Efficient multiplication of a vector by a matrix MDS

Pablo Freyre Arrozarena, Ernesto Dominguez Fiallo

***Abstract*— An algorithm is proposed for the efficient multiplication of a vector by an $n \times n$ MDS matrix defined on $\mathbb{F}_q$ or by its inverse. The algorithm is based on the multiplication of two polynomials modulo a polynomial of degree n Reed-Solomon code generator and has complexity $O(n \log_2 \log_2(n \log_2 q))$.**

The algorithm only needs to store n values of the $\mathbb{F}_q$ field for the multiplication of a vector by an $n \times n$ MDS matrix and $2n$ values for the multiplication of the vector by the inverse matrix.

***Tóm tắt*— Một thuật toán được đề xuất cho phép nhân hiệu quả của một vectơ với ma trận MDS $\mathrm{n} \times \mathrm{n}$ được xác định trên $\mathbb{F}_\mathrm{q}$ hoặc nghịch đảo của nó. Thuật toán dựa trên phép nhân hai đa thức modul một đa thức bậc n Bộ tạo mã Reed-Solomon và có độ phức tạp $O(n\, log_2\, log_2(n\, log_2\, q))$.**

Thuật toán chỉ cần lưu trữ n giá trị của trường $\mathbb{F}_q$ cho phép nhân vectơ với ma trận MDS $n \times n$ và $2n$ giá trị cho phép nhân vectơ với ma trận nghịch đảo.

***Keywords*— MDS matrices; multiplication of polynomials.**

***Từ khóa*— Ma trận MDS; phép nhân đa thức.**

I. INTRODUCTION

MDS matrices are of great importance in the design of block ciphers and hash functions because they allow the building of optimal linear diffusion layers. However, MDS matrices are not sparse and have a large description, which induces costly implementation in software and hardware. Research focuses on circulant [1] and recursive matrices [2] to reduce implementation costs, but the solutions found to date are only for small dimensions. When the dimensions considered grow, both generating the MDS matrices and reducing their implementation costs, become extremely complex problems that have not yet been resolved.

In this paper, an algorithm is proposed for the multiplication of a vector by an $n \times n$ MDS matrix defined over $\mathbb{F}_q$ or by its inverse, which does not need to store them. With the proposed algorithm, the problem of obtaining an MDS matrix, its inverse and the efficient multiplication of a vector by an nxn MDS matrix are solved. This is of great importance for cryptography, in particular, in light-weight cryptography.

The algorithm is based on the multiplication of two polynomials modulo a polynomial of degree n Reed-Solomon code generator and has complexity $O(n \log_2 \log_2(n \log_2 q))$ [3].

The algorithm only needs to store n values of the $\mathbb{F}_q$ field for the multiplication of a vector by an $n \times n$ MDS matrix and $2n$ values for the multiplication of the vector by the inverse matrix.

The paper is organized as follows. In the preliminaries (Section II) are basic and necessary mathematical elements. Section III describes and bases the algorithms proposed for the generation of non-singular matrices and it concludes with an example that serves as an introduction to the algorithm to generate MDS matrices, which is described in Section IV and the paper ends with the conclusions.

II. PRELIMINARIES

Let $G = GL_n(\mathbb{F}_q)$ be the general linear group and Ω the vector space over $\mathbb{F}_q$. The action of $g \in G$ on $\beta \in \Omega$ is denoted by $\beta^g \in G$. The stabilizer of points $\beta_1, \beta_2, \dots, \beta_i \in \Omega$ in G is denoted by $G_{\beta_1,\beta_2,\dots,\beta_i}$. Let $B = (\beta_1, \beta_2, \dots, \beta_k)$ be a basis for G, then it holds that $G_{\beta_1,\beta_2,\dots,\beta_k} =$

This manuscript is received on November 24, 2022. It is commented on November 24, 2022 and is accepted on December 10, 2022 by the first reviewer. It is commented on December 02, 2022 and is accepted on December 20, 2022 by the second reviewer.

$\{I_n\}$, where I_n is the identity matrix of size $n \times n$. This determines a descending chain of stabilizers:

$$G = G^{(1)} \supseteq G^{(2)} \supseteq \cdots \supseteq G^{(k+1)} = \{I_n\},$$

where $G^{(i)} = G_{\beta_1,\beta_2,\dots,\beta_{i-1}},\ i = 1,2,\dots,k+1$. The orbit of a specific element $\beta \in \Omega$ is $\beta^G = \{\beta^g : g \in G\}$. Given the base $B = (\beta_1, \beta_2, \dots, \beta_k)$, the basic orbits are the sets of points [4]

$$\Delta^{(i)} = \left\{\beta_i^{G^{(i)}}, i = 1,2,\dots,k\right\}$$

The Schreier structure for the base B is defined as the arrangement

$$L = \left[L_{\beta_1}, L_{\beta_2}, \dots, L_{\beta_{\mathrm{k}}}\right]$$

in which

$$L_{\beta_{\mathrm{i}}} = \begin{bmatrix} \beta_i & \alpha_1 & \alpha_2 & \cdots & \alpha_{s_i} \\ I_n & g_1^{(i)} & g_2^{(i)} & \dots & g_{s_i}^{(i)} \end{bmatrix},$$

$\Delta^{(i)} = (\beta_i \quad \alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_{s_i}),$

$g_1^{(i)}, g_2^{(i)}, \dots, g_{s_i}^{(i)} \in G^{(i)},\ \beta_i^{g_1^{(i)} g_2^{(i)} \dots g_j^{(i)}} = \alpha_j,$

$i = 1,2,\dots,k$ and $j = 1,2,\dots,s_i$.

