On the Direct Construction of Recursive MDS Matrices

Kishan Chand Gupta, Sumit Kumar Pandey, Ayineedi Venkateswarlu

On the Direct Construction of Recursive MDS Matrices

Kishan Chand Gupta[1], Sumit Kumar Pandey[2], and Ayineedi Venkateswarlu[3]

[1] Applied Statistics Unit, Indian Statistical Institute,
203, B.T.Road, Kolkata - 700108, INDIA.
kishan@isical.ac.in
[2] C.R.Rao AIMSCS, University of Hyderabad Campus,
Prof. C.R. Rao Road, Hyderabad - 500046, INDIA.
emailpandey@gmail.com
[3] Computer Science Unit, Indian Statistical Institute - Chennai Centre,
MGR Knowledge City Road, Taramani, Chennai - 600113, INDIA.
venku@isichennai.res.in

Abstract. MDS matrices allow to build optimal linear diffusion layers in the design of block ciphers and hash functions. There has been a lot of study in designing efficient MDS matrices suitable for software and/or hardware implementations. In particular recursive MDS matrices are considered for resource constrained environments. Such matrices can be computed as a power of simple companion matrices, *i.e.*, an MDS matrix $M = C_g^k$ for some companion matrix corresponding to a polynomial $g(X) \in \mathbb{F}_q[X]$ of degree k. In this paper we first show that if the polynomial $g(X)$ has no multiple of weight $\leq k$ and degree $\leq 2k-1$ then the matrix $M = C_g^k$ satisfies MDS property. In fact, we show that the corresponding MDS code $\mathcal{M}$ is a shortened code of the cyclic code Γ defined by $g(X)$ (code words of $\mathcal{M}$ are given by the multiples of $g(X)$ with degree at most $2k-1$). This characterization answers the issues raised by Agout *et. al.* in FSE-2014 paper to some extent. We then revisit Augot *et al.* method (FSE-2014) to construct recursive MDS matrices using shortened BCH codes (which are also MDS) and propose an improved method. Our method improves upon their method by confining the computation to only the potential cases where we are guaranteed to get recursive MDS matrices, and thus greatly reducing the complexity. As a consequence we are able to provide formula for the number of such recursive MDS matrices, whereas in FSE-2014 paper, the same numbers are provided by exhaustively searching for some small parameter choices. We also present a few ideas making the search faster for finding efficient recursive MDS matrices in this class. Using our approach it is possible to exhaustively search this class for some bigger parameter choices which was not possible earlier.

Keywords: Diffusion layer, MDS codes, Recursive MDS matrices, Companion matrices, Cyclic Codes, BCH codes, Shortened Codes.

1 Introduction

MDS matrices provide maximal diffusion which is useful in cryptographic applications, but in general, MDS matrices are not sparse and have large description resulting costly implementations in hardware and software. It is non trivial to find efficient MDS matrices which are suitable for software and/or hardware implementation. The first thing to know is what makes an MDS matrix efficient or suitable in a particular application before proceeding to search for/construct one such. Several different techniques have been studied in this direction, for example circulant matrices (or modifications of circulant matrices) are used in AES [5] or FOX [9]. The paper [8] studies MDS matrices that are suitable for software implementation.

Recently a new approach has been proposed which is more suited for lightweight applications. In this approach it is suggested to use recursive MDS matrix which can be computed as power of a simple companion matrix, *i.e,* an MDS matrix $M = C_g^k$ for some companion matrix corresponding to a polynomial $g(X) \in \mathbb{F}_q[X]$ (see Definition 3). The final output of the diffusion layer $M\mathbf{v}$ can be obtained by applying C_g iteratively for k times. Known examples that employ such matrices are PHOTON family of hash functions [6] and LED block cipher [7]. Following the work of Guo et. al [6] some ad-hoc techniques have been proposed to search for good matrices suitable for software/hardware implementation. The search techniques of Sajadieh et. al. [12] and Wu et. al. [13] use symbolic computations to find good recursive MDS matrices, finally the indeterminate is replaced with a linear operator of $\mathbb{F}_q$. Then Augot et. al. [1] proposed a method to get rid of expensive symbolic computations. The main advantage of companion matrices is that they can be efficiently implemented in hardware using LFSRs where the last row (or column) gives the connection polynomial of LFSR.

In 2013, Berger [4] proposed a method to construct recursive MDS matrix from Gabidulin Codes. Then in FSE-2014, Augot and Finiasz [2, 3], gave another construction of recursive MDS matrices using shortened BCH codes. They described an algorithm which would explore all BCH codes to find all recursive MDS matrices that can be obtained for given parameters. This naively approach becomes impractical to search for recursive MDS matrices for bigger choices of parameters. In fact, it was mentioned in [3, page 13]:

> *"The reader might note the absence of larger fields in Table 2. One could for example want to obtain a 128-bit diffusion layer using $k = 8$ symbols of $s = 16$ bits. However, going through all the possible values of z and l takes too long with $q = 2^{16}$."*

Our Contribution: A recursive MDS matrix of size k over $\mathbb{F}_q$ is of the form $M = C_g^k$ for some companion matrix corresponding to a polynomial $g(X) \in \mathbb{F}_q[X]$ of degree k. In this paper we first show that if the polynomial $g(X)$ has no multiple of weight $\leq k$ and degree $\leq 2k-1$ then the matrix $M = C_g^k$ satisfies MDS property (see Theorem 1). This characterization answers the issues raised by Augot *et. al.* in [2, Section 5.3] to some extent (see Remarks 1 & 2). In fact,

we show that the corresponding MDS code $\mathcal{M}$ with generator matrix $[M|I]$ is a shortened code of the cyclic code Γ generated by $g(X)$ (code words of $\mathcal{M}$ are given by the multiples of $g(X)$ with degree at most $2k-1$). In such a case, it may happen that the cyclic code Γ has minimum distance $\leq k+1$, and the code $\mathcal{M}$ corresponding to the matrix M is MDS, *i.e.*, minimum distance of $\mathcal{M}$ is equal to $k+1$. We also show that if the cyclic code Γ has minimum distance $k+1$ then its generating polynomial $g(X)$ yields a recursive MDS matrix $M = C_g^k$. For this reason, Augot *et. al.* use BCH codes (which are cyclic codes with guaranteed minimum distance) to construct recursive MDS matrices. They described an algorithm which tests all possible BCH codes of length $2k$ to $q+1$ to find all recursive MDS matrices that can be obtained by shortening for given parameters k and $q = 2^s$.

