Journal of Engineering and Applied Sciences 11 (11): 2440-2445, 2016
ISSN: 1816-949X
© Medwell Journals, 2016

The Technique for Enhancing Effectiveness of One Modification's of Algorithm Peterson-Gorenstein-Zierler

[1]Mehrdad A. Babavand Arablou, [2]Fikrat G. Feyziyev and [3]Maral R. Mekhtiyeva
[1]Sama Technical and Vocational Training College, Islamic Azad University,
Parsabad Moghan Branch, Parsabad Moghan, Iran
[2]Sumgait State University, Sumgait, Republic of Azerbaijan,
[3]Baku State University, Baku, Republic of Azerbaijan

Abstract: It is proposed a modification of the algorithm Peterson-Gorenstein-Zierler on the based of Gauss method. To enhancing effectiveness this modification, i.e., to accelerate the detection and correction of errors in the non-binary Bose-Chaudhuri-Hocquenghem codes the technique is proposed. In the technique used computations apply tables of operations over an the elements finite field and instead of an elements is used an exponent of power from its representation in form power on the base primitive elements. The detailed description of decoding algorithm of received messages is given.

Key words: Non-binary BCH codes, the algorithm Peterson-Gorenstein-Zierler, primitive element finite field, locators errors, values errors

INTRODUCTION

At present for protection of data in computer systems and networks are wide spread methods of coding theory, cryptography and etc. (Richard, 1983; Ivanov, 2001; William and Vera, 2003). One of the effective error-correcting codes are codes Bose-Chaudhuri Hocquenghem (BCH) (Richard, 1983; Ivanov, 2001; William and Vera, 2003; Birkoff and Barti, 1970; Eeyziyev and Babarand, 2012). For decoding BCH codes, i.e. detecting errors in the received messages, correct them and separating from them information messages the different methods and algorithms are used. The algorithm Peterson-Gorenstein-Zierler (PGZ) is one of these algorithms. PGZ algorithm is based on the solution of a special System of Linear Algebraic Equations (SLAE) relatively unknown error locators using the method of matrix inversion. In this study a modification of the PGZ algorithm where instead the matrix inversion method is applied and Gauss method is proposed. Also the technique of enhancing effectiveness this modification for detecting and correcting errors in the received messages based on the use of the table operations on elements of a finite field is proposed. In these tables instead of the element exponent of its presentation as a power of the primitive element is used.

Statement of the problem: Let's m is natural numbers and α primitive element of fields GF (q^m) (Birkoff and Barti, 1970), i.e., an element of order n = q^m-1 where q is prime number. For t natural number BCH code, correcting t errors is a cyclic code of length n with a generator polynomial g (x) = LCM [f_1 (x), f_2 (x), ..., f_{2t} (x)] where f_β (x) is a minimal polynomial of the element $\alpha^\beta \in$ GF (q), $\beta = \overline{1,2t}$ LCM [f_1 (x), f_2 (x), ..., f_{2t} (x)] is the least common dividend of polynomials f_1 (x), f_2 (x), ..., f_{2t} (x). Let's k = n-deg g(x) and i = (i_0, i_1, ..., i_{k-1}) there k- dimensional arbitrary vector information over the field GF(q). Then the vector information can be encoded by means of operations c(x) = I(x).g(x) coding polynomial $i(x) = i_{k-1}x^{k-1} + ... + i_1x + i_0$ where $c(x) = c_{n-1}x^{n-1} + ... + c_1x + c_0$. For numbers n, k and t must be satisfied the relation $2t \le n - k$ (Birkoff and Barti, 1970). Let's transmitted over the communication channel polynomial c(x) and the other end received polynomial $\upsilon(x) = \upsilon_{n-1}x^{n-1} + ... + \upsilon_1x + \upsilon_0$.