A right transversal for $G^{(i+1)}$ in $G^{(i)}$ is the set $U_i = \{u_0, u_1, \dots, u_{t_i}\}$ of the representatives of the right cosets of $G^{(i+1)}$ in $G^{(i)}$, being $x_0 = 1_G$ and $t_i + 1$ the index of $G^{(i+1)}$ in $G^{(i)}$. Every element $g \in G$ can be expressed in a unique way as the product of elements of a right transversal $U_i, 1,2,\dots,k$ for $G^{(i+1)}$ in $G^{(i)}$, i.e.

$$\forall g \in G, g = \prod_{i=k}^{1} u_i$$

where

$$u_i \in U_i = \left\{\prod_{j=0}^{l} g_j^{(i)} : g_0^{(i)} = I_n, l = 0,1,\dots,s_i\right\}$$

A random selection of the elements of G can be reached by randomly selecting the elements of $U_i, 1,2,\dots,k$ [5].

Given a monic polynomial $g(x) \in \mathbb{F}_q[x], g(x) = g_0 + g_1 x + \cdots + g_{n-1}x^{n-1} + x^n$, the companion matrix is:

$$A = \begin{pmatrix} 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & 0 \\ 0 & 0 & \cdots & 1 \\ -g_0 & -g_1 & \cdots & -g_{n-1} \end{pmatrix}$$

The order $e \in \mathbb{N}$ of the matrix A is the smallest positive integer such that $g(x)$ divides $x^e - 1$. Let $\hat{a}(x) = \hat{a}_0 + \hat{a}_1 x + \cdots + \hat{a}_{n-1}x^{n-1}$, and $a(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}$ be two polynomials such that $\hat{a}(x), a(x) \in \mathbb{F}_q[x]$. It is to be noted that

$$(\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1}) = (a_0, a_1, \dots., a_{n-1}) \cdot A^k$$

it is equivalent to

$$\hat{a}(x) = a(x) \cdot x^k \ mod\ g(x)$$

see [5].

From now on, the base considered for $G = GL_n(\mathbb{F}_q)$ is the base $B = (\beta_1, \beta_2, \dots, \beta_n)$, where $\beta_1 = (1,0 \dots 0)$ and $\beta_i = \left(0, \dots, 0, \underbrace{1}_{i}, 0 \dots 0\right), 2 \leq i \leq n$, are the canonical vectors with 1 at position i and 0 elsewhere.

III. Algorithm

Next, an algorithm is presented with its foundation for the generation of an invertible matrix from the set of all invertible matrices defined over $\mathbb{F}_q$, another algorithm for obtaining the inverse matrix of a matrix obtained by the previous algorithm, and the multiplication of a vector by an invertible matrix.

In the algorithms that are presented below:

- e_1 meets that $n \leq e_1 \leq q^n - 1$ and is defined in L_{β_1} in the algorithm foundation.
- The γ_{1_i} fulfill that $1 \leq \gamma_{1_i} \leq q^n - 2$ and $1 \leq i \leq \gamma_1$.

Algorithm III.1: Generation of a non-singular matrix.

Input:

- The monic polynomial

$v_1(x) = v_{1,0} + v_{1,1}x + \cdots + v_{1,n-1}x^{n-1} + x^n \in \mathbb{F}_q$[x] with $v_{1,0} \neq 0$.

- The primitive polynomials

$$g_i(x) = g_{i,0} + g_{i,1}x + \cdots + g_{i,n-i}x^{n-i} + x^{n-i+1} \in \mathbb{F}_q[x], 1 \le i \le n$$

- The values $c_{i,j} \in \mathbb{F}_q, 2 \le i \le n$ and $0 \le j \le i-2$.
- The values μ_i indicate the position of μ_i in $L_{\beta_\mathrm{i}}, 0 \le \mu_i \le q^{n-i+1} - 2 \; and \; 1 \le i \le n.$

Begin

//Calculation of the first row of matrix A

Input: $(a_0, a_1, \dots, a_{n-1}) = (1, 0, \dots, 0)$

Switch (μ_1)

- Case $\mu_1 = i, i = 0, 1, \dots, e_1 - 1$.

$$(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1}x^{n-1}) = (a_0 + a_1 x + \cdots + a_{n-1}x^{n-1})(x^{\mu_1}) \; mod \; v_1(x)$$

- Case $\mu_1 = \gamma_{1_i}, \; i = 1, 2, \dots, \gamma_1$.

$$(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1}x^{n-1}) = (((a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}) \; (x^{e_1-1}) mod \; v_1(x)) \; (x^{\left(\gamma_{1_1}+\gamma_{1_2}+\cdots+\gamma_{1_i}\right)}) mod \; g_1(x))$$

$$(a_0, a_1, \dots, a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$$

Output: *Row*$_1$ = $(a_0, a_1, \dots, a_{n-1})$

//Calculation of the row j of matrix A, $2 \le j \le n$

Step 1

Input: $(a_0, a_1, \dots, a_{n-1}) = (0, \dots, 0, \underbrace{1}_{j}, 0 \dots 0), 2 \le j \le n$. They are the canonical vectors with 1 at position i and 0 elsewhere.

For *i = j* down to 2 do

Begin

$\hat{a}_0 = a_0 + c_{i,0}a_{\mathrm{i}-1}, \hat{a}_1 = a_1 + c_{i,1}a_{\mathrm{i}-1}, \dots,$

$\hat{a}_{i-2} = a_{\mathrm{i}-2} + c_{i,i-2}a_{\mathrm{i}-1}$

$(\hat{a}_{i-1} + \hat{a}_i x + \dots + \hat{a}_{n-1}x^{n-i})$

$= (a_{i-1} + a_i x + \cdots + a_{n-1}x^{n-i})(x^{\mu_i}) \; mod \; g_i$

$(a_0, a_1, \dots, a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$

End

Step 2

Input: $(a_0, a_1, \dots, a_{n-1})$

Switch (μ_1)

- Case $\mu_1 = i, i = 0, 1, \dots, e_1 - 1$.

$$(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1}x^{n-1}) = (a_0 + a_1 x + \cdots + a_{n-1}x^{n-1})(x^{\mu_1}) \; mod \; v_1(x)$$

- Case $\mu_1 = \gamma_{1_i}, \; i = 1, 2, \dots, \gamma_1$.

$$(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1}x^{n-1}) = (((a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}) \; (x^{e_1-1}) mod \; v_1(x)) \; (x^{\left(\gamma_{1_1}+\gamma_{1_2}+\cdots+\gamma_{1_i}\right)}) mod \; g_1(x))$$

$$(a_0, a_1, \dots, a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$$

Output: *Row j* = $(a_0, a_1, \dots, a_{n-1})$

End

Output: *Non-singular matrix A.*

Proof:

We will perform the proof of the algorithm by induction. We will first show that L_{β_1} and L_{β_2} can be written as shown in the algorithm and then we will assume that it holds for $\mathrm{L}_{\beta_{\mathrm{i}-1}}$ and show that it holds for $\mathrm{L}_{\beta_\mathrm{i}}$.

