Received: July 17, 2018

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

New Efficient and Fast Method to Compute the Weights Enumerators of Some Large Doubly Even Self Dual Quadratic Residue Codes

Saïd Nouh[1]* **Moulay Seddiq El Kasmi Alaoui[1]** **Mostafa Belkasmi[2]** **Abdelaziz Marzak[1]**

[1]*Information Technology and Modeling Laboratory, Faculty of Sciences Ben M'sik, Hassan II University, Casablanca, Morocco*
[2]*National School of Computer Science and Systems Analysis, Mohammed V University, Rabat, Morocco*
* Corresponding author's Email: said.nouh@univh2c.ma

Abstract: Quadratic Residue codes are among the best codes. They have high capacity of error correction but they are very difficult to enumerate and therefore to analyse. Despite all developed methods in this domain, the weights enumerators of Quadratic Residue codes are known only for lengths less than or equal to 167. For the lengths 191 and 199 only estimations are available. In this paper, we present a new method based on the Multiple Impulse Method (MIM) and hash techniques to find the weights enumerators of Quadratic Residue codes having lengths in the form 8m-1, for an integer m. The proposed method Hash_MIM_Weights_Enumerators is validated on all Quadratic Residue codes of known weights enumerators; its reduced spatial and temporal complexities yields to new important results. So, the weights enumerators for the lengths 191, 199 and 223 are determined. These three codes are the best binary linear block codes in terms of minimum distance known until today and their analytical performances are remained unknowns in more than 60 years ago and they are available now.

Keywords: Quadratic residue codes, Error-correcting codes, Weights enumerators, MIM method, Hash techniques.

1. Introduction

The weights enumerator of a binary linear code C(n, k) is the polynomial $A(x)=\sum_{i=0}^{n} A_i x^i$, where A_i represents the number of codewords having the weight i, n is the code length and k is its dimension. Finding the polynomial $A(x)$ is a very interesting problem in coding theory [1-3]. The minimum distance d of C is the less non zero weight i for which the coefficient A_i is not null. The problem of finding d is NP-hard [4] and therefore finding $A(x)$ is a more difficult problem, because it requires finding the number of all codewords of each weight.
In [5] we have presented the method PWEH(Partial Weights Enumerator with Hash techniques) for finding an approximation of Partial Weights Enumerator by integration of Hash techniques in the PWE(Partial Weights Enumerator) [6] in order to reduce its temporal complexity. In [7], authors have proposed a method to find only an approximation of the weights enumerators of quadratic residue codes especially for the lengths 191 and 199. In [8], the authors proposed the use of the complete weights enumerator in order to deduce the weights enumerator for linear code. In [9-10], the authors establish the matroid structures corresponding to data-local and local maximally recoverable codes (MRC). In [11] the authors have determined the weights enumerator for the duals of a class of cyclic codes with three zeros, few months later, in [12] they have generalized their method for the codes whose duals have 2i zeros, where $\left(2\leq i\leq \frac{j+1}{2}\right) for\ i,j\ \in IN^2$.

In [13] authors have determined the weights enumerators for every irreducible cyclic code of length n over a finite field F_q , in the case which each prime divisor of n is also a divisor of q−1.
In [14] the authors have used Gauss periods to determine weight distribution of some cyclic codes.

In [15], the authors obtained the weight distribution and constructed some classes of cyclic codes whose duals have two Niho type zeroes; the advantage of the class thus constructed is that it contains optimal cyclic codes with two or three non-zeros weights.
For a linear block code over a Binary Symmetric Channel (BSC) with a transition probability p, the upper bound of decoding error probability [16] is given by the Eq. (1).

$$P_e(C) \leq \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i} \qquad (1)$$

Where t is the code correcting capacity
Proakis [17] exposes that the transition probability p can be formulated as in Eq. (2):

$$p = Q\left(\sqrt{2R\frac{E_b}{N_0}}\right) and\ Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^{\infty} e^{-z^2/2}\,dz \qquad (2)$$

Where R represent the code rate ($R = \frac{k}{n}$) and $\frac{E_b}{N_0}$ represents the ratio signal/noise.

On a Gaussian channel AWGN (Additive white Gaussian noise) an upper bound about decoding error probability [16] is given by Eq. (3).

$$P_e(C) \leq \sum_{w=d}^{n} A_w\, Q(\sqrt{2wR\frac{E_b}{N_0}}) \qquad (3)$$

The authors of [18] have demonstrated that for a systematic linear block code over a decoded AWGN channel by the maximum likelihood decoder (MLD) algorithm, the binary error probability $P_e(C)$ has the following upper bound Eq. (4):

$$P_e(C) \leq P_a = \sum_{w=0}^{n} \frac{wA_w}{n} Q(\sqrt{2wR\frac{E_b}{N_0}}) \qquad (4)$$

The bound P_a represents the analytical performances over the AWGN channel for the code C. Despite the various works proposed in this field, the weights enumerators of several codes are still unknown, for example the largest Quadratic Residue code QR(n) of which the weights enumerator is known is that of length 167 [19-22]. The best linear codes known today are given in [23]; This web site regularly updated contains for each length n less than 256 and each dimension k the best known code of highest minimum distance called the lower bound LB.

Table 1. Comparison between lower bounds and the minimum distances of some QR codes

n	k	Minimum weight of QR(n)	Lower Bound
191	96	27	27
199	100	31	31
223	112	31	31

Before presenting QR codes, we give in table 1 a comparison between the values of LB and those of the minimum distances of some quadratic residue codes of lengths up to 223.

The minimum distances of quadratic residue code of lengths 191, 193, 199 and 223 are respectively 27, 27, 31 and 31 [24-25]. Those of higher lengths are given in [26-29]; they are found by using the Multiple Impulse Method (MIM) and its improvements.

It is well known that in the binary case all Extended Quadratic Residue codes (EQR) of lengths in the form 8.m are doubly even self-dual and all EQR codes with lengths in the form 8m+1 are formally self-dual [30]. Let E(n) the weights enumerator of EQR(n), we have the following equalitie Eq. (5) obtained from the MacWilliams-identity [3]:

$$\forall j \leq n: E_j = 2^{-k}\sum_{i=0}^{n} E_i \sum_{l=0}^{j}(-1)^l \binom{i}{l}\binom{n-i}{j-l} \qquad (5)$$

Let B(n) the binary weights enumerator of EQR(n), $B(x) = \sum_{i=0}^{n} B_i x^i$ where B_i is equal to 1 if there are some codewords of weight i in EQR(n) and it is equal to 0 otherwise. For n in the form 8m-1 where $m \in IN^*$, EQR(n) are doubly even self dual code, therfore if 4 doesn't divide j then $B_j = E_j = 0$.

From $B(n)$ and Eq. (5) we obtain a linear system S(n) of integer variables E_j. The resolution of S(n) permits to considerably reduce the number of unknown values in E.