The algorithm of Augot *et. al.* simply tests all possible candidates without trying to be smart about which could be eliminated faster or completely. In this paper we identify exactly what candidates yield recursive MDS matrices, and thus we can directly compute all the generating polynomials in this class. As a consequence we will be able to provide formula for the number of recursive MDS matrices that can be obtained using shortened BCH codes. With our approach it is possible to search exhaustively for parameter choices $s = 16$ and $k = 8$, which was not the case earlier. Using some precomputed tables, our algorithm to find all recursive MDS matrices in this class has time complexity at most $O(qk^2(\log q)^2 + q^2 k \log q)$. These ideas can also be adapted to search for good matrices in this class with an appropriate cost function/criterion suitable for software/hardware or any other platform.

We first present some preliminaries in Section 2. We then study the relationship between cyclic codes and recursive MDS matrices and give a brief overview of the construction method of Augot *et. al.* in Section 3. We discuss our other main results in Section 4. We present our algorithm and an efficient strategy to search for good matrices in Section 5.

2 Definition and Preliminaries

Let us first fix some notation used throughout the paper. Let s be a positive integer and $\mathbb{F}_q$ denote the finite field of q elements, where $q = 2^s$, and denote the order of an element $\beta \in \mathbb{F}_q^*$ by $\mathrm{ord}(\beta)$. Let n be an odd integer and denote the group of n-th roots of unity over $\mathbb{F}_q$ by U_n. Let $\alpha \in U_n$ be a generator. Then the set of all generators of U_n is given by $\mathcal{G}_n = \{\beta_{n,i} = \alpha^i \ : \ 1 \leq i \leq n-1 \text{ and } \gcd(i,n) = 1\}$. For simplicity, where it is clear from the context, we omit n and use β_i instead of $\beta_{n,i}$. Let $e = ord_n(2)$ be the multiplicative order of 2 modulo n and let $s' = \mathrm{lcm}(e,s)$ and $e' = s'/s$. Then $\mathbb{F}_{q^{e'}} (\approx \mathbb{F}_{2^{s'}})$ is the smallest finite field containing U_n.

Let $\mathbb{Z}_n$ denote the ring of integers modulo n, and sometimes, where obvious, we treat elements of $\mathbb{Z}_n$ as integers in $\{0, 1, \ldots, n-1\}$. For $A \subseteq \mathbb{Z}$ we denote $A \pmod n = \{a \pmod n : a \in A\} \subseteq \mathbb{Z}_n$. Also for $a \in \mathbb{Z}_n$ and $B \subseteq \mathbb{Z}_n$, the set $a + B$ is given by $\{a + b : b \in B\}$. Similarly, the other usual operations over

subsets of $\mathbb{Z}_n$ are understood to be done element-wise. Also for $S_1, S_2 \subset \mathbb{Z}$ we say $S_1 \equiv S_2 \pmod{n}$ if for each $a \in S_1$ there exists some $b \in S_2$ such that $a \equiv b \pmod{n}$ and vice-versa. Let us now present a few definitions.

Definition 1. *The* cyclotomic coset *modulo n with respect to q containing t is given by*

$$C_t^{(n,q)} = \{tq^i \bmod n : i = 0, 1, \ldots\} \subseteq \mathbb{Z}_n.$$

Let $t' = t/\gcd(t, n)$ and $n' = n/\gcd(t, n)$. Let $e_t^{(n,q)} = ord_{n'}(q)$ be the multiplicative order of q modulo n'. One can see that $e_t^{(n,q)}$ is the smallest positive integer such that $tq^{e_t^{(n,q)}} \equiv t \pmod{n}$. Then $C_t^{(n,q)} = \{t, tq, tq^2, \ldots, tq^{e_t^{(n,q)}-1}\} \pmod{n}$ and its size is equal to $e_t^{(n,q)}$.

Definition 2. *Let γ be an element in some extension of $\mathbb{F}_q$. The* minimal polynomial *of γ over $\mathbb{F}_q$, denoted by $\mathrm{Min}_{\mathbb{F}_q}(\gamma)$, is the lowest degree monic polynomial $\mu(X)$ with coefficients from $\mathbb{F}_q$ such that $\mu(\gamma) = 0$.*

We have the following relation between minimal polynomial and cyclotomic coset.

Lemma 1. *Let $\gamma = \alpha^t \in U_n$. Then we have*

$$\mathrm{Min}_{\mathbb{F}_q}(\gamma) = \prod_{i \in C_t^{(n,q)}} (X - \alpha^i).$$

Definition 3. *Let $g(X) = a_0 + a_1 X + \cdots + a_{k-1} X^{k-1} + X^k$ be a monic polynomial over $\mathbb{F}_q$ of degree k. The* companion matrix *C_g associated to the polynomial g is given by*

$$C_g = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \ldots & \ldots & 1 \\ -a_0 & -a_1 & \ldots & \ldots & -a_{k-1} \end{pmatrix}.$$

In further discussions, we ignore the negative signs in last row of the above matrix as we are working over binary fields. We sometimes use the notation $\mathsf{Companion}(a_0, a_1, \ldots, a_{k-1})$ to represent the companion matrix C_g. The inverse of C_g can be given by

$$C_g^{-1} = \begin{pmatrix} \frac{a_1}{a_0} & \cdots & \frac{a_{k-1}}{a_0} & \frac{1}{a_0} \\ 1 & 0 & \ldots & 0 \\ & \ddots & & \vdots \\ 0 & & 1 & 0 \end{pmatrix}.$$

Observe that if a_0 is equal to 1 then the elements of the last row of C_g and the elements of the first row of C_g^{-1} are same, and we say such solutions are *regular*. Another interesting type is *symmetric* solutions provided by monic self-reciprocal polynomials defined below. In fact symmetric solutions are regular.

Definition 4. *The* reciprocal f^* *of a polynomial* $f = a_0 + a_1X + \cdots + a_tX^t \in \mathbb{F}_q[X]$ *of degree* t *is defined by*

$$f^* = X^t f(1/X) = a_0X^t + a_1X^{t-1} + \cdots + a_t.$$

A polynomial is called self-reciprocal *if it coincides with its reciprocal.*

If f is self-reciprocal then we get $a_j = a_{t-j}$, and also if β is a root of f then β^{-1} is also a root of f.

2.1 Cyclic Codes

We now recall some concepts from coding theory. For more details refer to [11]. A linear code Γ of length n and dimension k over $\mathbb{F}_q$ is denoted as an $[n,k]_q$ code. If the minimum distance of Γ is equal to d then we denote it as an $[n,k,d]_q$ code. If Γ is an $[n,k,d]_q$ code, then $n-k \geq d-1$, and this bound is known as *Singleton Bound*. For an $[n,k]_q$ code Γ its generator matrix G is of size $k \times n$, and we say G is in systematic form when it contains (usually on the left most positions) the $k \times k$ identity matrix I_k. The redundancy part of G is the $k \times (n-k)$ matrix next to I_k. For convenience, in the discussions below, abusing the conventional notation, we place the identity matrix on the right side.