Let the Schreier structure be for the canonical basis $B = (\beta_1, \beta_2, \dots, \beta_n)$. Be the arrangement $L = \left[L_{\beta_1}, L_{\beta_2}, \dots, L_{\beta_\mathrm{n}}\right]$.

Let

$$L_{\beta_1} = \begin{bmatrix} \beta_1 & \alpha_{1_1} & \alpha_{1_2} & \dots & \alpha_{1_{e_1-1}} & \delta_{1_1} & \cdots & \delta_{1_{\gamma_1}} \\ I_n & A_1 & A_1 & \dots & A_1 & B_1^{\gamma_{1_1}} & \dots & B_1^{\gamma_{1_{\gamma_1}}} \end{bmatrix}$$

where:

- A_1 is the companion matrix of the arbitrary monic polynomial $v_1(x) = (v_{1,0} + v_{1,1}x + \cdots + v_{1,n-1}x^{n-1} + x^n) \in \mathbb{F}_q[\mathrm{x}]$ with $v_{1,0} \neq 0$ and $e_1, n \le e_1 \le q^n - 1$, is the order of the matrix A_1.
- $\beta_1 = (1, 0, \dots, 0), \alpha_{1_1} = \beta_1 A_1$

  $\alpha_{1_{s+1}} = \alpha_{1_s}A_1, \; 1 \le s \le e_1 - 2.$

- $\alpha_{1_s} \neq \delta_{1_{\gamma_j}}, \quad 1 \leq s \leq e_1 - 1 \; and \; 1 \leq j \leq \gamma_1.$
- B_1 is the companion matrix of a primitive polynomial $g_1(x) = (g_{1,0} + g_{1,1}x + \cdots + g_{1,n-1}x^{n-1} + x^n) \in \mathbb{F}_q[x]$.
- $\delta_{1_l} \neq \delta_{1_m}$ for $l \neq$ m, $1 \leq l, m \leq \gamma_1$ and γ_1 are such that $\delta_{1_1} = \alpha_{1_{e_1-1}} B_1^{\gamma_{1_1}}$ and $\delta_{1_z} = \delta_{1_{z-1}} B_1^{\gamma_{1z}}, 2 \leq z \leq \gamma_1$.

Writing L_{β_1} in the form shown, the right transversal U_1 of $\mathrm{G}^{(2)}$ in $\mathrm{G}^{(1)}$ is determined and given $u_1 \in \mathrm{U}_1$, $((\mathrm{a}_0, \mathrm{a}_1, \ldots., \mathrm{a}_{n-1})\, \mathrm{u}_1)$ is as shown below:

If $i = 0, 1, \ldots, e_1 - 1.$

$$(\hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{n-1}x^{n-1}) = (a_0 + a_1 x + \cdots + a_{n-1}x^{n-1})(x^i) \; mod \; v_1(x)$$

Or

$If \; i = \gamma_{1_i}, \; i = 1, 2, \ldots, \gamma_1.$

$$(\hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{n-1}x^{n-1}) = (((a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}) \, (x^{e_1-1}) mod \; v_1(x))$$

$$(x^{\left(\gamma_{1_1} + \gamma_{1_2} + \cdots + \gamma_{1_i}\right)}) mod \; g_1(x))$$

It is being demonstrated in the form that L_{β_1} takes in the algorithm.

Let

$$L_{\beta_2} = \begin{bmatrix} \beta_2 & \alpha_{2_1} & & \alpha_{2_{q^{n-1}-2}} \\ I_n & A_2 & \cdots & A_2 \\ \beta_{2_{\alpha^0}} & \alpha_{2^0_1} & & \alpha_{2^0_{q^{n-1}-2}} \\ D_{2_{\alpha^0}} & A_2 & & A_2 \\ \vdots & & \ddots & \vdots \\ \beta_{2_{\alpha^{q-2}}} & \alpha_{2^{q-2}_1} & \cdots & \alpha_{2^{q-2}_{q^{n-1}-2}} \\ D_{2_{\alpha^{q-2}}} & A_2 & & A_2 \end{bmatrix}$$

where:

- $\beta_2 = (0,1,0,\ldots,0), \beta_{2_{\alpha^0}} = (\alpha^0, 1, 0, \ldots, 0), \ldots, \beta_{2_{\alpha^{q-2}}} = (\alpha^{q-2}, 1, 0, \ldots, 0).$
- $\alpha_{2_1} = (0, 0, 1, 0, \ldots, 0), \; \alpha_{2^0_1} = (\alpha^0, 0, 1, 0, \ldots, 0), \ldots, \alpha_{2^{q-2}_1} = (\alpha^{q-2}, 0, 1, 0, \ldots, 0).$
- $A_2 = \begin{bmatrix} I_1 & \cdots & 0 \\ 0 & & \\ & \vdots A_{2_{(n-1)}} & \\ 0 & & \end{bmatrix}$, where $A_{2_{(n-1)}}$ is the companion matrix of a primitive polynomial $g_2(x) = (g_{2,0} + g_{2,1}x + \cdots + g_{2,n-2}x^{n-2} + x^{n-1}) \in \mathbb{F}_q[x]$. The matrix A_2 stabilizes all the elements of the base before β_2, i.e. β_1 in this case.
- $\alpha_{2_{q^{n-1}-2}} = (0, h_1, \ldots., h_{n-2}, h_{n-1}), \quad \ldots, \alpha_{2^{q-2}_{q^{n-1}-2}} = \left(\alpha_{2^{q-2}_{q^{n-1}-2}}, h_1, \ldots., h_{n-2}, h_{n-1}\right),$
  $$h_t, 1 \leq t \leq n - 1$$

are arbitrary values of $\mathbb{F}_q$.

