# MCELIECE CRYPTOSYSTEM BASED ON MODERATE DENSITY PARITY-CHECK CODES

DANIEL CLEMENTE CENTENO

*Universidad pedagógica y tecnología colombia*

HENRY CHIMAL-DZUL

*University of Notre Dame, Notre Dame 46556, IN, USA*

## PROJECT DESCRIPTION

Modern cryptosystems based on integer factorization or the discrete logarithm problem are vulnerable to attacks by quantum computers. New cryptosystems are required to resist these future computers or to strengthened information security in current devices. In 2016, the National Institute for Standards (NIST) made a call for proposal and after 7 years of rigorous evaluations, the McEleice cryptosystem and its variant using Quasi-Cyclic Moderate Density Parity-Check codes are among the finalists in the NIST post-quantum competition. In this project we will focus in learning the mathematical aspects of the McEliece cryptosystem and its variant, which are presented in the paper [1]. Notes based our discussions will be written and our contribution in this project will be to construct various examples to illustrate concepts and results in order to make the material accessible to a broad mathematical audience. In some cases we are planning on providing proofs to facts stated without proof in [1].

[1] R. Misoczki, J. -P. Tillich, N. Sendrier and P. S. L. M. Barreto, "MDPC-McEliece: New McEliece variants from Moderate Density Parity-Check codes," 2013 IEEE International Symposium on Information Theory, Istanbul, Turkey, 2013, pp. 2069-2073, doi: 10.1109/ISIT.2013.6620590.

*E-mail addresses*: daniel.centeno@uptc.edu.co, hchimald@nd.edu.

## 1. Introduction to Coding Theory

Coding Theory is a branch of Mathematics born in the late 40's with the works of Claude Shannon and Richard Hamming (add references). The main object of study in Algebraic Coding Theory are linear error-correcting codes. These can be broadly understood as a finite dimensional subspace of a finite dimensional vector space over a finite field. The main goal of this section is to give an introduction to Algebraic Coding Theory.

1.1. **Finite Fields.** Recall that a *field* is is a set $F$ with two binary operations, addition "$+$" and multiplication "$\cdot$", such that $(F,+)$ and $(F^*,\cdot)$ are abelian groups, where $F^*$ is the set of all nonzero elements of $F$. A *finite field* is a field with a finite number of elements.

The following result will provide an infinite number of examples of finite fields.

**Lemma 1.** *Let $n \geq 1$ be an integer and consider the set $\mathbb{Z}_n$ of integers modulo n. Then $\mathbb{Z}_n$ is a finite field if and only if n is a prime number.*

The finite field $\mathbb{Z}_p$ is often written as $\mathbb{F}_p$ and the elements of $\mathbb{F}_p$ can be considered to be $\{0,1,2,\ldots,p-1\}$. We follow this notation throughout these notes.

**Example 2** (The binary field)**.** The *binary field* is the finite field $\mathbb{F}_2 = \{0,1\}$. The operations in $\mathbb{F}_2 = \{0,1\}$ are addition and multiplication modulo 2, which are given in the next tables:

| $+$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |

| $\cdot$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |

**Example 3.** The set $\mathbb{F}_3$ with the operations of addition and multiplication modulo 3 is a finite field with 3 elements. The operations are given next:

| $+$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 1 | 2 |
| 1 | 1 | 2 | 0 |
| 2 | 2 | 0 | 1 |

| $\cdot$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 2 | 1 |

Finite fields other than $\mathbb{F}_p$ can be constructed using polynomial rings and maximal ideals in that ring. The following result characterizes (up to isomorphism) all finite fields. The interested reader is addressed to (add references) for more details.