A polynomial error of $e(x) = e_{n-1}x^{n-1} + ... + e_1x + e_0$ i.e., e(x) = v(x)+c(x) and no more then of t coefficients are equal 1 (where record GF(q) indicates, that the polynomials v(x) and c(x) are summing over the field GF(q). Let's considerer at the date moment there were v error where $0 \le v \le t$ and that these errors correspond to the unknown position p_1, p_2,...,p_v. In this case, the error polynomial e(x) can written as $e(x) = e_{p_1}x^{p_1} + e_{p_2}x^{p_2} + ... + e_{p_v}x^{p_v}$. In these relations, the exponents (index) p_1, p_2,...,p_v the coefficients e_{p1},e_{p2},...,e_{pv} and the value of v are unknowns.

Corresponding Author: Mehrdad A. Babavand Arablou, Sama Technical and Vocational Training College, Islamic Azad University, Parsabad Moghan Branch, Parsabad Moghan, Iran

For error detection and correction is necessary to find all of these unknowns. To find the value v and $p_1, p_2,...,p_v$ in (Richard, 1983) proposed to use the components of the syndrome s_β, $\beta=\overline{1,2t}$ where $s_\beta = \upsilon(\alpha^\beta)$. Since c(x) = i(x).g(x) and α^β, $\beta=\overline{1,2t}$ are roots of generator polynomial g(x) then $c(\alpha^\beta)=0$, $\beta=\overline{1,2t}$ Given these facts in determining Eq. 1, we get:

$$S_\beta = \upsilon(\alpha^\beta) = c(\alpha^\beta) + e(\alpha^\beta) = e_{p_1}(\alpha^{p_1})^\beta + \quad (1)$$

$e_{p_2}(\alpha^{p_2})^\beta + ... + e_{p_v}(\alpha^{p_v})^\beta$, $GF(q^m)$

Equation 1 shows that if $S_\beta = 0$, $\beta=\overline{1,2t}$ then in the received message is not error and otherwise that is error. Let, $Y_l = e_{pl}$ (values errors) and $X_l = \alpha^{pl}$ (locators errors), l = 1,....., v. Since the order α of the element is equal n then all locators of the presented configuration errors are different. For each $\beta \in \{1,...,2t\}$ from the Eq. 1 have: $S_\beta = \upsilon(\alpha^\beta) = Y_1X_1^\beta + Y_2X_2^\beta + ... + Y_vX_v^\beta$. Thus, we get the following system of equations for the unknown locator errors $X_1,...,X_v$ and value errors $Y_1,...,Y_v$:

$$S_\beta = Y_1X_1^\beta + Y_2X_2^\beta + ... + Y_vX_v^\beta \quad \beta=\overline{1,2t} \quad (2)$$

System of nonlinear Eq. 2 decide to indirectly (Richard, 1983). For this purpose, the error locator polynomial $\Lambda(x)=\Lambda_v x^v + ... + \Lambda_1 x + 1$ whose roots are X_l^{-1}, $l=1,...,\upsilon$. If the coefficients of the polynomial $\Lambda(x)$ are known calculating the error locator must find its roots. In (Richard, 1983) it was get SLAE, relating the components of the syndrome with the coefficients of the polynomial $\Lambda(x)$. This SLAE has the following matrix form:

$$A \times col(\Lambda_v, \Lambda_{v-1},..., \Lambda_1) = col(-S_{v+1}, -S_{v+2},..., -S_{2v}) \quad (3)$$

Where:

$$A = (a_{\rho,\beta}),\ \rho=\overline{1,v},\ \beta=\overline{1,v}$$

Where:

$$a_{\rho,\beta} = S_{\rho-1+\beta}.$$

After determination of the coefficients $\Lambda_1, \Lambda_2,..., \Lambda_v$ of the polynomial of $\Lambda(x)$, for determine of its roots for each element $x \in GF(q^m)$ needs to calculate $\Lambda(x)$ and identify those values x in which $\Lambda(x) = 0$. By means of the Horner scheme $\Lambda(x)$ is calculated recurrently as follows:

$$\Lambda_0 := 1,\ \Lambda(x) := \Lambda_v x + \Lambda_{v-1}\Lambda(x) := \Lambda(x) \times x + \Lambda_l \quad (4)$$
$$l = v-2, v-3,...,0$$

The known (Richard, 1983), that if the matrix A is non-singular, then the system has a unique solution with of $\Lambda_1, \Lambda_2, ..., \Lambda_v$. It is already proved (Richard, 1983) that: If $M = (S_{\rho-1+\beta})$, $\rho=\overline{1,\mu}$, $\beta=\overline{1,\mu}$ and if μ is equal to number v when the same number of errors that have occurred, then the matrix is non singular and if μ more v, then the matrix is singular. On the base of these facts in (Richard, 1983) a construction decoding algorithm where the SLAE Eq. 3 is solved by inversion of matrix A.