- $D_{2_{\alpha^0}} = \begin{bmatrix} I_1 & \cdots & 0 \\ 0 & & \\ & \vdots A_{2_{(n-1)}} & \\ 0 & & \\ & (h_{n-1})^{-1}\alpha^0 & \end{bmatrix}$
- $D_{2_{\alpha^k}} = \begin{bmatrix} I_1 & \cdots & 0 \\ 0 & & \\ & \vdots A_{2_{(n-1)}} & \\ 0 & & \\ -(h_{n-1})^{-1}\alpha^{k-1} + \alpha^k & & \end{bmatrix},$

  $1 \leq k \leq q - 2.$

Writing L_{β_2} in the form shown, the right transversal U_2 of $\mathrm{G}^{(3)}$ in $\mathrm{G}^{(2)}$ is determined and given $u_2 \in \mathrm{U}_2$, $((\mathrm{a}_0, \mathrm{a}_1, \ldots., \mathrm{a}_{n-1})\, \mathrm{u}_2)$ is as shown below:

$$\widehat{a}_0 = a_0 + c_{2,0}a_1, (\hat{a}_1 + \hat{a}_2 x + \ldots + \hat{a}_{n-1}x^{n-2}) =$$

$$= (a_1 + a_2 x + \cdots + a_{n-1}x^{n-2})(x^{\mu_2}) \; mod \; g_2$$

The value μ_2 indicates the position of μ_2 in $\mathrm{L}_{\beta_2}, 0 \leq \mu_2 \leq \mathrm{q}^{\mathrm{n}-1} - 2.$

It is being demonstrated in the form that L_{β_2} takes in the algorithm.

Assuming that the algorithm is true for i-1, we will show that it is true for i.

Let

$$L_{\beta_i} = \begin{bmatrix} \beta_i & \alpha_{i_1} & & \alpha_{i_{q^{n-i+1}-2}} \\ I_n & A_i & \cdots & A_i \\ \beta_{i_{\alpha^0}} & \alpha_{i_1^0} & & \alpha_{2^0_{q^{n-i+1}-2}} \\ D_{i_{\alpha^0}} & A_i & & A_i \\ \vdots & & \ddots & \vdots \\ \beta_{i_{\alpha^{q-2\ldots q-2}}} & \alpha_{i_1^{q-2\ldots q-2}} & \cdots & \alpha_{i^{q-2\ldots q-2}_{q^{n-i+1}-2}} \\ D_{2_{\alpha^{q-2\ldots q-2}}} & A_i & & A_i \end{bmatrix}$$

where

- $\beta_i = (0, 0, \ldots, 0, \underbrace{1}_{i}, 0 \ldots 0)$,

  $\beta_i = (\alpha^0, 0, \ldots, 0, \underbrace{1}_{i}, 0 \ldots 0), \ldots,$

  $\beta_{i_{\alpha^{q-2\ldots q-2}}} = (\alpha^{q-2}, \ldots, \alpha^{q-2}, \underbrace{1}_{i}, 0 \ldots 0)$

- $\alpha_{i_1} = (0, 0, \ldots, 0, \underbrace{1}_{i+1}, 0 \ldots 0)$,

  $\alpha_{i_1^0} = (\alpha^0, 0, \ldots, 0, \underbrace{1}_{i+1}, 0 \ldots 0), \ldots,$

  $\alpha_{i_1^{q-2,\ldots,q-2}}$
  $= (\alpha^{q-2}, \ldots, \alpha^{q-2}, 0, \underbrace{1}_{i+1}, 0 \ldots 0)$

- $A_i = \begin{bmatrix} I_{i-1} & \ldots & 0 \\ 0 & & \\ & \vdots A_{i_{(n-i+1)}} & \\ 0 & & \end{bmatrix}$, where $A_{i_{(n-i+1)}}$ is the companion matrix of a primitive polynomial $g_i(x) = (g_{i,0} + g_{i,1}x + \cdots + g_{i,n-i}x^{n-i} + x^{n-i+1}) \in \mathbb{F}_q[x]$. The matrix A_i stabilizes all the elements of the base before β_i.
- $\alpha_{i_{q^{n-i+1}-2}} = (0, 0, \ldots, 0, \underbrace{r_i}_{i}, \ldots, r_{n-1}), \ldots,$

  $\alpha_{i^{q-2\ldots q-2}_{q^{n-i+1}-2}}$
  $= (\alpha^{q-2}, \ldots, \alpha^{q-2}, \underbrace{r_i}_{i}, \ldots, r_{n-1})$

  $r_t, i \le t \le n-1$, are arbitrary value of $\mathbb{F}_q$.

- $D_{i_{\alpha^0}} = \begin{bmatrix} I_{i-1} & 0 \ldots & 0 \\ 0 & & \\ & \vdots A_{i_{(n-i+1)}} & \\ 0 & & \\ (r_{n-1})^{-1}\alpha^0 0 \ldots 0 & & \end{bmatrix} \ldots$

$$D_{i_{\alpha^{q^{n-2},\ldots,q^{n-2}}}} = \\ = \begin{bmatrix} I_{i-1} & 0 \quad \ldots & 0 \\ 0 & & \\ & \vdots A_{i_{(n-i+1)}} & \\ 0 & & \\ -(r_{n-1})^{-1}\alpha^{q-3} + \alpha^{q-2} \ldots & & \end{bmatrix}$$

Writing L_{β_i} in the form shown, the right transversal U_i of $G^{(i)}$ in $G^{(i+1)}$ is determined and given $u_i \in U_i$, $((a_0, a_1, \ldots, a_{n-1})\, u_i)$ is as shown below:

$\hat{a}_0 = a_0 + c_{i,0}a_{i-1}, \hat{a}_1 = a_1 + c_{i,1}a_{i-1}, \ldots,$

$\hat{a}_{i-2} = a_{i-2} + c_{i,i-2}a_{i-1}$

$(\hat{a}_{i-1} + \hat{a}_i x + \ldots + \hat{a}_{n-1}x^{n-i})$

$= (a_{i-1} + a_i x + \cdots + a_{n-1}x^{n-i})(x^{\mu_i}) \bmod g_i$

The values μ_i indicate the position of μ_i in $L_{\beta_i}, 0 \le \mu_i \le q^{n-i+1} - 2$.