**Theorem 4.** *Let $f(x) \in \mathbb{F}_p[x]$ be an irreducible polynomial in $\mathbb{F}_p[x]$. Then the quotient ring*

$$\mathbb{F}_{p^m} = \frac{\mathbb{F}_p[x]}{\langle f(x)\rangle}$$

*is a finite field with $p^m$ elements where $m = \deg(f(x))$ and $\langle f(x)\rangle$ is the principal ideal of $\mathbb{F}_q[x]$ generated by $f(x)$. Moreover, if $\mathbb{F}$ is a finite field then $\mathbb{F}$ has $p^m$ elements (for some prime p and integer $m \geq 1$) and $\mathbb{F}$ is isomorphic to $\mathbb{F}_{p^m}$.*

**Example 5.** A finite field with 4 elements can be constructed as

$$\mathbb{F}_{2^2} = \frac{\mathbb{F}_2[x]}{\langle f(x)\rangle},$$

where $f(x)$ is an irreducible polynomial over $\mathbb{F}_2$ with $\deg(f(x)) = 2$. The only irreducible polynomial of degree 2 over $\mathbb{F}_2$ is $f(x) = x^2 + x + 1$. Hence

$$\frac{\mathbb{F}_2[x]}{\langle f(x)\rangle} = p(x) + x^2 + x + 1 = \{ax + b + \langle f(x)\rangle \ : \ a, b \in \mathbb{F}_2\}.$$

If $\alpha = x + \langle f(x)\rangle$, then $\mathbb{F}_4 = \{0, 1, \alpha, \alpha + 1\}$. The Cayley tables for addition and multiplication are as follows.

| $+$ | $0$ | $1$ | $\alpha$ | $\alpha+1$ |
|---|---|---|---|---|
| $0$ | | | | |
| $1$ | | | | |
| $\alpha$ | | | | |
| $\alpha+1$ | | | | |

| $\cdot$ | $0$ | $1$ | $\alpha$ | $\alpha+1$ |
|---|---|---|---|---|
| $0$ | | | | |
| $1$ | | | | |
| $\alpha$ | | | | |
| $\alpha+1$ | | | | |

1.2. **The Hamming Space.** Let $n \geq 1$ be an integer, $\mathbb{F}_q$ be a finite field and

$$\mathbb{F}_q^n = \{(v_1, v_2, \ldots, v_n) \ : \ v_i \in \mathbb{F}_q, \ 1 \leq i \leq n\}.$$

Observe that $\mathbb{F}_q^n$ consists of all $n$-tuples with entries from the finite field $\mathbb{F}_q$. When this set is equipped with the usual operations of vector addition and scalar multiplication, $\mathbb{F}_q^n$ becomes a vector space over $\mathbb{F}_q$ of dimension $n$.

The *Hamming weight* of $v = (v_1, v_2, \ldots, v_n) \in \mathbb{F}_q^n$ is defined as the number of non-zero entries of $v$, i.e.,

$$w_H(v) = |\{i \ : \ 1 \leq i \leq n, v_i \neq 0\}|$$

The *Hamming distance* between $u, v \in \mathbb{F}_q^n$ is defined as the number of coordinates on which $u$ and $v$ differ, i.e.,

$$d_H(u, v) = w_H(u - v).$$

The Hamming distance satisfies the properties of a metric. That is, for all $, v, w \in \mathbb{F}_q^n$

(1) (Non-negativity) $d_H(u, v) \geq 0$, and $d_H(u, v) = 0$ if and only if $u = v$.
(2) (Symmetry) $d_H(u, v) = d_H(v, u)$.
(3) (Triangle inequality) $d_H(u, v) \leq d_H(u, w) + d_H(w, v)$.

Therefore, $(\mathbb{F}_q^n, d_H)$ is a metric space, known as the *Hamming space*.

1.3. **Linear codes.** Let $n \geq 1$ be an integer and $\mathbb{F}_q$ be a finite field. An $[n, k]$ *linear code over* $f\mathbb{F}_q$ is a $k$-dimensional subspace of $\mathbb{F}_q^n$. We refer to $n$ and $k$ as the *length* and *dimension* of the code, respectively. Along with these two parameters, the quality of an $[n, k]$ linear code $\mathscr{C}$ over $\mathbb{F}_q$ is measured by a third one, namely the *the minimum Hamming distance* of the code, which is defined as

$$d_H(\mathscr{C}) = \min\{d_H(u, v) \ : \ u, v \in \mathscr{C}, u \neq v\}.$$

Since $\mathscr{C}$ is a subspace, it can be shown that

$$d_H(\mathscr{C}) = \min\{w_H(u) \ : \ u \in \mathscr{C}, u \neq 0\}.$$

The number $w_H(\mathscr{C}) = \min\{w_H(u) \ : \ u \in \mathscr{C}, u \neq 0\}$ is called the *minimum weight* of the code $\mathscr{C}$. Thus, the previous relation can be restated as: for a linear code $\mathscr{C}$, its minimum Hamming distance coincides with its minimum Hamming weight. Of

course, computing the minimum weight of a code requires less effort than computing the minimum Hamming distance. This is one advantage of linear codes over non-linear codes.