MATERIALS AND METHODS

Modification of the Peterson-Gorenstein-Zierler algorithm on the base Gauss method: Primitive polynomial of degree m over a field GF (q) for constructing a field $GF(q^m)$ designated by P(x). Note that in the field $GF(q^m)$ primitive element α corresponds to a polynomial x (Richard, 1983). So instead P(x) you can use $p(\alpha)$. Calculations S_β by the Eq. 1 are carried out over the field $GF(q^m)$. This means that after the operations indicated on the right side of the equation, the result is divided by a polynomial $P(\alpha)$ and the remainder polynomial is taken. By definition it is true:

$$S_\beta = \upsilon_0 + \upsilon_1 a^\beta + \upsilon_2 a^{2\beta} + ... + \upsilon_{n-2} a^{(n-2)\beta} + \upsilon_{n-1} a^{(n-1)\beta}$$

Then by Horner scheme can get:

$$S_\beta = (...((\upsilon_{n-1}a^\beta + \upsilon_{n-2})a^\beta + \upsilon_{n-3})a^\beta + ... + \upsilon_1)a^\beta + \upsilon_0$$

Therefore, for finding S_β, $\beta = 1,...,2t$, can use the following algorithm (Algorithm A):

Step 0: $S_\beta := R_{P(\alpha)}[\upsilon_{n-1}a^\beta + \upsilon_{n-2}]$, $\gamma = 1$
Step 1: $S_\beta := R_{P(\alpha)}[S_\beta \alpha^\beta + \upsilon_{n-2-\gamma}]$
Step 2: $\gamma := \gamma+1$ if $n-2-\gamma \geq 0$ then go to step 1, else to step 3
Step 3: Stop

In this algorithm, the operator $R_{P(\alpha)}[j(\alpha)]$ is used to find the polynomial remainder division of the polynomial $\varphi(\alpha)$ by a polynomial $P(\alpha)$. If $X_1,..., X_v$ and the number of v are known, then to find the $Y_1,..., Y_v$ can use the following systems of nonlinear algebraic equations which consists of a first, second, ..., v-th equations of system Eq. 2 i.e.,:

$$S_\beta = Y_1X_1^\beta + Y_2X_2^\beta + ... + Y_vX_v^\beta \quad \beta=\overline{1,v} \quad (5)$$

Theorem 1: If $X_1,..., X_v$ and the value of v are known, then for solving Eq. 5 with respect to $Y_1,..., Y_v$ can be use following recurrence relations:

$$Y_v = \left(B_v^{(v-1)}X_?\right)^{-1} S_v^{(v-1)}$$

$$Y_l = \left(B_l^{(l-1)}X_l\right)^{-1}\left[S_l^{(l-1)} - \sum_{s=l+1}^{?} B_\sigma^{(l-1)}X_\sigma Y_\sigma\right]$$

$$l = v-1, v-2, \ldots, 1$$

Where:

$$B_i^{(0)} = 1,\ S_i^{(0)} = S_i\ i = 1,\ldots,v$$

$$B_i^{(l)} = B_i^{(l-1)}(X_i - X_l),\ S_i^{(l)} = S_i^{(l-1)} - X_l S_{i-1}^{(l-1)}$$
$$i = l+1,\ldots,v,\ l = 1,\ldots,v-1$$

The SLAE Eq. 3 can be solving by Gauss methods. Modification of the Peterson-Gorenstein-Zierler algorithm on the base Gauss method can describe as follows:

Step 0: It is based on the received polynomial v(x) calculated $S_\beta = v(\alpha^\beta)$, $\beta = \overline{1,2t}$ by algorithm A. If $S_\beta = 0$, $\beta = \overline{1,2t}$ then go to step 14, else accept v = t and go to step 1