It is being demonstrated in the form that L_{β_i} takes in the algorithm.

Then, having defined the input values of algorithm III.1, the generation of an invertible matrix A can be done as follows [5]:

Input:

$B = (\beta_1, \beta_2, \ldots, \beta_n)$

```
  For $k = 1$ to n
  $Row_k = (\beta_k)\mu_k\mu_{k-1} \ldots \mu_1$
```

Output: *Invertible matrix A.*

Then the algorithm is proved.

The inverse of a matrix A obtained by the previous algorithm is calculated as follows [5]:

Algorithm III.2: Generation of the inverse of a non-singular matrix.

Input:

The input values are similar to those of the algorithm III.1.

Begin

//Calculation of the row i of matrix A^{-1}, $1 \le i \le n$.

$(a_0, a_1, \dots., a_{n-1}) = \left(0, \dots, 0, \underbrace{1}_{i}, 0 \dots 0\right)$, $1 \le i \le n$. They are the canonical vectors with 1 at position i and 0 elsewhere.

```
**Step 1**
Switch ( $\mu_1$)
    • Case $\mu_1 = i, i = 0, 1, \dots, e_1 - 1$.
      $(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1} x^{n-1})$
      $= (a_0 + a_1 x + \cdots$
      $+ a_{n-1} x^{n-1})(x^{-\mu_1}) \, mod \, v_1(x)$

    • Case $\mu_1 = \gamma_{1_i}, \ i = 1, 2, \dots, r_1$.
      $(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1} x^{n-1}) =$
      $\left(\left((a_0 + a_1 x + \cdots\right.\right.$
      $\left.+ a_{n-1} x^{n-1})(x^{-\left(\gamma_{1_1} + \gamma_{1_2} + \cdots + \gamma_{1_i}\right)}) mod \, g_1(x)\right)$
                  $\left. x^{-(e_1 - 1)}\right) mod \, v_1(x)$

      $(a_0, a_1, \dots., a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$

Output: $(a_0, a_1, \dots., a_{n-1})$
**Step 2**
Input: $(a_0, a_1, \dots., a_{n-1})$
For $j = 2$ to $n$ do
Begin
$\hat{a}_0 = a_0 - c_{i,0} a_{\mathrm{i}-1}, \hat{a}_1 = a_1 - c_{i,1} a_{\mathrm{i}-1}, \dots,$
$\hat{a}_{i-2} = a_{\mathrm{i}-2} - c_{i,i-2} a_{\mathrm{i}-1}$
$(\hat{a}_{i-1} + \hat{a}_i x + \dots + \hat{a}_{n-1} x^{n-1})$
$= (a_{i-1} + a_i x + \cdots$
            $+ a_{n-1} x^{n-i}) \, (x^{-\mu_i}) \, mod \, g_i)$

      $(a_0, a_1, \dots., a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$

$(a_0, a_1, \dots., a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$

End
Output: $Row_i = (a_0, a_1, \dots., a_{n-1})$

**End**
**Output:** *Matrix* $A^{-1}$
```

The multiplication of $(a_0, a_1, \dots., a_{n-1}) \in \mathbb{F}_q{}^n$ by the matrix A obtained by the previous algorithm is done as follows [5]:

Algorithm III.3: Multiplication of $(a_0, a_1, \dots., a_{n-1})$ by the matrix *A*.

```
**Begin**

The input values are similar to those of the algorithm III.1.

**Step 1**
Input: $(a_0, a_1, \dots., a_{n-1})$
For *i = n* down to *2* do
Begin
$\hat{a}_0 = a_0 + c_{i,0} a_{\mathrm{i}-1}, \hat{a}_1 = a_1 + c_{i,1} a_{\mathrm{i}-1}, \dots,$
$\hat{a}_{i-2} = a_{\mathrm{i}-2} + c_{i,i-2} a_{\mathrm{i}-1}$
$(\hat{a}_{i-1} + \hat{a}_i x + \dots + \hat{a}_{n-1} x^{n-1})$
$= (a_{i-1} + a_i x + \cdots$
            $+ a_{n-1} x^{n-i})(x^{\mu_i}) \ mod \, g_i)$
$(a_0, a_1, \dots., a_{n-1})$
$= (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{i-2}, \hat{a}_{i-1}, \dots, \hat{a}_{n-1})$
End
**Step 2**
Input: $(a_0, a_1, \dots., a_{n-1})$
Switch ( $\mu_1$)
    • Case $\mu_1 = i, i = 0, 1, \dots, e_1 - 1$.
      $(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1} x^{n-1})$
      $= (a_0 + a_1 x + \cdots$
      $+ a_{n-1} x^{n-1})(x^{\mu_1}) \, mod \, v_1(x)$

    • Case $\mu_1 = \gamma_{1_i}, \ i = 1, 2, \dots, \gamma_1$.
      $(\hat{a}_0 + \hat{a}_1 x + \dots + \hat{a}_{n-1} x^{n-1})$
          $= (((a_0 + a_1 x + \cdots$
          $+ a_{n-1} x^{n-1}) \, (x^{e_1 - 1}) mod \, v_1(x))$
          $(x^{\left(\gamma_{1_1} + \gamma_{1_2} + \cdots + \gamma_{1_i}\right)}) mod \, g_1(x))$

Output: $(a_0, a_1, \dots., a_{n-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{n-1})$

**End**
**Output:**$(a_0, a_1, \dots., a_{n-1})$
```

Example: Below is an example of determining the Schreier structures and obtaining a non-singular matrix A ∈ GL4($\mathbb{F}_2$). In order to express the vector $(a_0, a_1, \dots, a_{n-1})$ like a number, the following expression is assumed $(a_0 + a_1 2 + a_2 2^2 + a_3 2^3)$.