Let A(n) be, the weights enumerator of the QR(n). by the Pless identity [31] we have:

$$for\ j \leq \frac{n-1}{2}: \quad 2jA_{2j} = (n-(2j-1))A_{2j-1} \qquad (6)$$

By definition of EQR codes and (6) we have:

$$for\ j \leq \frac{n-1}{2}:$$

$$E_{2j} = \frac{n+1}{n+1-2j} A_{2j} = \frac{n+1}{2j} A_{2j-1} \quad (7)$$

The formula (7) permits to deduce A form E and E from A.

Let $c = (c_0, c_1, \ldots\ldots, c_{n-3}, c_{n-2}, c_{n-1})$ a codeword from QR(n) which is cyclic, then $\pi(c) = (c_{n-1}, c_0, c_1, \ldots\ldots, c_{n-3}, c_{n-2})$ and all words $\pi^j(c)$ obtained by shifting c at j time are codewords in QR(n).

$$\forall\, j\epsilon\{1,2,3,\ldots,n-1\}: \pi^j(c) = \underbrace{\pi\left(\pi\left(\ldots\left(\pi(c)\right)\right)\right)}_{j\ time}$$

The codeword c is called of full order if the set $T = \pi^j(c): j\epsilon\{1,2,3,\ldots,n\}$ is of cardinal n and all n codewords obtained by j cyclic shift ($j\epsilon\{1,2,3,\ldots,n\}$) are distincts, otherwise c is called of incomplete order.

The remainder of this paper is organised as follows. In the next section, we describe briefly the method Hash_MIM_Weights_Enumerators proposed in this work. In section 3 we explain how to validate and check the results of Hash_MIM_Weights_Enumerators. In section 4, we give the main new results. In section 5, we present the advantages and strengths of the proposed method, finally, a conclusion and a possible future direction of this research are outlined in section 6.

2. Description of the proposed method Hash_MIM_Weights_Enumerators

Let N be a positive integer that represents the size of the Hash table and a list L of many codewords (only information part) of weight w. All elements of L belong to QR(n). In order to accelerate the search of an element in L, this latest is divided on N sub-sets L[0], L[1],...,L[N-1]; each one contains the words of the same hash value given by the Hash function presented in the algorithm A1 below.

Algorithm A1: The used hash function

```
1   Function hash (word, N)
2   Pos←0
3   For i=1 to the dimension k of the code
4   If word [i] =1 then
5   Pos←Pos + i;
6   End If
7   End For
8   Return (Pos modulo N)
9   EndFunction
```

To check if a codeword(only the information part) exist in the list L or not, we define the Fast_Search_By_Hash function presented in A2 algorithm. We also define, in A3 algorithm, a function that we called NumberOfCodeWordsWithoutCyclicCopie, this function is used to compute the number of codewords of a given weight in QR(n) code by saving a sample for each codewords class.

Algorithm A2: Fast Search of a vector in a list

```
1   Function Fast_Search_By_Hash (L,e)
2   L : a set of binary vectors of length k,
    divided on N sub-sets numbered from 0 to
    N-1.
3   e : a binary vector of length k
4   Outputs:
5   True if e in L and False otherwise.
6   Begin Function
7       h←hash(e,N)
8       If e in L[h] then
9               Return True
10      Else
11              Return False
12      End If
13  End Function
```

Algorithm A3: Find the number of codewords of a given weight in a given QR(n) code

```
1   Function NumberOfCodeWordsWithoutCyclicCopies (w, GEN, Half_Aut)
2    Inputs:
3       -   The weight w
4       -   The generator matrix G of QR(n)
5       -   The half of the sub group Half_Aut of Automorphisms without inverses
6   Begin Function
7       -   Initially the list L is empty : L←[]
8       -   $A_w$←0
9       -   Number_of_iterations ← 0
10      Repeat
11          -   Find a codeword c of weight w by the MIM method
```

```
        -  For each element σ from Half_Aut do:
12      -  If σ(c) is of full order then
13             If there not exists any integer p in [0,1,2,....,n-1]:
14                 Fast_Search_By_Hash(information_part( $\pi^p(\sigma(c))$),L)=True  then
15                     o  Add σ(c) in the list L.
16                     o  $A_w \leftarrow A_w+1$
17             End
18         Else   display("Element of incomplete order is found")
19         End
20      -  Number_of_iterations ← Number_of_iterations +1
21    Until there not exists any element to add in L
22  Outputs: The number N of elements in L
23  End Function
```

In A4 algorithm we present the method Hash_MIM_Weights_Enumerators:

Algorithm A4: hash_MIM_Weights_Enumerators steps

```
Inputs:
-  The length n of the quadratic residue code QR(n).
-  The generator matrix GE(n) of EQR(n).
Outputs:
-  The weights enumerator E(n) and A(n) of the EQR(n) and QR(n) codes.
1. Find the binary weights enumerators B(n) of EQR(n) using GE(n) and the MIM method
2. Create the system S(n) by using B(n) and the MacWilliams identity
3. Solve S(n) to obtain the form of E(n) and that of A(n) and find the list R(n) of the
   residual unknowns of the form $R_{4j}$ which are sufficient to determine A(n) and E(n)
4. Find A(n) as follows:
   For each element $R_{4j}$ in R(n) do
     4.1) find the number $\Gamma_{4j-1}$ of codewords of weight 4j-1 in the QR(n) code without cyclic copies:
       $\Gamma_{4j-1} \leftarrow$NumberOfCodeWordsWithout-CyclicCopies(4j-1, GEN, Half_Aut)
     4.2) $A_{4j-1} \leftarrow n * \Gamma_{4j-1}$
   End For
5. Determine the weights enumerator E(n) by using the Pless identity and A(n).
```

In order to reduce the temporal (run time) and spatial (memory) complexities of the proposed method, the codewords of QR(n) are divided in many classes as it is illustrated in the Fig. 1. Two codewords c and c' are in the same class if it exist an integer u such that $c' = \pi^u(c)$. The idea behind this reduction is to store only one representative element of each class to construct the set L_{4j-1} whose the size is Γ_{4j-1}, then we deduce A_{4j-1} by multiplying Γ_{4j-1} by n.

In order to clarify the proposed method Hash_MIM_Weights_Enumerators steps we give some examples.

Example 1:

The QR(7) code which can be generated by the polynomial g in binary form g={1,0,1,1}.