Step 1: Let's construct a matrix $A = (a_{\rho,\beta}), \rho = \overline{1,v},\ \beta = \overline{1,v}$ and a vektorb = col(b_1, b_2,....b_v) where $a_{\rho,\beta} = S_{\rho-1+\beta}$, $b\rho = -S_{v+\rho}$ If in matrix A have zero rows and columns then go to step 8, else accept l =1 and go to step 2

Step 2: If l+1>v then go to step 9, else go to step 3

Step 3: Let's smallest element of sets $Q = \{\xi \mid \xi \in \{1,\ldots, v\},\ a_{\xi 1} \neq 0\}$ is σ. If σ = l, then go to step 4, else interchange the l-th and σ-th rows of the matrix A and the l-th and σ-th component of the vector b, i.e., we accept $c = a_{l\beta}, a_{l\beta} = a_{\sigma\beta}, a_{\sigma\beta} = c, \beta = 1,\ldots, v$, $c = b_l$, $b_l = b_\sigma$, $b_\sigma = c$ and go to step 4

Step 4: The l-th row of the matrix A sequentially multiply by $-a_{l+1,\,l}/a_{ll}$, $-a_{l+2,\,l}/a_{ll}$, ..., $-a_{vl}/a_{ll}$, respectively and add to the l+1th, l-2th, ..., v-th row

$$a_{j\beta} := a_{j\beta} - (a_{jl}/a_{ll})a_{l\beta},\ GF(q),\ \beta = 1,\ldots,v,\ j = l+1,\ldots,v \quad (6)$$

Then, the l-th component of the vector b sequentially multiply by $-a_{l+1,\,l}/a_{ll}$, $-a_{l+2,\,l}/a_{ll}$, ..., $-a_{vl}/a_{ll}$ respectively and add to the l+1-th, l+2-th, ..., v-th component of the vector b:

$$b_j := b_j - (a_{jl}/a_{ll})b_l\ \ GF(q)\ j = l+1,\ldots, v \quad (7)$$

Go to step 5.

Step 5: l:= l+1. If l<v, then go to step 6, else go to step 7

Step 6: If in sub matrix $A_l = (a_{\rho\beta}), ?, \beta = \overline{l,v}$ have a zero row or column, then go to step 8, else go to step 3

Step 7: If $a_{ll} \neq$ then go to step 9, else go to step 8

Step 8: v: = v-1 Go to step 1

Step 9. Find the solutions of Eq. 3 on the recurrence formulas:

$$\Lambda_1 = \left(a_{vv}\right)^{-1} \times b_v\ \ \Lambda_\rho = \left(a_{v,?+1,v,?+1}\right)^{-1}$$
$$\left\{b_{v-\rho+1} - \sum_{s=1}^{\rho-1} a_{v-\rho+1,v-\rho+1+\sigma}\Lambda_{\rho-\sigma}\right\}, \quad (8)$$
$$GF(q)\ \ \rho = 2,3,\ldots, v$$

Step 10: Find the roots x_1, ..., x_v of the error locator Λ(x) and error locators on the equation $X_\beta = x_\beta^{-1}\ \beta = 1,\ldots, v$

Step 11: Define $B_i^{l}, S_i^{(l)}$ i = l+1,...,v, l = 0,1,...,v-1, by the recurrence formulas

$$B_i^{(0)} = 1,\ S_i^{(0)} = S_i\ i = 1,\ldots, v \quad (9)$$

$$B_i^{(l)} = B_i^{(l-1)}(X_i - X_l),\ S_i^{(l)} = S_i^{(l-1)} - X_l S_{i-1}^{(l-1)},\ GF(q) \quad (10)$$
$$i = l+1,\ldots,v,\ l = 1,\ldots,v-1$$

Step 12: Define Y_1,...., Y_v by the recurrence formulas:

$$Y_? = \left(B_v^{(v-1)}X_?\right)^{-1} S_v^{(v-1)}\ \ Y_1 = \left(B_1^{(l-1)}X_1\right)^{-1}$$
$$\left[S_l^{(l-1)} - \sum_{s=l+1}^{v} B_s^{(l-1)}X_s Y_s\right] \quad (11)$$
$$GF(q)\ l = v-1, v-2, \ldots, 1$$

Step 13: Find the values of index p_1,..., p_v and correct errors on the formula $v_{p_\ell} := v_{p_\ell} - Y_\ell, \ell = 1,\ldots, v\ \ GF(q)$.