Let the Schreier structure for the canonical basis $B = (\beta_1, \beta_2, \beta_3, \beta_4)$. Be the arrangement $L = [L_{\beta_1}, L_{\beta_2}, L_{\beta_3}, L_{\beta_4}]$.

To determine L_{β_1} it is assumed that:

$v_1(x) = (1 + x^2 + x^3 + x^4)$ and its companion matrix is:

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

$g_1(x) = (1 + x + x^4)$ and its companion matrix is:

$$B_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Then L_{β_1} is:

$$L_{\beta_1} = \begin{bmatrix} 1 & 2 & 4 & 8 & 13 & 7 & 14 & 3 \\ \mathrm{I}_4 & \mathrm{A}_1 & \mathrm{A}_1 & \mathrm{A}_1 & \mathrm{A}_1 & \mathrm{A}_1 & \mathrm{A}_1 & \mathrm{B}_1^8 \\ 5 & 10 & 15 & 9 & 6 & 12 & 11 & \\ \mathrm{B}_1^4 & \mathrm{B}_1 & \mathrm{B}_1^3 & \mathrm{B}_1^2 & \mathrm{B}_1^6 & \mathrm{B}_1 & \mathrm{B}_1 & \end{bmatrix}$$

To determine L_{β_2} it is assumed that: $g_2(x) = (1 + x^2 + x^3)$ and its accompanying matrix is:

$$\mathrm{A}_2' = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

and

$$A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$$D_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

and then L_{β_2} is:

$$L_{\beta_2} = \begin{bmatrix} 2 & 4 & 8 & 10 & 14 & 6 & 12 \\ I_4 & A_2 & A_2 & A_2 & A_2 & A_2 & A_2 \\ 3 & 5 & 9 & 11 & 15 & 7 & 9 \\ D_1 & A_2 & A_2 & A_2 & A_2 & A_2 & A_2 \end{bmatrix}$$

To determine L_{β_3} it is assumed that: $g_3(x) = (1 + x + x^2)$ and its accompanying matrix is:

$$A_3' = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \text{ and } A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$D_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \text{ and } D_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

and then L_{β_3} is:

$$L_{\beta_3} = \begin{bmatrix} 4 & 8 & 12 & 5 & 9 & 13 \\ I_4 & A_3 & A_3 & D_1 & A_3 & A_3 \\ 6 & 10 & 14 & 7 & 11 & 15 \\ D_2 & A_3 & A_3 & D_1 & A_3 & A_3 \end{bmatrix}$$

To determine L_{β_4} it is assumed that: $g_3(x) = (1 + x)$ and its accompanying matrix is:

$$A_3' = [1],$$

$$A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix},$$

$$D_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \text{ and } D_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

then L_{β_4} is:

$$L_{\beta_4} = \begin{bmatrix} 8 & 12 & 10 & 14 & 9 & 13 & 11 & 15 \\ I_4 & D_1 & D_2 & D_1 & D_3 & D_1 & D_2 & D_1 \end{bmatrix}$$

Starting from the Schreier structure defined above and applying the algorithm III.1 to generate a non-singular matrix A ∈ GL4($\mathbb{F}_2$) we have:

Input:

- The monic polynomial $v_1(x) = (1 + x^2 + x^3 + x^4)$ and its companion matrix has order $e_1 = 7$
- The primitive polynomials:

$$g_i(x) = g_{i,0} + g_{i,1}x + \cdots + g_{i,4-i}x^{4-i} + x^{4-i+1} \in \mathbb{F}_q[x], 1 \le i \le 4$$

- The values $c_{i,j} = 0, 2 \le i \le 4$ and $0 \le j \le i-2$.
- The value $\mu_1 = 4$ and the values $\mu_i = 0, 2 \le i \le 4$

```
Begin
//Calculation of the rows of matrix A, $1 \le j \le 4$
Input: $(a_0, a_1, \dots, a_3) = (0, \dots, 0, \underbrace{1}_{j}, 0 \dots 0)$ (the canonical vectors with 1 at position i and 0 elsewhere).
For $i = j$ down to 1 do
    $(\hat{a}_0 + \hat{a}_1x + \dots + \hat{a}_3x^3) = (a_0 + a_1x + \cdots + a_3x^3)(x^{\mu_i}) \bmod v_1(x)$
    $(a_0, a_1, \dots, a_3) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_3)$
Output: $Row_j = (a_0, a_1, \dots, a_3)$
End
```

Output: $A = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

Taking into account the previous example, we see that when algorithm III.1 holds that:

- The values $c_{i,j} = 0, 2 \le i \le n$ and $0 \le j \le i-2$.
- The values $\mu_i = 0,\ 2 \le i \le n$.

then, it is not necessary to define the primitive polynomials in the input.

It is to be noted that the submatrix RA of matrix A, where $R_A = \begin{bmatrix} 1011 \\ 1110 \\ 0111 \end{bmatrix}$, it is similar with the rows and columns permuted to the submatrix R of the generating matrix of the binary cyclic code V = [7,3] with generating polynomial h(x) = 1 + x2 + x3 + x4.

$$G = [I_3, R] = \begin{bmatrix} 1001110 \\ 0100111 \\ 0011101 \end{bmatrix}.$$

This similarity between the matrices serves as motivation for the development of the following section.