Firstly, g is used to construct the generator matrices GEN(7) and GE(7) of the QR(7) and EQR(7) codes respectively.

$$G = \begin{pmatrix} 1000101 \\ 0100111 \\ 0010110 \\ 0001011 \end{pmatrix} \qquad GE = \begin{pmatrix} 10001011 \\ 01001110 \\ 00101101 \\ 00010111 \end{pmatrix}$$

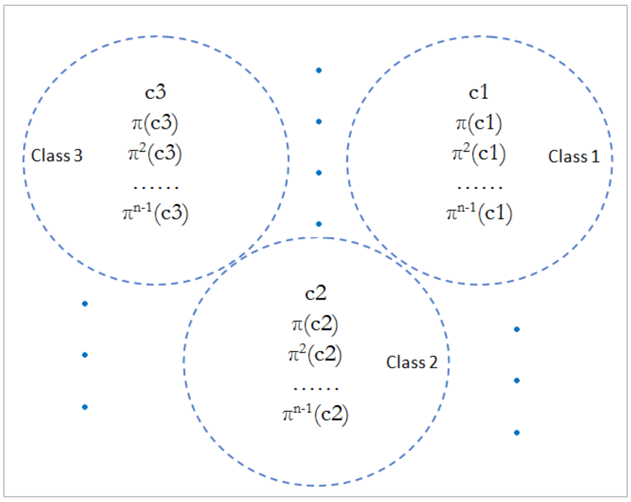

Figure. 1 Representation of cyclic class

Received: July 17, 2018

Table 2. The system S(7)

E[0]=1, E[1]=0,
E[2]=0, E[3]=0,
E[5]=0, E[6]=0,
E[7]=0,E[8]=1,
-4E[6]+2E[3]+8E[0]+6E[1]+4E[2]-6E[7]-8E[8]-2E[5]=0,
4E[6]+14E[7]-2E[3]+4E[2]+14E[1]-2E[5]+28E[0]+28E[8]-4E[4]=0,
-14E[7]-56E[8]-6E[3]+4E[6]+6E[5]-4E[2]+56E[0]+14E[1]=0,
14E[7]+56E[0]-4E[2]-14E[1]+6E[3]-56E[8]-6E[5]+4E[6]=0,
28E[8]-14E[1]+2E[3]-14E[7]-4E[4]+28E[0]+2E[5]+4E[2]+4E[6]=0,
8E[0]+4E[2]+6E[7]-4E[6]-8E[8]-2E[3]+2E[5]-6E[1]=0,

The binary weights enumerator of the EQR(7) code is below:

B={1,0,0,0,1,0,0,0,1}

From B and the MacWilliams identity (5), the system S(7) for the EQR(7) code is presented as Table 2. Solving the system (S) above gives the following solution

{E[0]=1, E[1]=0, E[2]=0, E[3]=0, E[4]=14, E[5]=0, E[6]=0, E[7]=0, E[8]=1}

Which doesn't contains any unknown, therefore the weights enumerators E of EQR(7) is obtained without executing the fourth step of the method Hash_MIM_Weights_Enumerators.

By the formula (7) the weights enumerator A(7) is obtained and it is as follows:

{A[0]=1,A[1]=0, A[2]=0, A[3]=7, A[4]=7, A[5]=0, A[6]=0, A[7]=1}

Example 2:

The QR(71) code can be generated by the polynomial g in binary form

g={1,0,1,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,1,1,0,1,1,0,0,1,1}.

Like in the first example, the generator matrices GEN(71) and GE(71) are constructed, the binary weights enumerator B(71) is found and the system S(71) is made.

By solving S(71), the form of E(71) is obtained and it is given in Table 3. By the Pless identity, the form of A(71) is also obtained and it is given in Table 4.

From Tables 3 and 4, the list R(71) contains only the unknown variable R={R12}={X}. By the step (4.1), the value of Γ11 is 7 and that of A11 is 497. The value of R12 is 2982 and the corresponding enumerators A(71) and E(71) are given respectively in Tables 5 and 6.

Table 3. The form of E(71)

E[12]=6*x
E[16]=249849-72*x
E[20]=18106704+396*x
E[24]=462962955-1320*x
E[28]=4397342400+2970*x
E[32]=16602715899-4752*x
E[36]=25756721120+5544*x

Table 4. The form of A(71)

A[11]=x
A[12]=5*x
A[15]=55522-16*x
A[16]=194327-56*x
A[19]=5029640+110*x
A[20]=13077064+286*x
A[23]=154320985-440*x
A[24]=308641970-880*x
A[27]=1710077600+1155*x
A[28]=2687264800+1815*x
A[31]=7378984844-2112*x
A[32]=9223731055-2640*x
A[35]=12878360560+2772*x

Table 5. The weights enumerator A(71)

A[0]=1
A[11]=497
A[12]=2485
A[15]=47570
A[16]=166495
A[19]=5084310
A[20]=13219206
A[23]=154102305
A[24]=308204610
A[27]=1710651635
A[28]=2688166855
A[31]=7377935180
A[32]=9222418975
A[35]=12879738244

Table 6. The weights enumerator E(71)

E[0]=1
E[12]=2982
E[16]=214065
E[20]=18303516
E[24]=462306915
E[28]=4398818490
E[32]=16600354155
E[36]=25759476488

3. Check of the results of the method Hash_MIM_Weights_Enumerators method (Mykkeltveit congruence check)

3.1 Check by congruence of the number of codewords of a given weight

Let G the projective special linear group G=PSL$_2$(n). In [22],the authors have demonstrated that it is possible to compute the weights enumerator E of EQR(n) modulo $|G| = \frac{n(n^2-1)}{2}$ as follows:

i. Factor $|G|$ in prime numbers $|G| = \prod_{i=1}^{l} {q_i}^{m_i}$, where q$_i$ are prime numbers and m$_i$ is the highest power of q$_i$ that divides $|G|$.

ii. For each divisor $q_i \neq 2$:
   a) Find a permutation g$_i$ of order q$_i$ from G, g$_i$ is a generator of a group S$_i$ called a Sylow q$_i$-subgroup of G.
   b) Find E^{q_i} the weights enumerator of the subcode C$_i$ of EQR(n) fixed by g$_i$.

iii. For the divisor $q_i = 2$:
   a) Find the highest integer m such that 2^m divide $\frac{n+1}{2}$ or $\frac{n-1}{2}$.
   b) Find two permutations a and b verifying : a∈G and b∈G, $a^{2^m} = 1, b^2 = 1, bab = a^{-1}$.
   c) Find F^2 the weights enumerator of the subcode C^2 fixed by: $H_2 = \{1, a^{2^{m-1}}\}$.
   d) Find F^0 the weights enumerator of the subcode C^0 fixed by: $G_4^0 = \{1, a^{2^{m-1}}, b, a^{2^{m-1}}b\}$.
   e) Find F^1 the weights enumerator of the subcode C^1 fixed by: $G_4^1 = \{1, a^{2^{m-1}}, ab, a^{1+2^{m-1}}b\}$.
   f) Find E^2 the weights enumerator of the subcode fixed by S2, a Sylow 2-subgroup of G by :

$$\forall j \leq n: E_j^2 = (2^m + 1)F_j^2 - 2^{m-1}(F_j^0 + F_j^1)$$

iv. For each divisor q$_i$ of $|G|$ and for each integer j less than or equal to n, compute E$_j$ modulo ${q_i}^{m_i}$ according to the following equality:$E_j \ mod \ {q_i}^{m_i} = E_j^{q_i} \ mod \ {q_i}^{m_i}$.

v. For each integer j ≤n, compute E$_j$ modulo $|G|$ by using the Chinese remainder theorem.

3.2 Validation of the Hash_MIM_Weights_ Enumerators method

In order to validate the proposed method, we present here its application on all quadratic residue codes of the form 8m-1 for which these metrics are available.

3.2.1. The QR codes of lengths 7 and 71

For QR codes of lengths 7 and 71, their weights enumerators found by the proposed method as explained in the examples above, coincide with those already known.