An $[n,k,d]$ *linear code* $\mathscr{C}$ is an $[n,k]$ linear code with $d_H(\mathscr{C}) = d$. It should be noted that the three parameters do not define a code uniquely.

A $[n,k]$ linear code $\mathscr{C}$ over $\mathbb{F}_q$ can be completely specified by one of its basis. A *generator matrix* of $\mathscr{C}$ is defined as an $k \times n$ matrix whose rows form a basis for $\mathscr{C}$. A second common way to specify a linear code is by describing a basis of its orthogonal complement. Recall that on the $n$-dimensional vector space $\mathbb{F}_q^n$ we define the *dot product* or *scalar product* of $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$ as

$$u \cdot v = u_1 v_1 + \cdots + u_n v_n \in \mathbb{F}_q.$$

Two vectors $u, v \in \mathbb{F}_q^n$ are said to be orthogonal if $u \cdot v = 0$. The *orthogonal complement* of a subspace $V$ of $\mathbb{F}_q^n$ is defined as

$$V^{\perp} = \{u \in \mathbb{F}_q^n \,:\, u \cdot v = 0 \ \ \forall v \in V\}.$$

An important result concerning s subspace and its orthogonal complement is the following.

**Theorem 6.** *Let $V$ a subspace of $\mathbb{F}_q^n$ of dimension $k$. Then $(V^{\perp})^{\perp} = V$ and*

$$\dim_{\mathbb{F}_q}(V) + \dim_{\mathbb{F}_q}(V^{\perp}) = n.$$

A *parity-check matrix* of an $[n,k]$ linear code $\mathscr{C}$ over $\mathbb{F}_q$ is an $(n-k) \times n$ matrix $H$ such that

$$GH^t = 0,$$

where $G$ is a generator matrix of $\mathscr{C}$. In other words, the rows of $H$ form a basis of $C^{\perp}$. The $[n, n-k]$ linear code $\mathscr{C}^{\perp}$ is called the *dual code* of $\mathscr{C}$ and $H$ is one of its generator matrices. The following result shows how to compute the parity check matrix from the generator matrix.

**Proposition 7.** *Let $\mathscr{C}$ be an $[n,k]$ linear code over $\mathbb{F}_q$. The matrix $G = [I_k \mid A]$ is a generator matrix of $\mathscr{C}$ if and only if $H = [-A^t \mid I_{n-k}]$ is a parity-check matrix of $\mathscr{C}$.*

The generator matrix $G = [I_k \mid A]$ of $\mathscr{C}$ is said to be in *standard form*. It is important to note that not every code has a generator matrix in standard form, but every generator matrix can be turned into a one in standard form by doing elementary row operations and column permutations. That is, if $G$ is a generator matrix of an $[n,k]$ linear code over $\mathbb{F}_q$, then there exists a matrix $S \in GL_k(\mathbb{F}_q)$ and a permutation matrix $P \in Mat_{n\times n}(\mathbb{F}_q)$ such that

$$G' = [I_k \mid A] = SGP.$$

A parity check matrix $H$ of $\mathscr{C}$ such that $GH^t = 0$ can be shown to be

$$H = [-A^t \mid I_{n-k}]P^{-1}.$$

An interesting application of the parity-check matrix of a code is that the minimum distance can be computed by knowing the linear independence relationship between its columns. More precisely, we have:

**Proposition 8.** *Let $\mathscr{C}$ be an $[n,k]$ linear code over $\mathbb{F}_q$ and $H$ a parity-check matrix of $\mathscr{C}$. Then $d = d_H(\mathscr{C})$ if and only if every set of $d-1$ columns of $H$ is linearly independent but there is a set of $d$ linearly dependent columns of $H$.*