IV. Generations of MDS matrices

Let *G = [Ik,−R]* be the generating matrix of a cyclic code of length *n* where *Ik* is the *k* × *k* identity matrix and the matrix −*R* is a matrix of *k* rows and *r* = *n*−*k* columns that satisfies that:

$$-R = \begin{bmatrix} -(x^{n-r-1} mod\ h(x)) \\ \cdots \\ -(x^{r+1} mod\ h(x)) \\ -(x^{r} mod\ h(x)) \end{bmatrix},$$ where $h(x) = (h_{1,0} + h_{1,1}x + \cdots + h_{1,r-1}x^{r-1} + x^r) \in \mathbb{F}_q$[x] is the generating polynomial of the code.

From the algorithm III.1 defined in the previous section to generate a matrix A ∈ GLn($\mathbb{F}_q$) we have:

Input:

- The monic polynomial

$$h(x) = (h_{1,0} + h_{1,1}x + \cdots + h_{1,r-1}x^{r-1} + x^r) \in \mathbb{F}_q[\mathrm{x}]$$

which is the generator polynomial of the code, $h_{1,0} \neq 0$.

- The values $c_{i,j} = 0, 2 \le i \le r$ and $0 \le j \le i-2$.
- The value $\mu_1 = r$ and the values $\mu_i = 0, 2 \le i \le r$.

```
Begin
//Calculation of the rows of matrix A, $1 \le j \le r$
Input: $(a_0, a_1, \dots, a_{r-1}) = (0, \dots, 0, \underbrace{1}_{j}, 0 \dots 0)$
(the canonical vectors with 1 at position i and 0 elsewhere).
For $i = j$ down to 1 do
    $(\hat{a}_0 + \hat{a}_1x + \dots + \hat{a}_{r-2}x^{r-2}) = \left(a_0 + a_1x + \cdots + a_{r-1}x^{r-1})(-(x^{\mu_i}) mod\ h(x)\right)$
    $(a_0, a_1, \dots, a_{r-1}) = (\hat{a}_0, \hat{a}_1, \dots, \hat{a}_{r-1})$
Output: $Row_j = (a_0, a_1, \dots, a_{r-1})$
End
```

Output: $-A = \begin{bmatrix} -(x^r mod\ h(x)) \\ -(x^{r+1} mod\ h(x)) \\ \cdots \\ -(x^{2r-1} mod\ h(x)) \end{bmatrix}$

which is a submatrix of the matrix *–R*. It must be fulfilled that $k \geq r$.

Observation: If the cyclic code is a Reed-Solomon (RS) code, denoted by $RS = [q - 1, q - d, d]$, $q > 2$ which has as generating polynomial $h(x) = (x-\alpha)(x-\alpha^2) \ldots (x-\alpha^{d-1}) \in \mathbb{F}_q[x]$ where $\alpha \in \mathbb{F}_q$ is a primitive element [6], then with the use of the algorithm III.1 we have:

Input:

- The monic polynomial $h(x) = (h_{1,0} + h_{1,1}x + \cdots + h_{1,d-2}x^{d-2} + x^{d-1}) \in \mathbb{F}_q[x]$

It is the generating polynomial of the RS code.

- The values $c_{i,j} = 0, 2 \leq i \leq d-1$ and $0 \leq j \leq i-2$.
- The value $\mu_1 = d-1$ and the values $\mu_i = 0, 2 \leq i \leq d-1$.

Begin

//Calculation of the rows of matrix A, $1 \leq j \leq d-1$.

Input: $(a_0, a_1, \ldots., a_{d-2}) = (0, \ldots, 0, \underbrace{1}_{j}, 0 \ldots 0)$ (the canonical vectors with 1 at position i and 0 elsewhere).

For $i = j$ down to 1 do

$$(\hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{d-2}x^{d-2}) = \big(a_0 + a_1 x + \cdots + a_{d-2}x^{d-2}\big)\big(-(x^{\mu_i}) mod\ h(x)\big)$$

$$(a_0, a_1, \ldots., a_{d-2}) = (\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{d-2})$$

Output: $Row_j = (a_0, a_1, \ldots., a_{d-2})$

End

Output: $-R_{RS} = \begin{bmatrix} -(x^{d-1} mod\ h(x)) \\ -(x^{d} mod\ h(x)) \\ \cdots \\ -(x^{2d-3} mod\ h(x)) \end{bmatrix}$

and $-R_{RS}$ is an MDS matrix.

Applying algorithm III.3, the multiplication of $(a_0, a_1, \ldots., a_{d-2}) \in \mathbb{F}_q{}^{d-1}$ by the matrix $-R_{RS}$ is done as follows:

Input:

- The monic polynomial $h(x) = (h_{1,0} + h_{1,1}x + \cdots + h_{1,d-2}x^{d-2} + x^{d-1}) \in \mathbb{F}_q[x]$

$$(\hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{d-2}x^{d-2}) = \big(a_0 + a_1 x + \cdots + a_{d-2}x^{d-2}\big)\big(-(x^{d-1}) mod\ h(x)\big)$$

$$(a_0, a_1, \ldots., a_{d-2}) = (\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{d-2})$$

Output: $(a_0, a_1, \ldots, a_{d-2})$

Applying algorithm III.3, the multiplication of $(a_0, a_1, \ldots., a_{d-2}) \in \mathbb{F}_q{}^{d-1}$ by the matrix $(-R_{RS})^{-1}$ is done as follows:

Input:

- The monic polynomial $h(x) = (h_{1,0} + h_{1,1}x + \cdots + h_{1,d-2}x^{d-2} + x^{d-1}) \in \mathbb{F}_q[x]$

$$(\hat{a}_0 + \hat{a}_1 x + \ldots + \hat{a}_{d-2}x^{d-2}) = \big(a_0 + a_1 x + \cdots + a_{d-2}x^{d-2}\big)(-x^{d-1})^{-1} mod\ h(x)\big) =$$

$$(a_0, a_1, \ldots., a_{d-2}) = (\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{d-2})$$

Output: $(a_0, a_1, \ldots., d-2)$

If calculated in advance $(-x^{d-1})^{-1}\ mod\ h(x)$, the multiplication of the vector by the inverse of the nxn MDS matrix requires storing $2(d-1)$ elements of the field $\mathbb{F}_q$ and has complexity $O(n \log_2 \log_2 (n \log_2 q))$ which is the complexity of multiplying two polynomials $f(x)$ and $t(x)$ modulo a polynomial $s(x)$, where $f(x)$,$t(x), s(x) \in \mathbb{F}_q[x]$ and the degree of $s(x)$ is $d - 1$ [3].