3.2.2. The QR codes of lengths n in {23,31,47,79,103}

For these codes, only the three fist steps are required. Solving the system S(n) for these codes yield to find the weights enumerators without any residual unknown variables in the list R(n). The obtained results coincide with the true available values.

3.2.3. The QR code of length 127

The QR(127) code can be generated by the polynomial g in binary form g={ 1,1,1,0,0,1,0,1,0,0,1,0,0,1,0,0,0,0,1,1,0,0,0,0,0,1,0,1,0,0,1,0,0,1,0,1,0,1,1,0,1,0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,0,1,0,0,1}.

The generator matrices GEN(127) and GE(127) are constructed and the system S(127) is made.
By solving S(127), the form of E(127) is obtained and it is given in Table 7. By the Pless identity, the form of A(127) is also obtained and it is given in Table 8.

From Tables 7 and 8, the list R(127) contains only the unknown variable R={R20}={X}. By the step (4.1), the value of Γ$_{19}$ is 70 and that of A$_{19}$ is 8890. The value of R$_{20}$ is 56896 and the corresponding enumerators A(127) and E(127) are given respectively in Tables 9 and 10.

Table 7. The form of E(127)

E[20]=32*x
E[24]=13228320-192*x
E[28]=2940970496-2848*x
E[32]=320411086380+48000*x
E[36]=18072021808640-349600*x
E[40]=552523816524960+1637952*x
E[44]=9491115264030720-5550432*x
E[48]=94116072808107840+14387712*x
E[52]=549827773219608576-29457600*x
E[56]=1920594735166941760+48579200*x
E[60]=4051982995220321280-65302848*x
E[64]=5193576851944293670+72021248*x

In order to validate the implementation of the proposed check, we give in Tables 11 and 12 the Mykkeltveit method results for EQR(127) code and the congruence of the number of codewords of weight 20 for this code. The weights enumerators given in Tables 9 and 10 are then successfully checked.

Table 8. The form of A(127)

A[19]=5*x
A[20]=27*x
A[23]=2480310-36*x
A[24]=10748010-156*x
A[27]=643337296-623*x
A[28]=2297633200-2225*x
A[31]=80102771595+12000*x
A[32]=240308314785+36000*x
A[35]=5082756133680-98325*x
A[36]=12989265674960-251275*x
A[39]=172663692664050+511860*x
A[40]=379860123860910+1126092*x
A[43]=3262570872010560-1907961*x
A[44]=6228544392020160-3642471*x
A[47]=35293527303040440+5395392*x
A[48]=58822545505067400+8992320*x
A[51]=223367532870465984-11967150*x
A[52]=326460240349142592-17490450*x
A[55]=840260196635537020+21253400*x
A[56]=1080334538531404740+27325800*x
A[59]=1899367029009525600-30610710*x
A[60]=2152615966210795680-34692138*x
A[63]=2596788425972146835+36010624*x

Table 9. The weights enumerator A(127)

A[0]=1
A[19]=8890
A[20]=48006
A[23]=2416302
A[24]=10470642
A[27]=642229602
A[28]=2293677150
A[31]=80124107595
A[32]=240372322785
A[35]=5082581311830
A[36]=12988818908010
A[39]=172664602751130
A[40]=379862126052486
A[43]=3262567479655902
A[44]=6228537915706722
A[47]=35293536896047416
A[48]=58822561493412360
A[51]=223367511592873280
A[52]=326460209251122496
A[55]=840260234424082176
A[56]=1080334587116677120
A[59]=1899366974583683328
A[60]=2152615904528174336
A[63]=2596788489999036416

Table 10. The weights enumerator E(127)

E[0]=1
E[20]=56896
E[24]=12886944
E[28]=2935906752
E[32]=320496430380
E[36]=18071400219840
E[40]=552526728803616
E[44]=9491105395362624
E[48]=94116098389459776
E[52]=549827720843995776
E[56]=1920594821540759296
E[60]=4051982879111857664
E[64]=5193576979998072832

Table 11. The Mykkeltveit method results for EQR(127) code.

127	H2	G_4^0	G_4^1	S3	S7	S127
k		17	16	22	10	1
20	64	0	0	7	0	0

Table 12. Congruence of the number of codewords of weight 20 in EQR(127)

n	*|G|*	*w*	*E_w mod |G|*	*Check*
127	1024128	20	56896	Ok

Table 13. The form of E(167)

E[24]=7*y
E[28]=6*x
E[32]=5776211364-168*x+2541*y
E[36]=1251098739072+2268*x-60144*y
E[40]=166068570988089-19656*x+562947*y
E[44]=13047071967014400+122850*x-2761920*y
E[48]=629049676288183920-589680*x+5856697*y
E[52]=19087122102289097472+2260440*x+15900192*y
E[56]=372099732633702386736-7104240*x-188636133*y
E[60]=4739291366078578079232+18648630*x+875355712*y
E[64]=39973673769401063697390-41441400*x-2764837383*y
E[68]=225696676750383595333248+78738660*x+6663305712*y
E[72]=860241110734660092710580-128845080*x-12836992553*y
E[76]=2227390680768729820388352+182530530*x+20247545472*y
E[80]=3935099590080354173030112-224652960*x-26494540443*y
E[84]=4755747408657232763578880+240699600*x+28958598592*y
E[88]=3935099590080354173030112-224652960*x-26494540443*y
E[92]=2227390680768729820388352+182530530*x+20247545472*y
E[96]=860241110734660092710580-128845080*x-12836992553*y

Received: July 17, 2018

E[100]=225696676750383595333248+78738660*x+6663305712*y
E[104]=39973673769401063697390-41441400*x-2764837383*y

Table 14. The form of A(167)

A[23]=y
A[24]=6*y
A[27]=x
A[28]=5*x
A[31]=1100230736-32*x+484*y
A[32]=4675980628-136*x+2057*y
A[35]=268092586944+486*x-12888*y
A[36]=983006152128+1782*x-47256*y
A[39]=39540135949545-4680*x+134035*y
A[40]=126528435038544-14976*x+428912*y
A[43]=3417090277075200+32175*x-723360*y
A[44]=9629981689939200+90675*x-2038560*y
A[47]=179728478939481120-68480*x+1673342*y
A[48]=449321197348702800-21200*x+4183355*y
A[51]=5907918745946625408+699660*x+4921488*y
A[52]=13179203356342472064+1560780*x+10978704*y
A[55]=124033244211234128912-2368080*x-62878711*y
A[56]=248066488422468257824-4736160*x-125757422*y
A[59]=1692604059313777885440+6660225*x+312627040*y
A[60]=3046687306764800193792+11988405*x+562728672*y
A[63]=15228066197867071884720-15787200*x-1053271384*y
A[64]=24745607571533991812670-25654200*x-1711565999*y
A[67]=91353416779917169539648+31870410*x+2697052312*y
A[68]=134343259970466425793600+46868250*x+3966253400*y
A[71]=368674761743425754018820-55219320*x-5501568237*y
A[72]=491566348991234338691760-73625760*x-7335424316*y
A[75]=1007629117490615871128064+82573335*x+9159603904*y
A[76]=1219761563278113949260288+99957195*x+11087941568*y
A[79]=1873856947657311510966720-106977600*x-12616447830*y
A[80]=2061242642423042662063392-117675360*x-13878092613*y
A[83]=2377873704328616381789440+120349800*x+14479299296*y
A[84]=2377873704328616381789440+120349800*x+14479299296*y
A[87]=2061242642423042662063392-117675360*x-13878092613*y
A[88]=1873856947657311510966720-106977600*x-12616447830*y
A[91]=1219761563278113949260288+99957195*x+11087941568*y
A[92]=1007629117490615871128064+82573335*x+9159603904*y
A[95]=491566348991234338691760-73625760*x-7335424316*y
A[96]=368674761743425754018820-55219320*x-5501568237*y
A[99]=134343259970466425793600+46868250*x+3966253400*y
A[100]=91353416779917169539648+31870410*x+2697052312*y
A[103]=24745607571533991812670-25654200*x-1711565999*y