An $[n,k]_q$ is said to be cyclic if a cyclic shift of any element of the code remains in the code. In algebraic terms, cyclic codes can be seen as ideals of $\mathbb{F}_q[X]/(X^n - 1)$. That is each cyclic code Γ can be defined by a generator polynomial $g(X)$ such that $\Gamma =< g(X) >$ and $g(X)$ divides $X^n - 1$. The elements of the code are given by the multiples of $g(X)$ with degree less than n. Then the code Γ defined by $g(X)$ has dimension $k = n - \deg(g)$. A generator matrix of the code Γ can be given by

$$G_1 = \underbrace{\begin{pmatrix} g(X) \\ Xg(X) \\ \vdots \\ X^{n-k-1}g(X) \end{pmatrix}}_{\text{size } n},$$

where the polynomials $X^i g(X)$ are treated as vectors of length n formed by their coefficients. Let

$$G = \begin{pmatrix} \underbrace{\begin{matrix} -X^k \mod g(X) \\ -X^{k+1} \mod g(X) \\ \vdots \\ -X^{n-1} \mod g(X) \end{matrix}}_{\text{size } k} & \vline & \underbrace{\begin{matrix} 1\ 0\ 0 \ldots 0 \\ 0\ 1\ 0 \ldots 0 \\ \ddots \\ 0\ 0\ 0 \ldots 1 \end{matrix}}_{\text{size } n-k} \end{pmatrix}. \quad (1)$$

One can see that G is also a generator matrix of the code Γ, and it can be derived from G_1 by elementary row operations. Note that the matrix G is in systematic form. Since we are working over binary fields, we can ignore the negative signs in the right hand side of the above matrix, and so we do not mention them in further discussions.

2.2 MDS Matrices

The linear codes which satisfy Singleton bound are called *maximum distance separable* or MDS for short. If the linear code Γ is MDS then the redundancy part of its generator matrix is called as an MDS matrix. The following definition of MDS code and MDS matrix provides another interesting property of MDS matrix.

Definition 5. *An $[n, k, d]_q$ code Γ with generator matrix $G = [M|I]$, where M is a $k \times (n-k)$ matrix, is MDS if and only if every square submatrix of M is nonsingular. We say M is an MDS matrix if the corresponding code Γ is MDS.*

Fact 1 *Let M be a square matrix of size k over $\mathbb{F}_q$. Then M is MDS if and only any k columns of G are linearly independent over $\mathbb{F}_q$. Also, M is MDS if and only any k rows of $\bar{G} = \left[\frac{I}{M}\right]$ are linearly independent over $\mathbb{F}_q$.*

The major application of MDS matrices in cryptography is the design of linear diffusion layers in block ciphers and hash functions. They provide maximal diffusion. The input and output of diffusion layers in those applications are generally of same size. So we are interested in only square matrices which are MDS. We also have $d = k+1$ if Γ is MDS. And thus our interest is to construct $[2k, k, k+1]_q$ codes from which we can get an MDS matrix of size $k \times k$ over $\mathbb{F}_q$.

3 Recursive MDS Matrices and Cyclic Codes

A recursive MDS matrix is an MDS matrix which can be computed as power of a simple companion matrix, *i.e,* an MDS matrix $M = C_g^k$ for some companion matrix corresponding to a polynomial $g(X) \in \mathbb{F}_q[X]$ of degree k. Then we say that the polynomial $g(X)$ yields a recursive MDS matrix. The following result gives a characterization of the polynomials that yield recursive MDS matrices.

Theorem 1. *Let $g(X) \in \mathbb{F}_q[X]$ be a polynomial of degree k. Then the matrix $M = C_g^k$ is MDS if and only if the polynomial $g(X)$ has no multiple with weight $\leq k$ and degree $\leq 2k-1$.*

Proof. Note that

$$C_g = \begin{pmatrix} X \\ X^2 \\ \vdots \\ X^{k-1} \\ \underbrace{X^k \bmod g(X)}_{\text{size } k} \end{pmatrix} \Rightarrow C_g^2 = \begin{pmatrix} X^2 \\ X^3 \\ \vdots \\ X^k \bmod g(X) \\ \underbrace{X^{k+1} \bmod g(X)}_{\text{size } k} \end{pmatrix} \Rightarrow C_g^k = \begin{pmatrix} X^k \bmod g(X) \\ X^{k+1} \bmod g(X) \\ \vdots \\ X^{2k-2} \bmod g(X) \\ \underbrace{X^{2k-1} \bmod g(X)}_{\text{size } k} \end{pmatrix}$$

By Fact 1, the matrix C_g^k is MDS if and only if any k rows of the matrix $\bar{G} = \left[\frac{I}{C_g^k}\right]$

are linearly independent. We can interpret the matrix $\bar{G}$ as follows.

$$\bar{G} = \left[\frac{I}{C_g^k}\right] = \begin{pmatrix} 1 \\ \vdots \\ X^{k-1} \\ X^k \bmod g(X) \\ \vdots \\ \underbrace{X^{2k-1} \bmod g(X)}_{\text{size } k} \end{pmatrix}.$$

Now one can easily check that the latter condition is equivalent to: $g(X)$ has no multiple with weight $\leq k$ and degree $\leq 2k-1$. Alternatively, let $\mathcal{M}$ be the code with generator matrix

$$G_{2k} = \begin{pmatrix} g(X) \\ Xg(X) \\ \vdots \\ \underbrace{X^{k-1}g(X)}_{\text{size } 2k} \end{pmatrix}.$$

The elements of $\mathcal{M}$ correspond to the multiples of $g(X)$ with degree less than $2k$. One can see that the matrix $G' = [C_g^k | I]$ is also a generator matrix of the code $\mathcal{M}$ (which is in systematic form) which can be derived from G_{2k} by elementary row operations. Hence the proof. □

If suppose the cyclic code Γ defined by $g(X) \in \mathbb{F}_q[X]$ of degree k has minimum distance $k+1$ then the latter condition of Theorem 1 is true. In that case Γ is also MDS, and the polynomial $g(X)$ yields a recursive MDS matrix, and so $\mathcal{M}$ is an $[2k, k, k+1]_q$ code. We now give another interpretation in terms of shortened code defined below.

Definition 6. *Given a $[n, k, d]_q$ code Γ, and a set R of z indices $\{i_1, \ldots, i_z\}$, the* shortened code *Γ_R is the set of words from Γ which are zero at positions $i_1, \ldots, i_z$ and whose zero coordinates are deleted, thus effectively shortening these words by z positions. The shortened code Γ_R has length $n-z$, dimension $\geq k-z$ and minimal distance $\geq d$.*

Let $\mathcal{M}$ be the code with generator matrix $G' = [C_g^k | I]$. We can check from (1) that the generator matrix G of the cyclic code $\Gamma =< g(X) >$ contains G' (the submatrix formed by the first k rows and $2k$ columns of G). Thus the code $\mathcal{M}$ can be seen as a shortened code of the cyclic code Γ (shortening on the last $n-2k$ positions by taking G as a generator matrix of Γ).