**Example 9.** Let $\mathscr{C}$ be the linear code over $\mathbb{F}_2$ with generator matrix

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Elementary row operations transform $G$ into the matrix

$$\widetilde{G} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Applying the permutation $\pi = ()$ to the columns of $\widetilde{G}$ gives

$$G' = [I_3 \mid A] = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

Therefore

$$H' = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

Applying the inverse of $\pi$ to the columns of $H'$ we obtain

$$H = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

, which is a parity-check matrix of $\mathscr{C}$. Lastly, since the rows of $G$ are elements of the code $\mathscr{C}$, we know that $d_H(\mathscr{C}) \leq 2$. By Proposition 8, the minimum Hamming distance is the minimum number of positions in which two distinct codewords differ. In this case, $(d_H(C) \leq 2$ indicates that the minimum Hamming distance is at most 2.

## 2. Quasi-cyclic Moderate Density Parity-Check Codes

This section focuses on the class of Moderate Density Parity-Check codes (MDPC) which are also quasi-cyclic (QC). Broadly speaking, this family of binary codes can be defined by giving a parity-check matrix $H$ which is block circulant with the property that every row of $H$ has a moderate density of 1's. To be ore precise, we firts introduce the family of quasi-cyclic codes and then the family of QC-MDPC codes.

### 2.1. **Quasi-cyclic codes.**

**Definition 10.** If $C \subset F_q^n$ a $(n,k)$-linear code. It is stated that $C$ is cyclic if if the following property is satisfied:

$$\forall c_0 \cdots c_{n-1} \in C, c_{n-1} c_0 \cdots c_{n-2} \in C.$$

we note that if $C$ is a cyclic code, then given $c_0 \cdots c_{n-1} \in C$, the words $c_{n-1} c_{0n-2}$ , $c_{n-2} c_{n-1} c_{0n-3}$ are also in C.

On the other hand, we can also characterize cyclic codes by looking at what happens at the base of the code and also by its structure. For this there are the following two characterizations.

**Proposition 11.** *First characterization of the cyclic codes: If $C \in F_q^n$ a $(n,k)$ linear code with base $B = \{x_1, \cdots x_k\}$. then, C is cyclic if and, only if,$\forall x_i \in B$, with $i = 1, \cdots, k$, it follows that, $x_{in-1}x_{i0in-2} \in C$, where $x_i = x_{i0in-2}x_{in-1}$.*

**Example 12.** If $C \subseteq F_2^7$ an linear code the generating matrix of which is given by:

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

Then, a base of C is $B = 1101000, 0110100, 0011010, 0001101$, We can see that the first three are in C, since:

$$1101000 \xrightarrow{+} 0110100$$
$$0110100 \xrightarrow{+} 0011010$$
$$0011010 \xrightarrow{+} 0001101$$

i.e., the three previous ones are translations, but the last one is not.

$$0001101 \xrightarrow{+} 1000110$$

We can see that by translation it seems that it is not from C, but we can use properties of the field where we are to see that:

$$1000110 = 1101000 + 0110100 + 0011010.$$

So this word belongs to the base of $C$. Then $C$ is a cyclic code.

**Proposition 13.** *Second characterization of the cyclic codes: If $C \subseteq F_q^n$ cyclic code. Then, C is cyclic if and only if , $C(x)$ is an ideal of $F_q \setminus [x]\,(x^n - 1)$.*

To determine what the generator matrices of these codes are like, we must first determine what their generator polynomial is.

**Proposition 14.** *If $C \subseteq F_q^n$ a cyclic code. Then, exthere is a single monic polynomial $g(x)$ of a minimum degree such that.*

$$C(x) = (\overline{g(x)}) = \overline{t(x)g(x)} \in F_q \setminus [x]\,(x^n - 1) | t(x) \in F_q^n.$$

*where, $g(x)$ is a factor in $x^n - 1$ in $F_q[x]$*

**Example 15.** Let's calculate the codes of length 7 in $F_2$, then let us first determine their irreducible factors over $F_2$ at $x^7 - 1$. Then:

$$x^7 - 1 = (x^3 + x + 1)(x^3 + x^2 + 1)(x + 1)$$

**Proposition 16.** *Generator Matrix: Let $C \subseteq F_q^n$ be a cyclic code with a generator polynomial $g(x) = \sum_{i=0}^{n-k} g_i x^i$ of degree $n-k$. In such a case, C is a code of dimension k, and its generator matrix is expressed as follows.*

$$\begin{pmatrix} g_0 & g_1 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & g_0 & g_1 & \cdots & g_{n-k} \end{pmatrix}$$

**Example 17.** If we consider $C_3$ as the cyclic binary code of length 7 with the generator polynomial $g_3(x) = x^3 + x + 1$, then, applying the above, the generator matrix of $C_3$ is expressed as follows.