3.2.4. The QR code of length 167

From Tables 13 and 14, the list R(167) contains two unknown variables R={R24, R28}={X, Y}. By the step (4.1), the value of Γ23 is 664 and that of A23 is 110888 and that of Γ27 is 18094 and that of A27 is 3021698. The value of R24 is 776216 and that of R28 is 18130188. Therefore the corresponding enumerators A(167) and E(167) are obtained and they coincide with those already found in [21].

Tables 15 and 16 give the Mykkeltveit method results for the EQR(167) code and the congruence of the number of codewords of weights 24 and 28 for this code. The weights enumerators A(167) and E(167) are then successfully checked.

Table 15. The Mykkeltveit method results for EQR(167) code

167	**H2**	G_4^0	G_4^1	**S3**	**S7**	**S83**	**S167**
k		22	21	28	12	2	1
E[24]	252	6	4	140	0	0	0
E[28]	1812	36	0	0	6	0	0

Table 16. Congruence of the number of codewords of weights 24 and 28 in EQR(167)

n	**|G|**	**w**	E_w	E_w **mod |G|**	**Check**
167	2328648	24	776216	776216	Ok
167	2328648	28	18130188	1829652	Ok

Table 17. The form of E(191)

E[28]=48*x
E[32]=6*y
E[36]=69065734464+11568*x-192*y
E[40]=16681003659936-387072*x+2976*y
E[44]=2638181865286080+4662144*x-29760*y
E[48]=260118707412159120-30019584*x+215760*y
E[52]=16506204128755716672+102079872*x-1208256*y
E[56]=688919563458768198624-7108608*x+5437152*y
E[60]=19261567021963529559744-2055291840*x-

20195136*y
E[64]=366292346792783194741815+13670572032*x+63109800*y
E[68]=479823029129154938804640 0-56511000000*x-168292800*y
E[72]=437537327036943202521038 40+175210813440*x+ 387073440*y
E[76]=280144274178089715889150656-434619319680*x-774146880*y
E[80]=126828970918971772145588 2224+890278318080*x+1354757040*y
E[84]=408246437392952797379480 6080-1533608219520*x-2084241600*y
E[88]=938222403866579312909702 0640+2246629754880*x+2828613600*y
E[92]=154396045640367799744504 36032-2818036032480*x-3394336320*y
E[96]=182248321490698368776989 45680+3037942333440*x+3606482340*y

Table 18. The form of A(191)

A[27]=7*x
A[28]=41*x
A[31]=y
A[32]=5*y
A[35]=12949825212+2169*x-36*y
A[36]=56115909252+9399*x-156*y
A[39]=3475209095820-80640*x+620*y
A[40]=13205794564116-306432*x+2356*y
A[43]=604583344128060+1068408*x-6820*y
A[44]=2033598521158020+3593736*x-22940*y
A[47]=65029676853039780-7504896*x+53940*y
A[48]=195089030559119340-22514688*x+161820*y
A[51]=4470430284871339932+27646632*x-327236*y
A[52]=12035773843884376740+74433240*x-881020*y
A[55]=200934872675474057932-2073344*x+1585836*y
A[56]=487984690783294140692-5035264*x+3851316*y
A[59]=6019239694363602987420-642278700*x-6310980*y
A[60]=13242327327599926572324-1413013140*x-13884156*y
A[63]=122097448930927731580605+4556857344*x+21036600*y
A[64]=244194897861855463161210+9113714688*x+42073200*y
A[67]=1699373228165757074933100-20014312500*x-59603700*y
A[68]=3098857063125792313113300-36496687500*x-108689100*y
A[71]=16407649763885370094538940+65704055040*x+145152540*y
A[72]=27346082939808950157564900+109506758400*x+241920900*y
A[75]=110890441862160512539455468-172036814040*x-306433140*y
A[76]=169253832315929203349695188-262582505640*x-467713740*y
A[79]=528454045495715717273284260+370949299200*x+564482100*y
A[80]=739835663694002004182597964+519329018880*x+790274940*y
A[83]=1786078163594168488535227660-670953596040*x-911855700*y
A[84]=2296386210335359485259578420-862654623480*x-1172385900*y
A[87]=4300186017721821850836134460+1029705304320*x+1296447900*y
A[88]=5082038020943971278260886180+1216924450560*x+1532165700*y
A[91]=7398143853600957071090833932-1350308932230*x-1626452820*y
A[92]=8041460710435822903359602100-1467727100250*x-1767883500*y
A[95]=9112416074534918438849472840+1518971166720*x+1803241170*y

Table 19. Results of Hash_MIM_Weights_ Enumerators method for EQR(191)

QR(191)	Weight w	Sizeof (Half_Aut)	Γ_{w-1}	A_{w-1}	E_w
	28	9073	665	127015	870960
	32	9073	140049	26749359	160496154

Table 20. The Mykkeltveit method results for EQR(191)

191	H2	G_4^0	G_4^1	S3	S5	S19	S191
k		25	24	32	20	6	1
E[28]	144	6	0	0	0	0	0
E[32]	5274	30	42	0	19	0	0

Table 21. Congruence of the number of codewords of weights 28 and 32 in EQR(191)

N	|G|	w	E_w	E_w mod |G|	Check
191	3483840	28	870960	870960	Ok
191	3483840	32	160496154	239514	Ok

4. New results obtained by the proposed Hash_MIM_Weights_Enumerators method

4.1 For the QR(191) code

The forms of E(191) and A(191) are given in Tables 17 and 18. From Tables 17 and 18, the list R(191) contains two unknown variables R={R28, R32}={X, Y}. By the step (4.1), the value of Γ27 is 665 and that of A27 is 127015 and that of Γ31 is 140049 and that of A31 is 26749359. The value of R28 is 870960 and that of R32 is 160496154.

Tables 20 and 21 give the Mykkeltveit method results for the EQR(191) code and the congruence of the number of codewords of weights 28 and 32 for this code. The obtained weights enumerators A(191) and E(191) are then successfully checked.

4.2 For the QR(199) code

The forms of E(199) and A(199) are given in Tables 22 and 23. From Tables 22 and 23, the list R(199) contains one unknown variable R={R32}={X}. By the step (4.1), the value of Γ31 is 40260 and that of A31 is 8011740. The value of R32 is 50073375.