Remark 1. Due to the Signleton bound, the minimum distance of Γ must be at most $k+1$. But it may happen that the cyclic code Γ has minimum distance strictly less than $k+1$, and the (shortened) code $\mathcal{M}$ corresponding to the matrix $M = C_g^k$ is MDS, *i.e.*, minimum distance of $\mathcal{M}$ is equal to $k+1$. For example the 4×4 matrix used in PHOTON is obtained from the polynomial $f(X) = 1 + \alpha X + X^2 + \alpha^2 X^3 + X^4$, where α is the constructing element of $\mathbb{F}_{2^8}$ (root of the polynomial $Z^8 + Z^4 + Z^3 + Z + 1 \in \mathbb{F}_2[Z]$). We have verified using SAGE

software that the cyclic code generated by $f(X)$ is of length $2^{16}-1$ over $\mathbb{F}_{2^8}$ which has minimum distance 3, but the shortened code has minimum distance 5. As it is necessary the polynomial $f(X)$ has no multiple with weight ≤ 4 and degree ≤ 7, we have verified it using SAGE software.

Remark 2. We believe that, in terms of complexity, in general testing the condition of Theorem 1 for a given polynomial $g(X)$ of degree k is equivalent to testing MDS property of the matrix C_g^k. However, this condition can be used to explore various other new/known techniques which can be fitted directly like BCH codes and Gabadulin codes. Recall that if an $[n,k,d]_q$ cyclic code is MDS ($d=k+1$) then its generating polynomial yields a recursive MDS matrix. To some extent, this and the previous remark answer the issues raised by Augot *et. al.* in [2, Section 5.3].

Note also that a multiplication by a the companion matrix C_g can also be expressed in terms of LFSR, and full cyclic codes can be generated using such structures (elements of the cyclic code are essentially the periodic part of the sequences generated by the LFSR with feedback polynomial $g(X)$). The concept of LFSR is well-studied, see for example [10, Chap. 8]. And the required $[2k,k,k+1]_q$ MDS codes (which can be implemented recursively) can be viewed as cut short version of cyclic codes (with a constraint on the generator polynomial $g(X)$). We now present the method of Augot *et. al.* to construct such codes from the well-known class of cyclic codes : BCH codes.

3.1 Constructuion of Recursive MDS Matrices using Shortened BCH Codes

In this section we present an overview of the technique proposed in [3] for the construction of recursive MDS matrices by shortening the suitable BCH codes appropriately. BCH codes are a special type of cyclic codes with guaranteed minimum distance. For more details, refer to [3, Section 3] and for related concepts in coding theory refer to [11].

Definition 7. *A BCH code over $\mathbb{F}_q$ is defined using an element β in some extension field of $\mathbb{F}_q$. First, pick integers ℓ and d and take the $(d-1)$ consecutive powers $\beta^\ell, \beta^{\ell+1}, \ldots, \beta^{\ell+d-2}$ of β, then compute*

$$g(X) = \operatorname{lcm}\left(\operatorname{Min}_{\mathbb{F}_q}(\beta^\ell), \ldots, \operatorname{Min}_{\mathbb{F}_q}(\beta^{\ell+d-2})\right),$$

where $\operatorname{Min}_{\mathbb{F}_q}(\gamma)$ is the minimal polynomial of γ over $\mathbb{F}_q$. The cyclic code over $\mathbb{F}_q$ of length $\operatorname{ord}(\beta)$ defined by $g(X)$ is called a BCH code*, and it has dimension $(\operatorname{ord}(\beta) - \deg(g))$ and has the minimal distance at least d.*

For such a BCH code to be MDS, $g(X)$ must have degree $(d-1)$. In that case $g(X)$ cannot have any other roots except $\beta^\ell, \ldots, \beta^{\ell+d-2}$. This means that all the conjugates of the elements in $\{\beta^{\ell+i} : i = 0, 1, \ldots, d-2\}$ must be in that set itself. We call such a cyclic code as an *MDS BCH code*.

The MDS BCH code discussed above has length $\text{ord}(\beta)$ and dimension $k = (\text{ord}(\beta) - \deg(g))$. So the corresponding MDS matrix will be of size $\deg(g) \times k$, which may not be suitable for use as a diffusion layer, unless $\deg(g) = k$, as the sizes of input and output of diffusion layer are generally same. Also we can not have $\text{ord}(\beta) = 2k$ since we can not have elements of even order in extensions of $\mathbb{F}_2$. To overcome this problem it is suggested in [3, Section 3.2] to use a shortened MDS BCH code (see Definition 6) instead of a full length ($\text{ord}(\beta) > 2k$) MDS BCH code. From the discussion above, we can see that such an MDS BCH code yield a recursive MDS matrix of size k. We now discuss the technique proposed in [3] to find recursive MDS matrices of size k from MDS BCH codes by shortening appropriately.

The idea is to look for $[2k + z, k + z, k + 1]_q$ MDS BCH codes and shorten them on z positions to obtain the required $[2k, k, k+1]_q$ MDS codes for some odd integer z. So the first step is to construct a BCH code of length $n = 2k+z\ (\leq q+1)$ for some odd integer z. The upper bound on n is coming from the assumption that MDS conjecture holds (see [3, p. 2]). We pick a $\beta \in \mathcal{G}_n$ of order n (in some extension filed of $\mathbb{F}_q$) and ℓ, $0 \leq \ell < n$, to construct a BCH code (see Definition 7). The following lemma gives a condition when such a BCH code becomes MDS. The second step is to check whether the condition is satisfied and if so we can obtain a recursive MDS matrix from the generating polynomial.

Lemma 2. *([3, Lemma 1]) A BCH code Γ over $\mathbb{F}_q$ defined by the k roots $[\beta^\ell, \ldots, \beta^{\ell+k-1}]$ is MDS, if and only if $P(X) = \prod_{j=0}^{k-1}(X - \beta^{\ell+j})$ is in $\mathbb{F}_q(X)$. In this case, $g(X) = \text{lcm}\left(\text{Min}_{\mathbb{F}_q}(\beta^\ell), \ldots, \text{Min}_{\mathbb{F}_q}(\beta^{\ell+k-1})\right)$ is equal to $P(X)$.*

Thus essentially one needs to verify the condition:

$$P(X) = \prod_{j=0}^{k-1}(X - \beta^{\ell+j}) \in \mathbb{F}_q(X); \tag{2}$$

if true then our choice of n, β and ℓ yields an MDS BCH code and its generating polynomial is equal to $P(X)$. Later we will observe that, for some choice of n, if there exits some $\beta \in \mathcal{G}_n$ and ℓ, $0 \leq \ell < n$, together with which yields an MDS BCH code, then for such choice of n, ℓ and for any choice of $\beta \in \mathcal{G}_n$ we can get an MDS BCH code (see Remark 3). Thus for some choice of n if there exists some ℓ, $0 \leq \ell < n$, such that the condition (2) verifies to true, then we say n is a successful choice, and similarly we say the pair (n, ℓ) is a successful choice. To find all MDS BCH codes over $\mathbb{F}_q$ that can be obtained in this way, the algorithm of Augot *et. al.* (see [3, Section 4.2]) verifies the condition computing the polynomial $P(X)$, for all the candidates in the ranges given by $n = 2k + z \leq (q + 1)$, $\beta \in \mathcal{G}_n$ and $0 \leq \ell \leq n - 2$, where z is odd.