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

then:

$$1 \cdot g(x) = 1101000$$
$$x \cdot g(X) = 0110100$$
$$x^2 \cdot g(x) = 0011010$$
$$x^3 \cdot g(x) = 00011001$$

**Proposition 18.** *Control matrix: If $C \subset F_q^n$ a cyclic code with the control polynomial $h(x) = \sum_{i=0}^{k} h_i x^i$ of degree k. Then, a control of matrix the C is:*

$$\begin{pmatrix} h_k & h_{k-1} & \cdots & h_0 & 0 & \cdots & 0 \\ 0 & h_k & h_{k-1} & \cdots & h_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_k & h_{k-1} & \cdots & h_0 \end{pmatrix}$$

2.2. **QC-MDPC codes.** Quasi-cyclic codes are of great importance due to the limitation of cyclic codes of extension $q$, which cannot have a length greater than $q$. On the contrary, quasi-cyclic codes are based on modular structures in order to append multiple cyclic codes and thereby increase the length as needed.

**Note 1.** :A cyclic code can be seen as a particular case of quasi-cyclic codes when the length $l = 1$.

**Definition 19.** A quasi-cyclic code is a type of linear code in which cyclically shifting a codeword by a fixed number $n_0 \neq 1$ (or a multiple of $n_0$) of symbol positions, either to the right or to the left, results in another codeword. It is evident that when $n_0 = 1$, a quasi-cyclic code becomes a standard cyclic code. The integer $n_0$ is referred to as the shifting constraint. Additionally, it is noteworthy that the dual code of a quasi-cyclic code is also quasi-cyclic.

**Example 20.** Consider the $(9,3)$ code generated by the following generator matrix:

$$G = \begin{pmatrix} 111 & 100 & 110 \\ 110 & 111 & 100 \\ 100 & 110 & 111 \end{pmatrix}$$

Now, to see what a quasi-cyclic code is, we must list all of its words, which are in the following table: Suppose that we move the word fifth $(001011010)$ three positions to the right, i.e:

$$(001011010) \overset{3+}{\rightarrow} (010001011)$$

$$\begin{array}{ccc} 000 & 000 & 000 \\ 111 & 100 & 110 \\ 100 & 111 & 100 \\ 100 & 110 & 111 \\ 001 & 011 & 010 \\ 011 & 010 & 001 \\ 010 & 001 & 011 \\ 101 & 101 & 101 \end{array}$$

which is the seventh word, we see that it is equal to the cyclic codes, now if we assume the fifth word first one position to the right and then two respectively we see that they are not words, since:

$$(001011010) \overset{1+}{\rightarrow} (000101101)$$

$$(001011010) \overset{2+}{\rightarrow} (100010110)$$

Clearly, these two are not code words. So this is a quasi-cyclic restriction code $n_0 = 3$.

Now, let's describe its generator matrix, which has the characteristic of being a circulating matrix by blocks.

**Definition 21.** the generator matrix of an $(mn_0, mk_0)$ quasi-cyclic code is

$$\begin{pmatrix} G_0 & G_1 \cdots & G_{m-1} \\ G_{m-1} & G_0 \cdots & G_{m-2} \\ \vdots & \vdots & \vdots \\ G_2 & G_3 \cdots & G_1 \\ G_1 & G_2 \cdots & G_0 \end{pmatrix}$$