Table 22. The form of E(199)

E[32]=25*x
E[36]=21005534550-450*x
E[40]=6467522952660+1225*x
E[44]=1252975498471200+48800*x
E[48]=152872620852751800-824600*x
E[52]=12069364505468120400+7427600*x
E[56]=630615147670747950200-46927800*x
E[60]=22215915779698502141280+227986400*x
E[64]=535999851662996527356550-892437300*x
E[68]=8973312175360724436541800+2896038600*x
E[72]=105388467829350995361897825-7941316500*x
E[76]=876310274663366548170765600+18652452000*x
E[80]=5197894915757311013178267720-37900941000*x
E[84]=22129281942550350836000132400+67117542000*x
E[88]=67949637583204730713462120200-104150049000*x
E[92]=151037779970268049961942408800+142175052000*x
E[96]=243659108313146247784654076100-171190052250*x
E[100]=285720732951827690430040227204+182092000500*x

Table 23. The form of A(199)

A[31]=4*x
A[32]=21*x
A[35]=3780996219-81*x
A[36]=17224538331-369*x
A[39]=1293504590532+245*x
A[40]=5174018362128+980*x
A[43]=275654609663664+10736*x
A[44]=977320888807536+38064*x
A[47]=36689429004660432-197904*x
A[48]=116183191848091368-626696*x
A[51]=3138034771421711304+1931176*x
A[52]=8931329734046409096+5496424*x
A[55]=176572241347809426056-13139784*x
A[56]=454042906322938524144-33788016*x
A[59]=6664774733909550642384+68395920*x
A[60]=15551141045788951498896+159590480*x
A[63]=171519952532158888754096-285579936*x
A[64]=364479899130837638602454-606857364*x
A[67]=3050926139622646308424212+984653124*x
A[68]=5922386035738078128117588+1911385476*x
A[71]=37939848418566358330283217-2858873940*x
A[72]=67448619410784637031614608-5082442560*x
A[75]=332997904372079288304890928+7087931760*x
A[76]=543312370291287259865874672+11564520240*x
A[79]=2079157966302924405271307088-15160376400*x
A[80]=3118736949454386607906960632-22740564600*x
A[83]=9294298415871147351120055608+28189367640*x
A[84]=12834983526679203484880076792+38928174360*x
A[87]=29897840536610081513923332888-45826021560*x
A[88]=38051797046594649199538787312-58324027440*x
A[91]=69477378786323302982493508048+65400523920*x
A[92]=81560401183944746979448900752+76774528080*x
A[95]=116956371990310198936633956528-82171225080*x
A[96]=126702736322836048848020119572-89018827170*x
A[99]=142860366475913845215020113602+91046000250*x

Table 24. Results of Hash_MIM_Weights_ Enumerators method for EQR(199)

QR(199)	**Weight w**	**Sizeof (Half_Aut)**	**Γ_{w-1}**	**A_{w-1}**	**E_w**
	32	9851	40260	8011740	50073375

Table 25. The Mykkeltveit method results for EQR(199) code

199	**H2**	G_4^0	G_4^1	**S3**	**S5**	**S11**	**S199**
k		25	26	34	20	10	1
E[32]	2675	33	15	165	0	0	0

Table 26. Congruence of the number of codewords of weight 32 in EQR(199)

n	**|G|**	**w**	**Ew**	**E_w mod |G|**	**Check**
199	3940200	32	50073375	2790975	Ok

Tables 25 and 26 give the Mykkeltveit method results for the EQR(199) code and the congruence of the number of codewords of weight 32 for this code. The obtained weights enumerators A(199) and E(199) are then successfully checked.

4.3 For the QR(223) code

The forms of E(223) and A(223) are given in Tables 27 and 28. From Tables 27 and 28, the list R(223) contains two unknown variables R={R32,R36}={X, Y}. By the step (4.1), the value of Γ31 is 3219 and that of A31 is 717837 and the value of Γ35 is 301365 and that of A35 is 67204395. The value of R32 is 5024859 and the value of R36 is 418160680.

Table 27. The form of E(223)

E[32]=7*x
E[36]=56*y
E[40]=246417930220+2884*x-1232*y
E[44]=80690107161664-47936*x+7112*y
E[48]=17550496534206454-82670*x+92064*y
E[52]=2504471683086108736+8145088*x-2263800*y
E[56]=240078191252776863116-97417180*x+24670128*y
E[60]=15763939270240573508096+645783040*x-183186696*y
E[64]=720767184970029831323690-2704246405*x+1038487296*y
E[68]=23271686084583894860833280+6325921280*x-4740925728*y
E[72]=536980260826595701652416880+3049738832*x+17984612800*y
E[76]=8945959934878559088858602240-104783015680*x-57879288544*y
E[80]=108548656056240810065913755400+561702053976*x+160365374848*y
E[84]=966438422361296455257637094144-1999652402944*x-386702936928*y
E[88]=6353270924418855788746208965424+5522930624400*x+818266002624*y
E[92]=31000218573269999868742423985664-12508919711232*x-1528972452000*y
E[96]=112755414265555964374737628832469+23875660564350*x+2535090243840*y
E[100]=306759943483298927727889868797440-39008931601920*x-3743402125680*y
E[104]=625861933005349483774332446615016+55080443818680*x+4936109381280*y
E[108]=959338593364805114215735730445696-67605850032000*x-5822988919440*y
E[112]=1105989123247284833952831214328836+72360343810860*x+6151986456000*y

Table 28. The form of A(223)

A[31]=x
A[32]=6*x
A[35]=9*y
A[36]=47*y
A[39]=44003201825+515*x-220*y
A[40]=202414728395+2369*x-1012*y
A[43]=15849842478184-9416*x+1397*y
A[44]=64840264683480-38520*x+5715*y
A[47]=3760820685901383-17715*x+19728*y
A[48]=13789675848305071-64955*x+72336*y
A[51]=581395212144989528+1890824*x-525525*y
A[52]=1923076470941119208+6254264*x-1738275*y
A[55]=60019547813194215779-24354295*x+6167532*y
A[56]=180058643439582647337-73062885*x+18502596*y
A[59]=4222483733100153618240+172977600*x-49067865*y
A[60]=11541455537140419889856+472805440*x-134118831*y
A[63]=205933481420008523235340-772641830*x+296710656*y
A[64]=514833703550021308088350-1931604575*x+741776640*y
A[67]=7064618989962968082752960+1920368960*x-1439209596*y
A[68]=16207067094620926778080320+4405552320*x-3301716132*y
A[71]=172600798122834332673991140+980273196*x+5780768400*y
A[72]=364379462703761368978425740+2069465636*x+12203844400*y
A[75]=3035236406476653976577025760-35551380320*x-19637615756*y
A[76]=5910723528401905112281576480-69231635360*x-38241672788*y
A[79]=38767377162943146452112055500+200607876420*x+57273348160*y
A[80]=69781278893297663613801699900+361094177556*x+103092026688*y
A[83]=362414408385486170721613910304-749869651104*x-145013601348*y
A[84]=604024013975810284536023183840-1249782751840*x-241689335580*y
A[87]=2495927863164550488436010664988+2169722745300*x+321461643888*y
A[88]=3857343061254305300310198300436+3353207879100*x+496804358736*y
A[91]=12732232628307321374662066994112-5137592024256*x-627970828500*y
A[92]=18267985944962678494080356991552-7371327686976*x-901001623500*y
A[95]=48323748970952556160601840928201+10232425956150*x+1086467247360*y
A[96]=64431665294603408214135787904268+13643234608200*x+1448622996480*y
A[99]=136946403340758449878522262856000-17414701608000*x-1671161663250*y
A[100]=169813540142540477849367605941440-21594229993920*x-2072240462430*y
A[103]=290578754609626546038082921642686+25573063201530*x+2291765069880*y
A[104]=335283178395722937736249524972330+29507380617150*x+2644344311400*y
A[107]=462538250372316751496872584322032-32595677694000*x-2807512514730*y
A[108]=496800342992488362718863146123664-35010172338000*x-3015476404710*y
A[111]=552994561623642416976415607164418+36180