4 An Efficient Method for Finding All MDS BCH Codes

The major drawback of Augot *et. al.* algorithm is that there can be many unsuccessful choices of n and ℓ in the aforementioned respective ranges. In fact it

may happen that, as we see later in this section, for some choices of n and for any ℓ, $0 \leq \ell < n-1$, there can not be an MDS BCH code. Such choices for n and ℓ do not yield MDS BCH codes, and so the computation/verification of $P(X)$ with such choices is unnecessary. Moreover, for such unsuccessful choices of n, the computation has to be done in extension fields of $\mathbb{F}_q$. So it will be better if we can confine the computation only to the values of n and ℓ for which the condition: $P(X) \in \mathbb{F}_q(X)$ verifies to true, i.e., compute only the polynomials which are generating polynomials of MDS BCH codes. Another drawback of this algorithm is that for some successful choices of n, the same polynomials are computed twice.

In what follows, we present results on the values of n and the values of ℓ correspondingly for which the constructed BCH code becomes MDS. As a consequence we get formula for the number of such MDS BCH codes. In the rest of the section, we assume that $n = 2k + z\,(\leq q+1)$ for some odd integer z. We now start by presenting a few useful lemmas in order to prove our main theorem.

Lemma 3. *Let ℓ, $0 \leq \ell < n$, be an integer and $S = \{\ell, \ell+1, \ldots, \ell+k-1\}$. Let $P(X) = \prod_{j=0}^{k-1}(X - \beta^{\ell+j})$ for some $\beta \in \mathcal{G}_n$. Then $P(X) \in \mathbb{F}_q[X]$ if and only if $S \equiv qS \pmod{n}$.*

Proof. Proof is omitted due to page limitation. □

Remark 3. The second condition: $S \equiv qS \pmod{n}$ in the above lemma involves only n and ℓ, and does not depend on the choice of $\beta \in \mathcal{G}_n$. So for a successful choice pair (n, ℓ) we can get an MDS BCH code from (n, β, ℓ) for any $\beta \in \mathcal{G}_n$.

Now we will first derive a different but equivalent condition which connects Lemma 3, and then see the possible values of n for which we can get an MDS BCH code of length n. We will then also see the possible values of ℓ $(0 \leq \ell < n)$ corresponding to n satisfying (2) and yielding MDS BCH codes of length n. For this purpose we define the following.

Definition 8. *A set $A = \{i_1, \ldots, i_k\} \subseteq \mathbb{Z}_n$ will be called a* consecutive set *of length k modulo n if there exists an integer $j, 0 \leq j < n$, such that $A = \{j, j+1, \ldots, j+k-1\} \bmod n$.*

Obviously, the set $S = \{\ell, \ell+1, \ldots, \ell+k-1\} \bmod n$ is consecutive. Our next lemma provides an useful result connecting Lemma 3.

Lemma 4. *Let ℓ, $0 \leq \ell < n$, be an integer. If the set $S = \{\ell, \ell+1, \ldots, \ell+k-1\}$ satisfies $S \equiv qS \pmod{n}$ then $S' = \{0, q, 2q, \ldots, (k-1)q\} \bmod n$ is a consecutive set of length k modulo n.*

Proof. Suppose there exists some ℓ, $0 \leq \ell < n$, such that the set $S = \{\ell, \ell+1, \ldots, \ell+k-1\}$ satisfies $S \equiv qS \pmod{n}$. Then $qS = \{q\ell, q(\ell+1), \ldots, q(\ell+k-1)\} \pmod{n} = \{\ell, \ell+1, \ldots, \ell+k-1\} \pmod{n}$. Set $\ell' = \ell - q\ell \pmod{n}$ and observe that $S' = qS - q\ell = \{\ell', \ell'+1, \ldots, \ell'+k-1\} \pmod{n}$. □

The following lemma is very crucial in the proof our main theorem.

Lemma 5. *The set* $S' = \{0, q, 2q, \ldots, (k-1)q\} \bmod n$ *is a consecutive set of length* k *modulo* n *if and only if* $q \equiv \pm 1 \pmod{n}$.

Proof. First note that n is odd, $\gcd(q, n) = 1$ and $k < \frac{n}{2}$. Suppose that $q \equiv \pm 1 \bmod n$. Set $\ell' = 0$ if $q \equiv 1 \bmod n$ or $\ell' = n - k + 1$ if $q \equiv -1 \bmod n$. Now we can see that $S' = \{\ell', \ell' + 1, \ldots, \ell' + k - 1\} \bmod n$.

Suppose that $q \not\equiv \pm 1 \bmod n$. Let Q, $-(\lceil \frac{n}{2} \rceil - 1) \leq Q \leq \lfloor \frac{n}{2} \rfloor$, be the integer such that $q \equiv Q \bmod n$. Let $r = \lfloor \frac{n}{|Q|} \rfloor$. We have $\gcd(|Q|, n) = 1$ as $\gcd(n, q) = 1$. So we have $\frac{n}{2} > |Q| \geq 2$ and also $r \geq 2$. We have either $(k-1) \leq r$ or $(k-1) > r$.
Case: $(k-1) \leq r$

Then $S' = \{0, q, 2q, \ldots, (k-1)q\} \bmod n = \{0, Q, 2Q, \ldots, (k-1)Q\} \bmod n$. Note that $|(k-1)Q| < n$ and so the set S' is not consecutive modulo n.
Case: $(k-1) > r$

Let $\hat{S} = \{0, q, rq, (r+1)q\} \bmod n = \{0, Q, rQ, (r+1)Q\} \bmod n$. Let us assume that Q is positive. Observe that $\hat{S} \subset S'$ and $rQ < n < (r+1)Q$. Therefore the least residue of $(r+1)Q$ modulo n, treating it as an integer, will be in between 0 and Q. So if S' is consecutive then we must have $|S'| > n - |Q|$. But $|S'| = k > n - |Q| > \frac{n}{2}$, a contradiction to the fact that $k < \frac{n}{2}$. The other case where Q is negative can be dealt with similarly as $\hat{S} = \{0, n+Q, n+rQ, n+(r+1)Q\} \bmod n$. □

Remark 4. Note that the above lemma is still valid even if we replace q with some q', where $\gcd(q', n) = 1$. Of course we need $\gcd(q', n) = 1$ if otherwise S' can not be a consecutive set of length k modulo n.