Step 14: Define information polynomial by the Equation:
I(x) = v(x)/g(x).

Step 15: Stop

Elements of the matrix A in Eq. 3are elements of the field $GF(q^m)$, i.e. they are polynomials over the field GF(q) The non-zero elements of the field $GF(q^m)$ are powers of a primitive element. To execute the operations of addition and multiplication of elements of the $GF(q^m)$ corresponding tables can be used. Using harvested tables will reduce the execution time of those operations.

RESULTS AND DISCUSSION

Technique to accelerate the detection and correction of errors: S_β, $\beta = 1,\ldots, 2t$, receive the values in the finite field $GF(q^m)$. Therefore, they are 0 (zero element) or the power of the primitive element. The number is entered: N_β, $\beta = 1,\ldots, 2t$:

$$N_\beta = \begin{cases} -1, & \text{if } S_\beta = 0 \\ k, & \text{if } S_\beta = \alpha^k \text{ where } k \in \{0,\ldots, q^m - 2\} \end{cases} \quad (12)$$

Let's introduce massive of M1and M2. The element M(u, β, v) of massive of M1 where u∈GF(q) v∈GF(q) and β∈{1,.., q^m-2} is used to find exponent of power of the number $u+\alpha^{\beta v}$and is determined by the Eq:

$$M1(u,\beta,v) = \begin{cases} -1, & \text{if } u + \alpha^\beta v = 0 \\ k, & \text{if } u + \alpha^\beta v = \alpha^k \text{ where } k \in \{0,\ldots, q^m - 2\} \end{cases}$$

The element M2(τ, v) of massive M2 where τ∈{-1, 0, ..., q^m-2} and v∈ GF(q)is used to find exponent of power of the number α^τ and is determined by Eq.:

$$M2(\tau,\nu)=\begin{cases}\tau, & \text{if } \nu=0\\ -1, & \text{if } \nu=0 \text{ and } \tau=-1\\ \sigma, & \text{if } \nu\neq 0 \text{ and } \alpha^{\tau}+\nu=\alpha^{\sigma},\\ & \text{where } \sigma\in\{0,\ldots,q^m-2\}\end{cases}$$

When $x,y\in\{-1, 0, \ldots,q^m-2\}$ to find the exponent in the representation of an expression $\alpha^x\,\alpha^y$ as a power of the primitive element α of the field of GF(q^m), let's introduce an operation * what is defined as follows:

$$x*y=\begin{cases}-1, & \text{if } x=-1 \text{ or (and) } y=-1\\ x+y-(q^m-1), & \text{if } x^1\text{-}1,y^1\text{-}1,x+y^3q^m-1\\ x+y, & \text{if } x^1\text{-}1,y^1\text{-}1,x+y<q^m-1\end{cases}$$

If the construction massive of M1 and M2, then similarly to the above algorithm can be computed $N_\beta,\beta=\overline{1,2t}$ as follows:

Step 0: $N_\beta=M1(\upsilon_{n-1},\beta,\upsilon_{n-2}),\ \gamma=1$
Step 1: $N_\beta:=M2((N_\beta*\beta),\upsilon_{n-2-\gamma})$
Step 2: $\gamma:=\gamma+1$ if $n-2-\gamma\geq 0$ then go to step 1, else to step 3
Step 3: Stop

If the numbers $N_\beta,\beta=\overline{1,2t}$ are calculated by means of the algorithm, then

$$S_\beta=\begin{cases}0, & \text{if } N_\beta=-1\\ \alpha^k, & \text{if } N_\beta=k, \text{ where } k\in\{0,\ldots,q^m-2\}\end{cases}$$