171905430*x+3075993228000*y

Table 29. Results of Hash_MIM_Weights_Enumerators method for EQR(223)

QR(223)	*Weight w*	*Sizeof (Half_Aut)*	*Γ_{w-1}*	*A_{w-1}*	*E_w*
	32	12377	3219	717837	5024859
	36	12377	301365	67204395	418160680

Table 30. The Mykkeltveit method results for EQR(223) code

223	H2	G_4^0	G_4^1	S3	S7	S37	S223
k		29	28	38	16	4	1
E[32]	539	9	9	0	0	0	0
E[36]	7448	50	0	481	0	0	0

Table 31. Congruence of the number of codewords of weights 32 and 36 in EQR(223)

n	|G|	w	Ew	E_w mod |G|	Check
223	5544672	32	5024859	5024859	Ok
223	5544672	36	418160680	2310280	Ok

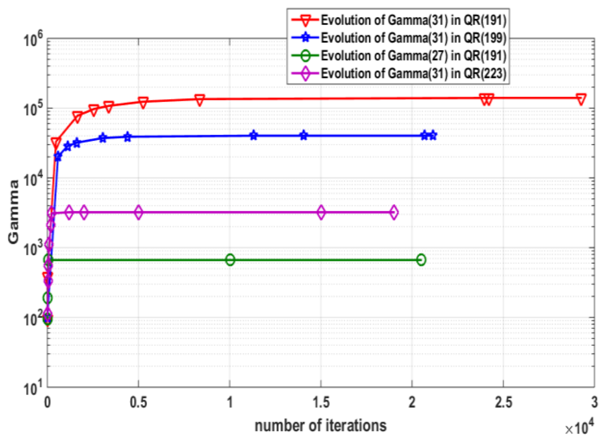

Figure. 2 Study of the convergence of A3 algorithm for QR(191), QR(199) and QR(223) for Γ_{27} and Γ_{31}

Tables 30 and 31 give the Mykkeltveit method results for the EQR(223) code and the congruence of the number of codewords of weights 32 and 36 for this code. The obtained weights enumerators A(223) and E(223) are then successfully checked.

The main advantage of the proposed method is its very low complexity comparing to known methods. All results found here are given by a simple configuration machine instead of the grid that contains about 1500 machines used in [20] for only the small length 137.

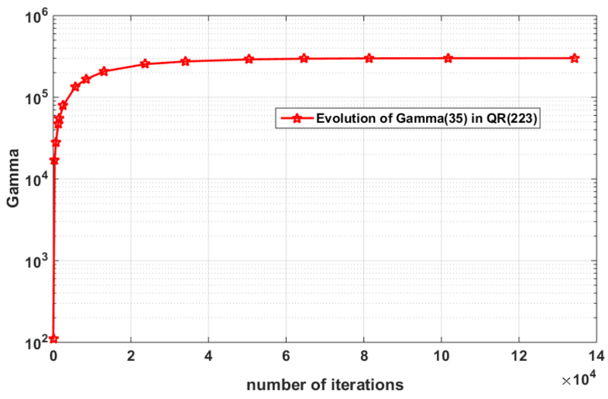

Figure. 3 Study of the convergence of A3 algorithm for QR(223) and Γ_{35}

We note that during the computing of the three weights enumerators of QR(191), QR(199) and QR(223) no codeword of incomplete order is found.

The stop criteria of the algorithm A3 isn't deterministic but when this algorithm seems converges to true value, the proposed check method permits to verify if the convergence is reached or not yet. For example, as it is explained in [7], for the QR(191) there exists an integer η such that Γ_{31}=3040*η+209.

When A3 converges to the value Γ31=140049 it means that η=46. In order to show the convergence of the algorithm A3 for the QR(191), QR(199) and QR(223) codes, we plot in Fig. 2 and 3 the value of Gamma versus number of iterations. These figures show clearly the convergence of A3 for these three codes. We note that the compute of Γ_{35} in QR(223) by A3 has required more number of iterations before its convergence.

5. Advantages and strengths of the proposed method Hash_MIM_Weights_Enumerators

The main succes of the proposed method is that it is succesfully used for enumerating three Quadratic Residue codes that they aren't enumerables despite all developped methods in this field.

The first main advantage is that without hash techniques, the latest step in the old method [7] is very complex and it is very difficult to find the complete list L_w of codewords of a given weight w because it was necessary to verify if a new codeword isn't in L_w before inserting it in L_w when the size of this latest increases. For solving this problem, the authors of [7] have proposed to use a statistical Monte Carlo method to estimate the value of the size A_w of L_w. This statistical way allows finding an estimation of A_w when it is relatively small. When A_w increases, a list S_w of codewords of weight w, which they found by applying the automorphism group on an initial list I_w obtained by a genetic algorithm, is used to estimate the size of L_w as a set, i.e. without repetition of any codeword. So, the old method uses an estimation based on other estimation and the use of the automorphism group generates a set of samples used in the Monte Carlo method that aren't uniformly distributed. These problems have a direct impact on the quality of the estimated values of A_w and they justified the partial differences between the results obtained in [7] and those obtained in this work. For the QR(191) code the values of A27 is 127015 both in [7] and in this work, but the value of A31 is 26749359 here and it is estimated by 19677165 in [7]. For the QR(199) code the value of A31 is 8011740 here but it is estimated by 6755539 in [7].

The second main advantage is that in [7] a genetic algorithm is used to catch codewords of a given weight, but here it is replaced by the powerful faster Multiple Impulse Method.

The third main advantage is that this new proposed method requires finding only the list Γ_{4j-1} of codewords of weight 4j-1 in the QR(n) code without cyclic copies. This improvement allows reducing considerably both the used space memory and the run time.

The spatial and temporal complexity is divided by the length of the code. The temporal complexity is much reduced due to the hash technique used to verify the non existence of a codeword before adding it and also due to the no use of the cyclic copies.

The fourth main advantage is that the congruence of the number of codewords for a given weight firstly proposed by Mykkeltveit is used here to check the obtained result and to verify if the new proposed method completely converges to the exact values, but it is used in the old method to estimate the most likelihood values of the weights enumerators.