We now present our main result which shows that the possible values for $n\,(> 2k)$ yielding MDS BCH codes over $\mathbb{F}_q$ are the divisors of either $(q-1)$ or $(q+1)$.

Theorem 2. *Let* $n > 2k$. *There exists an MDS BCH code of length* n *and of dimension* $(n-k)$ *over* $\mathbb{F}_q$ *if and only if* $q \equiv \pm 1 \bmod n$.

Proof. We can see that from Lemmas 2 & 3, there exists an MDS BCH code of length n and of dimension $(n-k)$ over $\mathbb{F}_q$ if and only if there exists some ℓ, $0 \leq \ell < n$, such that the set $S = \{\ell, \ell + 1, \ldots, \ell + k - 1\}$ satisfies $S \equiv qS \pmod{n}$. And observe that from Lemma 4, such an MDS BCH code can exist if and only if $S' = \{0, q, 2q, \ldots, (k-1)q\} \bmod n$ is a consecutive set of length k modulo n. Now we can see the required result from Lemma 5, and hence the proof. □

4.1 On the Number of MDS BCH Codes

In order to count all such MDS BCH codes of length n over $\mathbb{F}_q$, we count the possible values of ℓ corresponding to n satisfying (2). Below we give the formula for the number of such MDS BCH codes. As discussed above we only need to deal with the cases: *i)* $n|(q-1)$ and *ii)* $n|(q+1)$.

Theorem 3. *Let $n|(q-1)$. Then the number of MDS BCH codes of length n and of dimension $(n-k)$ over $\mathbb{F}_q$ is equal to $n \cdot \frac{\phi(n)}{2}$.*

Proof. In this case, $U_n \subseteq \mathbb{F}_q^*$ and $e_t^{(n,q)} = 1 = |C_t^{(n,q)}|$ for any $t,\ 0 \leq t \leq n-1$. Therefore, for $\beta \in \mathcal{G}_n$ and for any choice of $\ell,\ 0 \leq \ell \leq n-1$, we get an MDS BCH code defined by the k roots $[\beta^\ell, \ldots, \beta^{\ell+k-1}]$ and its generating polynomial is $\prod_{j=0}^{k-1}(X - \beta^{\ell+j})$. For a fixed β, we get n distinct MDS BCH codes from the different choices of $\ell,\ 0 \leq \ell \leq n-1$. Let $\beta_i = \alpha^i \in \mathcal{G}_n$ for $i \in \mathbb{Z}_n^*$ and $\mathcal{S}_i^{(n)}$ denote the set of all polynomials

$$\mathcal{S}_i^{(n)} = \{g_{(i,\ell)}^{(n)}(X) = \prod_{j=0}^{k-1}(X - \beta_i^{\ell+j}) : 0 \leq \ell \leq n-1\}.$$

Suppose $\mathcal{S}_{i_1}^{(n)} \cap \mathcal{S}_{i_2}^{(n)} \neq \emptyset$ for some $i_1, i_2 \in \mathbb{Z}_n^*$. Then there exists $0 \leq \ell_1, \ell_2 \leq n-1$ such that

$$\{\beta_{i_1}^{\ell_1+j}\}_{j=0}^{k-1} = \{\beta_{i_2}^{\ell_2+j}\}_{j=0}^{k-1}$$

which implies $\{i_1(\ell_1+j)\}_{j=0}^{k-1} \equiv \{i_2(\ell_2+j)\}_{j=0}^{k-1} \pmod{n}$. Therefore we get $\{(i_1/i_2)(\ell_1+j)\}_{j=0}^{k-1} \equiv \{\ell_2+j\}_{j=0}^{k-1} \pmod{n}$ and so $\{(i_1/i_2)(\ell_1+j)\}_{j=0}^{k-1}$ is a consecutive set of length k modulo n. By a similar argument as in the proof of Lemma 5 we get $i_1/i_2 \equiv \pm 1 \pmod{n}$. In fact one can check that $\mathcal{S}_{i_1}^{(n)} = \mathcal{S}_{i_2}^{(n)}$ if $(i_1/i_2) \equiv \pm 1 \pmod{n}$. So we get $n\frac{\phi(n)}{2}$ distinct MDS BCH codes of length n - given by the polynomials

$$\mathcal{S}^{(n)} = \bigsqcup_{i \in \mathcal{I}_n} \mathcal{S}_i^{(n)},$$

where $\sqcup$ denotes the disjoint union, and $\mathcal{I}_n \subset \mathbb{Z}_n^*$ with $|\mathcal{I}_n| = \frac{\phi(n)}{2}$ and $\nexists\, i_1, i_2 \in \mathcal{I}_n$ such that $i_1 = n - i_2$. We can take for example $\mathcal{I}_n = \{i : 1 \leq i \leq \frac{n-1}{2} \text{ and } \gcd(i,n) = 1\}$. □

The following result gives the number of regular solutions and symmetric solutions which are of practical interest. It is not difficult to check.

Corollary 1. *Let $n|(q-1)$ and $k' = \gcd(k,n)$. Then the number of regular solutions in $\mathcal{S}^{(n)}$ is equal to $k' \cdot \frac{\phi(n)}{2}$ and the number of symmetric solutions in $\mathcal{S}^{(n)}$ is equal to $\frac{\phi(n)}{2}$.*

We now present the formula for the number of MDS BCH codes in the other case: $n|(q+1)$.

Theorem 4. *Let $n|(q+1)$. Then the number of MDS BCH codes of length n and of dimension $(n-k)$ over $\mathbb{F}_q$ is $\frac{\phi(n)}{2}$. Moreover, all the polynomials obtained in this case are self-reciprocal (and so yield symmetric solutions).*

Proof. We have $\gcd(n, q-1) = 1$ since $n|(q+1)$. So, in this case, $U_n \subseteq \mathbb{F}_{q^2}^*$ and $e_t^{(n,q)} = 2 = |C_t^{(n,q)}|$ for any $t,\ 1 \leq t \leq n-1$. Also $U_n \cap \mathbb{F}_q^* = \{1\}$ and $e_0^{(n,q)} = 1$.