Therefore, in the future instead of $N_\beta,\beta=1,\ldots,2t$ can be used $N_\beta,\beta=1,\ldots,2t$. In the Eq. 6-11 operations are carried out on polynomials. Let's consider the transformation of these formulas to formulas which are used instead of the polynomial corresponding degrees of power primitive element. For that on the base of the matrix A we introduce the matrix of: $Z=(z_{\rho\beta})\,\rho=\overline{1,\nu}\,\beta=\overline{1,\nu}$ and on the base of the vector of b introduce v dimensional vector $\eta=col(\eta_1,\ldots,\eta_\nu)$. Where:

$$z_{\rho\beta}=\begin{cases}-1, & \text{if } a_{\rho\beta}=0\\ \sigma, & \text{if } a_{\rho\beta}=a^{\sigma}\\ & \text{where } \sigma\in\{0,\ldots,q^m\text{-}2\}\end{cases}\qquad(13)$$

$$\eta_\rho=\begin{cases}-1, & \text{if } b_\rho=0\\ \sigma, & \text{if } b_\rho=\alpha^{\sigma}\\ & \text{where } \sigma\in\{0,\ldots,q^m-2\}\end{cases}\qquad(14)$$

Using the Eq. 12 can be determined $z_{\rho\beta},\rho=\overline{1,\nu},\beta=\overline{1,\nu}$ and $\eta_\rho\ \rho=\overline{1,\nu}$ formulas $z_{\rho\beta}=N_{\rho-1+\beta}$ and $\eta_\rho=MP(N_{\rho+\nu})$ respectively. MP ($N_{\rho+\nu}$) is the value of a exponent of expression which contrary is an expression MP(x) and can be defined by means of the Eq:

$$MP(x)=\begin{cases}1, & \text{if } x=-1\\ \sigma, & \text{if } x\neq -1 \text{ and } \alpha^{\sigma}+\alpha^{x}=0\\ & \text{where } \sigma\in\{0,\ldots,q^m\text{-}2\}\end{cases}$$

Using primitive element α can write Eq. 6 and 7 on the base 13 and 14 in the form:

$$\alpha^{z_{j\beta}}:=\alpha^{z_{j\beta}}-(\alpha^{z_{ll}})^{-1}\alpha^{z_{jl}}\alpha^{z_{l\beta}},\quad \alpha^{\eta_j}:=\alpha^{\eta_j}-(\alpha^{z_{ll}})^{-1}\alpha^{z_{jl}}\alpha^{\eta_l}$$

Since, $(\alpha^{z_{ll}})^{-1}=\alpha^{2^m-1-z_{ll}}$, therefore:

$$\alpha^{z_{j\beta}}:=\alpha^{z_{j\beta}}-\alpha^{2^m-1-z_{ll}}\alpha^{z_{jl}}\alpha^{z_{l\beta}},$$
$$\alpha^{\eta_j}:=\alpha^{\eta_j}-\alpha^{2^m-1-z_{ll}}\alpha^{z_{jl}}\alpha^{\eta_l}$$

From these relations it turns out that:

$$z_{j\beta}:=MF(z_{j\beta},\ (2^m\text{-}1\text{-}z_{ll})*z_{j1}*z_{1\beta})\ \beta=1,\ldots,v,\ j=l+1,\ldots,v$$

$$\eta_j:=MF(\eta_j,\ (2^m\text{-}1\text{-}z_{ll})*z_{j1}*\eta_1)\ j=l+1,\ldots,v$$

Where MF(x, y) is the value of a exponent of difference α^x-α^y which can be defined by means of Equation:

$$MF(x,y)=\begin{cases}x, & \text{if } y=-1,x\geq 0\\ MP(y), & \text{if } x=-1,y\geq 0\\ -1, & \text{if } y=-1,x=-1\\ \sigma, & \text{if } y\neq -1,x\neq -1,\ \alpha^x-\alpha^y=\alpha^{\sigma}\\ & \text{where } \sigma\in\{0,\ldots,q^m-2\}\\ -1, & \text{if } y\neq -1,x\neq -1,\ \alpha^x-\alpha^y=0\end{cases}$$

For any $\rho\in\{1,2,\ldots,\}$ introduce designation:

$$\lambda_\rho=\begin{cases}-1, & \text{if } \Lambda_\rho=0\\ \sigma, & \text{if } \Lambda_\rho=\alpha^{\sigma} \text{where } \sigma\in\{0,\ldots,q^m-2\}\end{cases}$$