6. Conclusion and perspectives

In this paper we have presented a new efficient method for finding the weights enumerators for binary quadratic residue codes having lengths of in the form 8m-1. These codes are very old but they are very important; they were discovered by Prange in 1957. For the lengths 191, 199 and 223 the weights enumerators are remained unknowns in this long period of 60 years because they present a very hard and difficult problem. In consequence of the new results found in this paper, the analytical performances of these three codes are now available. The very important particular property of these three codes is that they are the best known linear block codes until now in terms of the minimum distances that they offer. In the perspectives we have to apply our method on QR Codes with higher lengths and also for other linear block codes like BCH and LDPC codes.

References

[1] G.C. Clark and J.B. Cain, "Error-Correction Coding for Digital Communications", *First edition, Springer, New York*, 1981.

[2] E.R. Berlekamp, "Algebraic Coding Theory", *Second Edition, Aegean Park Press, Laguna Hills, California*, 1984.

[3] F.J. MacWilliams and N.J.A. Sloane, "The theory of Error-Correcting Codes", *North-Holland*, 1977.

[4] A. Vardy, "The intractability of Computing the Minimum distance of a Code", In: *Proc. IEEE Transaction on Information Theory*, Vol. 43, No. 6, pp.1757–1766, 1997.

[5] S. El Kasmi Alaoui, S. Nouh, and A. Marzak, "A Fast Method to Estimate Partial Weights Enumerators by Hash Techniques and Automorphism Group", *International Journal of Advanced Computer Science and Applications*, Vol. 8, No. 9, 2017.

[6] S. Nouh, B. Aylaj, and M. Belkasmi, "A method to determine partial weight enumerator for linear block codes", *Computer Engineering and Intelligent Systems*, Vol. 3, No. 10, 2012.

[7] S. Nouh and M. BelkasmiI, "Genetic algorithms for finding the weight enumerator of binary linear block codes", *International Journal of Applied Research on Information Technology and Computing*, Vol. 2, No. 3, 2011.

[8] X. Wang, J. Gao, and F. Fu, "Complete weight enumerators of two classes of linear codes", In: *Proc. Cryptogr. Commun.*, pp. 599-624, 2017.

[9] V. Lalitha and S.V. Lokam, "Weight enumerators and higher support weights of maximally recoverable Codes", In: *Proc. IEEE Communication, Control and Computing, 53rd Annual Allerton Conference*, 2015.

[10] C. Greene, "Weight enumeration and geometry of linear codes", In: *Proc. Stud. Appl. Math.*, Vol. 55, pp.119–128, 1976.

[11] S. Yang, Z.A. Yao, and C.A. Zhao, "The weight enumerator of the duals of a class of cyclic codes with three zeros", *AAECC*, Vol. 26, pp.347-367, 2015.

[12] H. Yan and C. Liu, "Two classes of cyclic codes and their weight enumerator", *Designs, Codes and Cryptogr.*, Vol. 81, No. 1, 2016.

[13] F.E. Brochero Martínez and C.R. Giraldo Vergara, "Weight enumerator of some irreducible cyclic codes", *Designs, Codes and Cryptogr.*, Vol. 78, No. 703, 2016.

[14] C. Li, Q. Yue, and F. Li, "Hamming Weights of the Duals of Cyclic Codes with Two Zeros", In: *Proc. IEEE Transactions On Information Theory*, Vol. 60, No. 7, July 2014.

[15] S. Li, T. Feng, and G. Ge, "On the weight distribution of cyclic codes with Niho exponents", In: *Proc. IEEE Trans. Inf. Theory*, Vol. 60, No. 7, pp.3903–3912, Jul. 2014.

[16] R.H. Morelos-Zaragoza, "The art of error correcting coding", *John Wiley & Sons Second Edition*, 2006.

[17] J.G. Proakis, "Digital communications", *5th edition*, 2001

[18] M. P. C. Fossorier, S. Lin, and D. Rhee, "Bit-error probability for maximum-likelihood decoding of linear block codes and related soft decision decoding methods", In: *Proc. IEEE Transaction on Information Theory*, Vol. 44, pp.3083-3090, 1998.

[19] P. Gaborit, C. S. Nedeloaia, and A. Wassermann, "On the weight enumerators of duadic and quadratic residue codes", In: *Proc. IEEE Trans. Inf. Theory*, Vol. 51, pp.402–407, 2005.

[20] C. Tjhai, M. Tomlinson, M. Ambroze, and M. Ahmed, "On the weight distribution of the extended quadratic residue code of prime 137", In: *Proc. of the 7th International ITG Conference on Source and Channel Coding*, 2008.

[21] W. Su, C. Lee, T. Lin, T. Truong, and Y. Chang, "On Determination of the Weight Distribution of Binary (168, 84, 24) Extended Quadratic Residue Code", In: *Proc. of ISIT 2008*, 2008.

[22] J. Mykkeltveit, C. Lam, and R.J. McEliece, "On the weight enumerators of quadratic residue codes", *JPL Technical Report* 32-1526, Vol.12, pp.161–166, 1972.

[23] M. Grassl, "Bounds on the minimum distance of linear codes and quantum codes", Online available at http://www.codetables.de, Accessed June 24, 2017.

[24] W.K. Su, P.Y. Shih, T.C. Lin, and T.K. Truong, "On the minimum weights of binary extended quadratic residue codes", In: *Proc. of the 11th International Conference on Advanced Communication Technology*, pp.1912-1913, 2009.

[25] Y. Saouter and G. Le Mestre, "A FPGA implementation of Chen's algorithm", In: *Proc. of the 35th International Symposium on Symbolic and Algebraic Computation*, 2010.

[26] S. Nouh, I. A. Joundan, B. Aylaj, M. Belkasmi, and A. Namir, "New Efficient Scheme Based on Reduction of the Dimension in the Multiple Impulse Method to Find the Minimum Distance of Linear Codes", *International Review on Computers and Software*, Vol. 11, No. 9, pp.742-751, 2016.

[27] I.A. Joundan, S. Nouh, and A. Namir, "Comparative Study Of Two Minimum Distance Computing Methods Based On The Reduction Of Dimension of Linear Codes", In: *Proc. of the 3rd International Congress on Advanced Technologies*, 2017.

[28] A. Joundan, S. Nouh, and A. Namir, "New efficient techniques to catch lowest weights in large Quadratic Residue codes", In: *Proc. of the 5th International Conference on Advances in Computing, Electronics and Communication*, 2017.

[29] M. Askali, A. Azouaoui, S. Nouh, and M. Belkasmi, "On the computing of the minimum distance of linear block codes by heuristic methods", *International Journal of Communications, Network and System Sciences*, Vol. 5, No. 11, 2012.

[30] E.M. Rains and N.J.A. Sloane, "Self-Dual Codes", *Elsevier, North Holland*, 1998.

[31] V. Pless, "Handbook of Coding Theory", *Amsterdam, The Netherlands, North Holland*, 1998.

[32] E. Prange, "Cyclic error-correcting codes in two symbols", *Air Force Cambridge Research Center-TN-57-103*, Cambridge, MA, 1957.