Note that, for $\beta_i = \alpha^i \in \mathcal{G}_n$ its conjugate is $\beta_i^{-1} = \beta_{n-i} = \alpha^{n-i}$. Therefore the BCH code defined by the k roots $[\beta_i^{\ell}, \ldots, \beta_i^{\ell+k-1}]$ is MDS if and only if $\ell = \frac{n-k+1}{2}$ when k is even or $\ell = n - \frac{k-1}{2}$ otherwise. Also observe that both β_i and β_{n-i} give the same polynomial $g_i^{(n)}(X) = \prod_{j=0}^{k-1}(X - \beta_i^{\ell+j}) = \prod_{j=0}^{k-1}(X - \beta_{n-i}^{\ell+j}) = g_{n-i}^{(n)}(X)$ and hence the proof. □

Remark 5. Observe that, for a successful choice of n, if there exists some $\ell, \ell' \in \mathbb{Z}_n$ and $i_1, i_2 \in \mathbb{Z}_n^*$ with $i_1 \neq i_2$ such that both (n, β_{n,i_1}, ℓ) and $(n, \beta_{n,i_2}, \ell')$ yield the same polynomial if and only if (i) $i_1 = n - i_2$ and (ii) $\ell' = n - \ell$ if $n|(q-1)$ and in the other case $\ell' = \ell$.

We get a recursive MDS matrix from the generating polynomial of an MDS BCH code. So the total number $\mathcal{N}_k$ of recursive MDS matrices of size $k \times k$ over $\mathbb{F}_q$ that are coming from BCH codes is given by

$$\mathcal{N}_k = \sum_{n>2k,\, n|(q-1)} n\frac{\phi(n)}{2} + \sum_{n>2k,\, n|(q+1)} \frac{\phi(n)}{2}. \tag{3}$$

5 Faster Algorithm to Find All MDS BCH Codes

We can see from the proofs of Theorem 3 & 4 that the set of all MDS BCH codes over $\mathbb{F}_q$ can be obtained from the following choices: the length n of MDS BCH codes must be a divisor of either $(q-1)$ or $(q+1)$, and also $n > 2k$;

- if $n|(q-1)$ then the triplets $(n, \beta_{n,i}, \ell)$, where $1 \leq i \leq \frac{n-1}{2}$ with $\gcd(i, n) = 1$ and $0 \leq \ell \leq n-1$, yield distinct MDS BCH codes and the corresponding generating polynomials are given by

$$g_{(i,\ell)}^{(n)}(X) = \prod_{j=0}^{k-1}(X - \beta_{n,i}^{\ell+j}); \tag{4}$$

- if $n|(q+1)$ then the triplets $(n, \beta_{n,i}, \ell)$, where $1 \leq i \leq \frac{n-1}{2}$ with $\gcd(i, n) = 1$ and $\ell = \frac{n-k+1}{2}$ if k is even or otherwise $\ell = n - \frac{k-1}{2}$, yield distinct MDS BCH codes and the corresponding generating polynomials are given by

$$g_i^{(n)}(X) = \prod_{j=0}^{k-1}(X - \beta_{n,i}^{\ell+j}). \tag{5}$$

In the above, the set of choices for i can be any subset $\mathcal{I}_n \subset \mathbb{Z}_n^*$ as mentioned in the proof of Theorem 3. We take $\mathcal{I}_n \subset \mathbb{Z}_n^*$ to be as mentioned above.

This makes us to have an efficient algorithm to find all MDS BCH codes over $\mathbb{F}_q$; we just need to compute the generating polynomials $g_{(i,\ell)}^{(n)}$ and $g_i^{(n)}$ (which are of degree k) for the suitable choices of n, i and ℓ mentioned above. Thus we can obtain recursive MDS matrices of size $k \times k$ from these polynomials. But instead we first present an observation by considering coefficient-wise product

of polynomials which we will introduce below. We take this in to consideration to present our algorithm which can speed-up the computation of the generating polynomials in (4) and (5). These observations also suit our strategy to search for optimal/good recursive MDS matrices in this class which we discuss in Section 5.2. We will first introduce the mentioned product notation.

Definition 9. *Let* $f(X) = a_0 + a_1X + \cdots + a_rX^r$ *and* $f'(X) = b_0 + b_1X + \cdots + b_rX^r$ *be two polynomials over* $\mathbb{F}_q$*. We define the coefficient-wise product, denoted by* $\langle f, f' \rangle$*, of* f *and* f' *by*

$$\langle f, f' \rangle = a_0b_0 + a_1b_1X + \cdots + a_rb_rX^r.$$

The following lemma is easy to check.

Lemma 6. *Let* $n|(q-1)$ *and* $\beta \in \mathcal{G}_n$*. Let* $h(X) = \beta^k + \beta^{k-1}X + \cdots + \beta X^{k-1} + X^k \in \mathbb{F}_q[X]$*. Suppose* $f(X) = \prod_{j=0}^{k-1}(X - \beta^{\ell+j})$ *and* $f'(X) = \prod_{j=0}^{k-1}(X - \beta^{\ell+1+j})$*. Then we have* $f' = \langle f, h \rangle$*.*

Remark 6. Let $n|(q-1)$ and $i \in \mathcal{I}_n$ and we set $h_{n,i}(X) = \beta_{n,i}^k + \beta_{n,i}^{k-1}X + \cdots + \beta_{n,i}X^{k-1} + X^k \in \mathbb{F}_q[X]$. From the above lemma one can easily see that

$$g_{(i,\ell)}^{(n)} = \langle g_{(i,\ell-1)}^{(n)}, h_{n,i} \rangle \text{ for } 1 \leq \ell \leq n-1,$$

and so all these polynomials can be computed recursively from $g_{(i,0)}^{(n)}(X) = \prod_{j=0}^{k-1}(X - \beta_{n,i}^j)$, and each such product requires k field element multiplications.

5.1 Implementation Issues and Complexity

The algorithm of Augot *et. al.* simply tests all the candidates without trying to be smart about which could be eliminated faster or completely. It also finds each solution several times. On top of that the operations have to be done in extension fields of $\mathbb{F}_q$ and so it multiplies the complexity.

In our Algorithm 1, we compute only the generating polynomials of the MDS BCH codes that provide recursive MDS matrices of size $k \times k$. The total number $\mathcal{N}_k$ of such MDS BCH codes over $\mathbb{F}_q$, where $q = 2^s$, is given in (3) and it is at most of $O(q^2)$. Most of the generating polynomials (at Line 12) can be computed by just k field element multiplications due to the product notation that we used. So we can see that the upper bound on the total complexity of our Algorithm 1 is of $O(q^2k(\log q)^2)$, whereas the upper bound on the total complexity of the algorithm in [3] is of $O(q^4k(q-2k)(\log q)^2)$.

We can further reduce the complexity substantially by the fallowing trick: first create the two tables described below as part of the pre-computation. The elements of $\mathbb{F}_q^*$ can be represented as integers in the range $\{1, \ldots, q-1\}$ and let θ be a generator of $\mathbb{F}_q^*$.

1. $Expt[i] = \theta^i$ for $i \in \{0, 1, \ldots, q-2\}$;
2. $Dlog[\eta] = i$, where $\eta = \theta^i$.