Example of obtaining an 4 × 4 MDS matrix in $\mathbb{F}_{2^4}$ and multiplication of a vector by the matrix:

Let be the finite field $\mathbb{F}_{2^4}$ with irreducible polynomial $1 + x + x^4$. It is considered an RS code of parameters $[15, 11, 5]$ with generating polynomial

$$h(x) = (x+\beta)(x+\beta^2)(x+\beta^3)(x+\beta^4) = (1+\beta+\beta^2)+\beta^3x+\beta^6x^2+\beta^{13}x^3+x^4$$

where $\beta = (0,1,0,0\,) = 2$ is a root of $1+x+x^4$ that is, $\beta^4 = 1+\beta = (1,1,0,0) = 3, \beta^5 = \beta+\beta^2 = (0,1,1,0) = 6, \dots, \beta^{13} = 1+\beta^2+\beta^3 = (1,0,1,1) = D$.

The matrix $-R$ is:

$$-R = \begin{bmatrix} (x^{10}\ mod\ h(x)) \\ (x^9\ mod\ h(x)) \\ (x^8\ mod\ h(x)) \\ (x^7\ mod\ h(x)) \\ (x^6\ mod\ h(x)) \\ (x^5\ mod\ h(x)) \\ (x^4\ mod\ h(x)) \end{bmatrix}$$

From the algorithm III.1 defined in the previous section to generate a random matrix $A \in GL_n(\mathbb{F}_q)$ and using $h(x)$, we have the submatrix:

$$-R_{RS} = \begin{bmatrix} (x^4\ mod\ h(x)) \\ (x^5\ mod\ h(x)) \\ (x^6\ mod\ h(x)) \\ (x^7\ mod\ h(x)) \end{bmatrix}$$

$$= \begin{bmatrix} \beta^{10}+\beta^3x+\beta^6x^2+\beta^{13}x^3 \\ \beta^8+\beta^8x+\beta^7x^2+\beta x^3 \\ \beta^{11}+\beta^5x+\beta^{11}x^2+\beta x^3 \\ \beta^{11}+\beta^{13}x+\beta^{13}x^2+\beta^{10}x^3 \end{bmatrix}.$$

$$\begin{bmatrix} \beta^{10} & \beta^3 & \beta^6 & \beta^{13} \\ \beta^8 & \beta^8 & \beta^7 & \beta \\ \beta^{11} & \beta^5 & \beta^{11} & \beta \\ \beta^{11} & \beta^{13} & \beta^{13} & \beta^{10} \end{bmatrix}$$

$$= \begin{bmatrix} 7 & 8 & C & D \\ 5 & 5 & B & 2 \\ E & 6 & E & 2 \\ E & D & D & 7 \end{bmatrix}$$

is an 4×4 MDS matrix. Let be the vector $(\beta, \beta^2, \beta^3, \beta^4)$.

By the previous algorithm III.3 of multiplication of a vector by a matrix, it is obtained that:

$$(\beta+\beta^2x+\beta^3x^2+\beta^4x^3)(x^4)\ mod\ h(x)$$
$$= (\beta+\beta^2x+\beta^3x^2+\beta^4x^3)(\beta^{10}+\beta^3x+\beta^6x^2+\beta^{13}x^3)\ mod\ h(x)$$
$$= 1+\beta^8x+\beta^6x^2+\beta^7x^3$$

which is the output:

$$(1, \beta^8, \beta^6, \beta^{13}) = (1, 5, C, D).$$

V. Conclusions

In the work, algorithms are developed for generating MDS matrices, calculating their inverse, and multiplying a vector by an MDS matrix and by its inverse. The proposed algorithms allow using a large MDS matrix since to operate with it, it is not necessary to store it completely. This leaves aside the problem of generating large MDS matrices and their inverses with certain structures that guarantee computational efficiency in their implementation.

The multiplication of a vector by the $n \times n$ MDS matrix needs to store n values of the $\mathbb{F}_q$ field and $2n$ for the multiplication of the vector by the inverse matrix. In both cases, the complexity is $O(n \log_2 \log_2(n \log_2 q))$.

References

[1] Gupta, K. C., Pandey, S. K., and Venkateswarlu, A. “On the direct construction of recursive MDS matrices,” Designs, Codes and Cryptography, 82(1), 77-94, 2017.

[2] Gupta, K. C., Pandey, S. K., and Samanta, S. “Construction of Recursive MDS Matrices Using DLS Matrices,” In International Conference on Cryptology in Africa (pp. 3-27). Springer, Cham, 2022.

[3] J. Alman and V. V. Williams, “A refined laser method and faster matrix multiplication,” in Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA). SIAM, 2021, pp. 522–539.

[4] D. F. Holt, B. Eick, and E. A. O’Brien, “Handbook of computational group theory,” CRC Press, 2005.

[5] Freyre P., Díaz N. and Morgado E., “Some algorithms related to matrices with entries in a finite field,” Journal of Discrete Mathematical Sciences and Cryptography, vol. 12(5), 2009, pp. 509–519.

[6] W. W. Peterson, W. Peterson, E. J. Weldon, and E. J. Weldon, “Error-correcting codes,” MIT press, 1972.

ABOUT THE AUTHOR

Pablo Freyre Arrozarena

Workplace: Institute of Cryptography, University of Havana.

Email: pfreyre@matcom.uh.cu

Education: Graduated of Mathematics in 1988; receive his Doctor’s degree in 1998.

Recent research interests: Symmetric cryptography; Post-quantum cryptography.

Ernesto Dominguez Fiallo

Workplace: Institute of Cryptography, University of Havana.

Education: Graduated of Mathematics in 2015; receive his Master’s degree in 2019.

Recent research interests: Post-quantum cryptography.