Where each $G_i$ is a $k_0 \times n_0$ submatrix.We see that $G$ given displays the cyclic structure among the rows and columns in terms of the submatrices $G_i's$. For $0 \leq j < m$, let $M_j = [G_j, G_{j-1}, \cdots, G_{j+1}]^T$.The, we can $G$ in the following form:

$$G = [M_0, M_1, \cdots, M_{m-1}]$$

**Example 22.** Consider the (15,5) quasi-cyclic code with parameters $m = 5, n_0 = 3$ and $k_0 = 1$, the following generator matrix:

$$\begin{pmatrix} 001 & 100 & 010 & 110 & 110 \\ 110 & 001 & 100 & 010 & 110 \\ 110 & 110 & 001 & 100 & 010 \\ 010 & 110 & 110 & 001 & 100 \\ 100 & 010 & 110 & 110 & 001 \end{pmatrix}$$

$$\begin{array}{ccccc} M_0 & M_1 & M_2 & M_3 & M_4 \end{array}$$

**Definition 23.** The parity-check matrix $H$: of a quasi-cyclic code is defined similarly to cyclic codes. For a quasi-cyclic code, the parity-check matrix is constructed

by considering the quasi-cyclic structure of the code. The matrix $H$ has the general form:

$$H = \left[-P^T | I_{n-k}\right]$$

Where, $P$ is a matrix that defines the quasi-cyclic structure of the code. $T$ denotes the transpose of matrix $P$. And $I_{n-k}$ is the identity matrix of size $(n-k)$.

**Definition 24.** QC-MDPC code construction:The construction of the $(n,r,w)-$QC-LDPC/QC-MDPC code is based on the construction of the parity check matrix $H$ of length $n = rn_0$ and row-weigth $w$.There are several techniques to construc $H$ including the "circulants row" technique where the matrix $H$ is formed by $n_0(r \times r)$-circulant matrices $H = [H_0 H_1 \cdots H_{n_0-1}]$ such that $H_{n_0-1}$ is non-singular and $w = \sum_{i=0}^{n_0-1} w_i$. The corresponding generator matrix $G$ has the form:

$$G = G = \begin{pmatrix} & (H - 1_{n_0-1} H_0^T) \\ I(_{n-r}) \times (_{n-r}) & (H - 1_{n_0-1} H_1^T) \\ & \vdots \\ & (H - 1_{n_0-1} H_{n_0-2}^T) \end{pmatrix}$$

## 3. McEliece Cryptosystem based on MDPC codes

The McEliece cryptosystem is a public-key cryptographic system developed by Robert McEliece in 1978. It is based on the theory of codes, and its security primarily relies on the seemingly random nature of the code's generator matrix. The challenge of decoding a linear code whose structure is unknown is another key factor in the system's security. This problem is computationally difficult ($NP$-Hard), especially when the code size is large.
Their McEliece variant then works as follows:

### 3.1. **Construction:**

(1) Key-Generation: Construct an MDPC-code of length $n$ and row weight $w$ that can correct up to t errors. For this, generate a parity-check matrix $H, G \in F_2^{r \times n}$ and its corresponding generator matrix with one of the possible constructions.
Public key: generator matrix G.
Private key: parity-check matrix H

(2) Encryption: Let $m \in F_2^{r \times n}$ be a message to be encrypted into $x \in F_2^{r \times n}$ For this generate a codeword $e \in F_2^{r \times n}$ of weight less than $t$, $wt(e) \leq t$. Then the message m is encrypted by:

$$x = mG + e.$$

(3) For decryption choose a decoding algorithm $\phi_H$ (such as the modified bit-flipping decoding algorithm) which knows the parity-check matrix $H$. To decrypt the received word $x \in F_2^n$ into the original message m, compute first:

$$mG = \phi_H(x).$$

Then extract $m$ from the first $(nr)$ positions of $mG$.

Decoding of MDPC codes is carried out using techniques similar to those employed for LDPC codes. Maximum likelihood decoding for LDPC codes is computationally too complex, so iterative decoding algorithms are used instead. However, due to the higher density of the parity-check matrices, these algorithms exhibit lower efficiency in MDPC codes.
A simple family of iterative decoders is based on the concept of bit flipping. These decoders identify, in each iteration, a set of positions in the received word that are likely to be incorrect. The bits at these positions are flipped, and the iteration continues with the updated word. The decoding process continues until the word is fully decoded or a maximum number of iterations is reached.

3.2. **Gallager's bit-flipping.** :Let's delve into the bit-flipping decoding algorithm introduced by Gallager in 1963 for LDPC codes. As the name suggests, in this algorithm, some bits of the received word are flipped to recover the original message. In this context, a "bit" refers to an entry in a binary linear code. This decoding algorithm is well-known and user-friendly. However, it is limited to application only to a binary alphabet, i.e., $G \in F_2$. A modified version of the bit-flipping decoding algorithm for MDPC codes will be presented later.

