor, G.: BIKE website. https://bikesuite.org (2020)
3. Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., Santini, P.: LEDAkem: A post-quantum key encapsulation mechanism based on QC-LDPC codes. In: Lange, T., Steinwandt, R. (eds.) Post-Quantum Cryptography - 9th Int.'l Conference, PQCrypto 2018, Fort Lauderdale, April 9-11, 2018. Lecture Notes in Computer Science, vol. 10786, pp. 3–24. Springer (2018)
4. Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., Santini, P.: A finite regime analysis of information set decoding algorithms. Algorithms **12**, 209 (2019)
5. Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., Santini, P.: LEDAcrypt: QC-LDPC code-based cryptosystems with bounded decryption failure rate. In: Baldi, M., Persichetti, E., Santini, P. (eds.) Code-Based Cryptography. CBC 2019. Lecture Notes in Computer Science, vol. 11666, pp. 11–43. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-25922-8_2
6. Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., Santini, P.: A failure rate model of bit-flipping decoders for QC-LDPC and QC-MDPC code-based cryptosystems. In: Samarati, P., di Vimercati, S.D.C., Obaidat, M.S., Ben-Othman, J. (eds.) Proceedings of the 17th International Joint Conference on e-Business and Telecommunications, ICETE 2020 - Volume 2: SECRYPT, Lieusaint, Paris, France, July 8-10,

2020. pp. 238–249. ScitePress (2020). https://doi.org/10.5220/0009891702380249, https://doi.org/10.5220/0009891702380249

7. Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., Santini, P.: LEDAcrypt website. https://www.ledacrypt.org (2020)

8. Baldi, M., Chiaraluce, F., Garello, R., Mininni, F.: Quasi-Cyclic Low-Density Parity-Check Codes in the McEliece Cryptosystem. In: Proceedings of IEEE International Conference on Communications, ICC 2007, Glasgow, Scotland, UK, 24-28 June 2007. pp. 951–956 (2007). https://doi.org/10.1109/ICC.2007.161

9. Barenghi, A., Pelosi, G.: A comprehensive analysis of constant-time polynomial inversion for post-quantum cryptosystems. In: Palesi, M., Palermo, G., Graves, C., Arima, E. (eds.) Proceedings of the 17th ACM International Conference on Computing Frontiers, CF 2020, Catania, Sicily, Italy, May 11-13, 2020. pp. 269–276. ACM (2020). https://doi.org/10.1145/3387902.3397224

10. Barenghi, A., Pelosi, G.: Constant weight strings in constant time: a building block for code-based post-quantum cryptosystems. In: Palesi, M., Palermo, G., Graves, C., Arima, E. (eds.) Proceedings of the 17th ACM International Conference on Computing Frontiers, CF 2020, Catania, Sicily, Italy, May 11-13, 2020. pp. 132–141. ACM (2020). https://doi.org/10.1145/3387902.3392630

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

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

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

14. Fabšič, T., Hromada, V., Stankovski, P., Zajac, P., Guo, Q., Johansson, T.: A reaction attack on the QC-LDPC McEliece cryptosystem. In: Lange, T., Takagi, T. (eds.) PQCrypto 2017, pp. 51–68. Springer (2017)

15. Faugère, J.C., Otmani, A., Perret, L., Tillich, J.P.: Algebraic cryptanalysis of McEliece variants with compact keys. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 6110, pp. 279–298. Springer (2010). https://doi.org/10.1007/978-3-642-13190-5_14

16. Gallager, R.G.: Low-Density Parity-Check Codes. Ph.D. thesis, M.I.T. (1963)

17. Guo, Q., Johansson, T., Stankovski, P.: A key recovery attack on MDPC with CCA security using decoding errors. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Lecture Notes in Computer Science, vol. 10031, pp. 789–815. Springer (2016)

18. Hashemi, Y., Banihashemi, A.H.: 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 (2019)

19. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) Theory of Cryptography. TCC 2017. Lecture Notes in Computer Science, vol. 10677, pp. 341–371. Springer (2017). https://doi.org/10.1007/978-3-319-70500-2_12

20. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds.) Complexity of Computer Computations. pp. 85–103. The IBM Research Symposia Series, Springer, Boston, MA (1972)

21. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. Deep Space Network Progress Report **44**, 114–116 (Jan 1978)

22. Misoczki, R., Tillich, J.P., Sendrier, N., Barreto, P.L.: MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In: Proceedings of IEEE International Symposium on Information Theory (ISIT 2013), Istanbul, Turkey, 7-12 July 2013. pp. 2069–2073 (2013). https://doi.org/10.1109/ISIT.2013.6620590

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

24. NIST: Post-quantum cryptography. https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization

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

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

27. Santini, P., Battaglioni, M., Baldi, M., Chiaraluce, F.: Hard-decision iterative decoding of LDPC codes with bounded error rate. In: Proceedings of IEEE Conference on Communications (ICC 2019), Shanghai, China, 20-24 May 2019 (2019). https://doi.org/10.1109/ICC.2019.8761536

28. Santini, P., Battaglioni, M., Baldi, M., Chiaraluce, F.: A theoretical analysis of the error correction capability of LDPC and MDPC codes under parallel bit-flipping decoding and application to cryptography. IEEE Trans. Commun. **68**(8), 1017–1033 (2020)

29. Santini, P., Battaglioni, M., Chiaraluce, F., Baldi, M.: Analysis of reaction and timing attacks against cryptosystems based on sparse parity-check codes. In: Code-Based Cryptography. CBC 2019. Lecture Notes in Computer Science, vol. 11666, pp. 115–136. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-25922-8_7

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

31. Sendrier, N., Vasseur, V.: On the decoding failure rate of QC-MDPC bit-flipping decoders. In: Ding, J., Steinwandt, R. (eds.) PQCrypto 2019. Lecture Notes in Computer Science, vol. 11505, pp. 404–416. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-25510-7_22

32. Tillich, J.: The decoding failure probability of MDPC codes. In: Proceedings of IEEE International Symposium on Information Theory (ISIT 2018), Vail, CO, USA, 17-22 June 2018. pp. 941–945 (2018). https://doi.org/10.1109/ISIT.2018.8437843