Algorithm 1: Finding All MDS BCH Codes

input : parameters $k\,(>1)$ and s with $2k \leq 2^s$
output: a set $\mathcal{S}$ of generating polynomials of deg k over $\mathbb{F}_q$ yielding recursive MDS matrices

1 $q \leftarrow 2^s$;
2 $\mathcal{D}^- \leftarrow$ odd divisors of $(q-1)$ greater than $2k$;
3 $\mathcal{D}^+ \leftarrow$ odd divisors of $(q+1)$ greater than $2k$;
4 **for** $n \in \mathcal{D}^-$ **do**
5 $\quad \alpha \leftarrow$ primitive n-th root of unity of $\mathbb{F}_q$;
6 $\quad$ **for** $i \leftarrow 1$ **to** $\frac{n-1}{2}$ **with** $\gcd(i,n)=1$ **do**
7 $\quad\quad \beta \leftarrow \alpha^i$;
8 $\quad\quad h = \beta^k + \beta^{k-1}X + \cdots + \beta X^{k-1} + X^k$;
9 $\quad\quad g_0(X) \leftarrow \prod_{j=0}^{k-1}(X - \beta^j)$;
10 $\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{g_0(X)\}$;
11 $\quad\quad$ **for** $\ell \leftarrow 1$ **to** $n-1$ **do**
12 $\quad\quad\quad g_\ell(X) \leftarrow \langle g_{\ell-1}, h\rangle$;
13 $\quad\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{g_\ell(X)\}$;
14 **for** $n \in \mathcal{D}^+$ **do**
15 $\quad \alpha \leftarrow$ primitive n-th root of unity of $\mathbb{F}_q$;
16 $\quad$ **if** k *is even* **then** $\ell \leftarrow \frac{n-k+1}{2}$;
17 $\quad$ **else** $\ell \leftarrow n - \frac{k-1}{2}$;
18 $\quad$ **for** $i \leftarrow 1$ **to** $\frac{n-1}{2}$ **with** $\gcd(i,n)=1$ **do**
19 $\quad\quad \beta \leftarrow \alpha^i$;
20 $\quad\quad g(X) \leftarrow \prod_{j=0}^{k-1}(X - \beta^{\ell+j})$;
21 $\quad\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{g(X)\}$;

Observe that for $\eta_1, \eta_2 \in \mathbb{F}_q^*$, their product can be obtained from the summation $\eta_1\eta_2 = Expt[(Dlog[\eta_1] + Dlog[\eta_2]) \bmod (q-1)]$ using the precomputed tables. Thus the polynomial product $\langle\cdot,\cdot\rangle$ at Line 12 in Algorithm 1 can be obtained efficiently using the precomputed tables. The time complexity for creating the two tables $Expt[]$ and $Dlog[]$ is of at most $O(q(\log q)^2)$ and require $O(q \log q)$ memory. In this way we need to compute at most of $O(q)$ polynomials and the rest can be obtained using the precomputed tables. In such case the upper bound on the time complexity of Algorithm 1 will be of $O(qk^2(\log q)^2 + q^2 k \log q)$.

Remark 7. We can even further reduce the time complexity by a factor of approximately $s\,(=\log q)$ in the first component of the sum by taking in to consideration: Frobenius mapping, which will reduce the number polynomials actually to be computed by a factor approximately s, and the other polynomials can be obtained through table look-ups.

5.2 Search for Efficient Recursive MDS Matrices

The cost of implementation of a recursive MDS matrix depends mainly on the coefficients of its associated polynomial $g(X) \in \mathbb{F}_q[X]$. We first need to develop a criterion/cost function to implement multiplication by elements of $\mathbb{F}_q$ in a particular platform (software/hardware). For hardware implementation, the cost function can be taken as the number of XORs required to implement $\mathbb{F}_q$-element multiplication. Now one may apply similar strategy as above, that is to precompute two tables: $Dlog[]$ as mentioned above and another table $Cost[]$ with an appropriate cost function/criterion. Note that $Cost[i]$ denotes the cost of implementing multiplication by θ^i. Our Algorithm 1 can be modified to compute the cost of the polynomials without computing the polynomials: instead of computing the polynomials at Line 12 in Algorithm 1, we compute the total cost of its implementation using the precomputed tables. This way the exhaustive search for good/optimal MDS matrices in this class can be done efficiently. We have implemented our algorithm using SAGE. It took approximately 6 hours to exhaustively search this class with $k = 8$ and $s = 16$.

References

1. Augot, D. and Finiasz, M.: Exhaustive search for small dimension recursive MDS diffusion layers for block ciphers and hash functions. In *Proc. of the 2013 IEEE International Symposium on Information Theory*, p. 1551-1555, IEEE, 2013.
2. Augot, D. and Finiasz, M.: Direct Construction of Recursive MDS Diffusion Layers using Shortened BCH Codes. In *FSE 2014*, LNCS, Springer, 2014, to appear.
3. Augot, D. and Finiasz, M.: Direct Construction of Recursive MDS Diffusion Layers using Shortened BCH Codes. http://eprint.iacr.org/2014/566.pdf
4. Berger, T.P.: Construction of Recursive MDS Diffusion Layers from Gabidulin Codes. In *INDOCRYPT 2013*, LNCS, vol. 8250, p. 274-285, Springer, 2013.
5. Daemen, J. and Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. In *Information Security and Cryptography*, Springer 2002.
6. Guo, J., Peyrin, T. and Poshmann, A.: The PHOTON Family of Lightweight Hash Functions. In *CRYPTO 2011*, LNCS, vol. 6841, p. 222-239, Springer, 2011.
7. Guo, J., Peyrin, T., Poshmann, A. and Robshaw, M.J.B.: The LED block cipher. In *CHES 2011*, LNCS, vol. 6917, p. 326-341, Springer, 2011.
8. Junod, P. and Vaudenay, S.: Perfect diffusion primitives for block ciphers. In *SAC 2004*, LNCS, vol. 3357, p. 84-99, Springer, 2004.
9. Junod, P. and Vaudenay, S.: FOX: A new family of block ciphers. In *SAC 2004*, LNCS, vol. 3357, p. 114-129, Springer, 2004.
10. Lidl, R. and Niederreiter, H,: *Finite Fields.* Cambridge University Press, Cambridge, 2nd edition, 1997.
11. MacWilliams, F.J. and Sloane, N.J.A.: *The Theory of Error-Correcting Codes.* North Holland Publishing Co., 1988.
12. Sajadieh, M., Dakhilalian, M., Mala, H. and Sepehrdad, P.: Recursive diffusion layers for block ciphers and hash functions. In *FSE 2012*, LNCS, vol. 7549, p. 385-401, Springer, 2012.
13. Wu, S., Wang, M. and Wu, W.: Recursive diffusion layers for (lightweight) block ciphers and hash functions. In *SAC 2013*, LNCS, vol. 7707, p. 355-371, Springer, 2013.