**Algorithm 1** Bit-flipping decoding algorithm

**Require:** Parity-check matrix $H = (h_{i,j}) \in \{0,1\}^{r \times n}$,
received codeword $y \in \{0,1\}^n$,
maximal number of iterations $b_{max}$
**Ensure:** decoded codeword.

$s \leftarrow Hy^\top$ ▷ compute the syndrome.
**for** $j = 1$ **to** $n$ **do**
  $n_j \leftarrow |\{i \in \{1,\dots,r\} \,|\, h_{ij} = 1\}|$ ▷ Number of 1-entries per column.
**end for**

**for** $a = 1$ **to** $b_{max}$ **do** ▷ Round 1 to $b_{max}$ of algorithm.
  **for** $j = 1$ **to** $n$ **do**
    $u_j \leftarrow |\{i \in \{1,\dots,r\} \,|\, h_{ij} = 1, \sum_l h_{il} y_l = 1 \pmod 2\}|$ ▷ Number of unsatisfied parity checks.
  **end for**
  **for** $j = 1$ **to** $n$ **do**
    **if** $u_j > n_j/2$ **then** ▷ Flipping condition.
      $y_j \leftarrow 1 - y_j$ ▷ Flip bit $j$.
      $s = Hy^\top$ ▷ Recompute the syndrome.
    **end if**
  **end for**
  **if** $s = 0$ **then** ▷ Algorithm stops if syndrome is the zero-vector.
    **return** $y$
  **end if**
**end for**
**return** error

FIGURE 1. Bit-flipping decoding algorithm.

3.3. **Variant of the Gallager's bit flipping algorithm.** :This iterative decoding algorithm provides an error-correction capability for LDPC codes that increases

linearly with the code length and decreases more or less linearly with the weight $w$ of the parity checks. Therefore, when transitioning from LDPC codes to MDPC codes, a degradation in the error-correction capability is expected. However, in the field of cryptography, our interest often lies not in correcting a large number of errors but rather in ensuring an adequate level of security.

The procedure of Gallager's bit-flipping algorithm is as follows: In each iteration, the number of unsatisfied parity-check equations associated with each bit of the message is assessed. If a bit is associated with more than $b$ unsatisfied equations, it is flipped, and then the syndrome is recalculated. This process is repeated until the syndrome becomes zero or after a maximum number of iterations.The complexity of this algorithm is expressed as $O(nwI)$, where: $n$ is the length of the code, $w$ is the weight of the parity-checks, and $I$ is the average number of iterations.

Due to the increased row weight (and the existence of short cycles in the corresponding Tanner graph), MDPC codes can lead to a higher number of iterations. To minimize this problem, a variant of Gallager's algorithm is suggested by changing the choice of b. Some possibilities for b are presented below:

(1) We precalculate b by the following inequality:

$$\frac{1-p_0}{p_0} \leq \left[\frac{1+(1-2p_i)^{k-1}}{1-(1-2p_i)^{k-1}}\right]^{2b-j+1}$$

(2) $b$ is chosen as $Max_{upc}$, the maximum number of unsatisfied parity-check equations;

(3) Our approach: $b = Max_{upc} - \delta$, for a small integer $\delta$.

Approach 2 is broader than Approach 1, resulting in enhanced error-correction capability albeit at the expense of a higher number of iterations. On the other hand, Approach 3 combines the advantages of Approaches 1 and 2 by reducing the total number of iterations obtained by Approach 2 (with more bits flipped in each iteration) and providing error-correction capability as effective as Approach 2.

This latter benefit stems from the following strategy: every time the algorithm fails in decoding, the value of $\delta$ is decreased by 1, and the process is restarted. Clearly, when $\delta = 0$, we revert to Approach 2.

This strategic approach allows dynamically adjusting the parameter $\delta$ based on the algorithm's performance, adapting it to optimize both error-correction capability and iteration efficiency. The flexibility to decrease $\delta$ after each decoding failure contributes to the algorithm's effectiveness in error correction with a reduced number of iterations.