From Eq. 8 we'll accept of $\lambda_1=(q^m-1-z_{\nu\nu})*\eta_\nu$ where in the right side the multiplier $q^m-1-z_{\nu\nu}$ is used for indicating, if $a_{\nu\nu}=\alpha^{z_{\nu\nu}}$ then $(a_{\nu\nu})^{-1}=\alpha^{2^m-1-z_{\nu\nu}}$. In the right side of Eq. 8 the expression:

$$J_\rho=\sum_{\sigma=1}^{\rho-1}a_{\nu-\rho+1,\nu-\rho+1-\sigma}\Lambda_{\rho-\sigma}$$

can be calculated recurrently as follows:

$$J_\rho:=0;\ J_\rho:=J_\rho+a_{\nu-\rho+1,\nu-\rho+1-\sigma}\Lambda_{\rho-\sigma}\ \sigma=1,\ldots,\rho-1$$

Therefore, if $J_\rho = \alpha^{\gamma\rho}$, the exponent γ_ρ can also be defined recurrently as follows:

$$\gamma_\rho := -1; \gamma_\rho := MC(\gamma_\rho, z_{v-\rho+1,v-\rho+1+\sigma} * \lambda_{\rho-\sigma}), \sigma = 1,....., \rho - 1$$

where M(x, y) is the value of a exponent of sum $\alpha^x+\alpha^y$ which can be defined by means of Eq:

$$MC(x,y) = \begin{cases} y, & \text{if } x = -1 \\ x, & \text{if } y = -1 \\ -1, & \text{if } \alpha^x + \alpha^y = 0 \\ \tau, & \text{if } \alpha^x + \alpha^y = \alpha^\tau \quad \text{where } \tau \in \{0,...,q^m - 2\} \end{cases}$$

Thus, by means of Eq. 8:

$$\lambda_\rho = (2^m - 1 - z_{v-\rho+1,v-\rho+1}) * MC(\eta_{v-\rho+1}, \gamma_\rho)$$

For speed up the computation, instead x can use it as a description $x = \alpha^\beta$. Then the scheme Eq. 4 can be written as:

$$\lambda_0 := 0,\ \lambda(\beta) := MC((\lambda_v * \beta), \lambda_{v-1})$$

$$\lambda(\beta) := MC((\lambda(\beta) * \beta), \lambda_1)\ l = v-2, v-3, ..., 0$$

where $\lambda(\beta)$ defined by means of the following Eq:

$$\lambda(\beta) = \begin{cases} -1, & \text{if } \Lambda(\alpha^\beta) = 0 \\ \sigma, & \text{if } \Lambda(\alpha^\beta) = a^s \text{ where } \sigma \in \{0,...,q^m - 2\} \end{cases}$$

By definition, for each l = 1,..., v:

$$p_l = \begin{cases} -1, & \text{if } X_l = 0 \\ \sigma, & \text{if } X_l = \alpha^\sigma \text{where } s \in \{0,...,q^m - 2\} \end{cases}$$

Let's introduce the following:

$$\theta_i^{(l)} = \begin{cases} -1, & \text{if } B_i^{(l)} = 0, \\ k, & \text{if } B_i^{(l)} = \alpha^k \text{where } k \in \{0,...,q^m - 2\} \end{cases}$$

$$s_i^{(l)} = \begin{cases} -1, & \text{if } S_i^{(l)} = 0 \\ \tau, & \text{if } S_i^{(l)} = a^\tau \text{ where } t \in \{0,...,q^m - 2\} \end{cases}$$

$$y_l = \begin{cases} -1, & \text{if } Y_l = 0 \\ \xi, & \text{if } Y_l = a^\xi \text{ where } \xi \in \{0,...,q^m - 2\} \end{cases}$$

The $\theta_i^{(l)}$ can be calculated based on the Eq. 9 and10 in the following recursion Eq:

$$\theta_i^{(0)} = 0,\ i = 1,2,...,v$$

$$\theta_i^{(l)} = \theta_i^{(l-1)} * MF(d_i, d_l),\ i = l+1,..., v;\ l = 1,..., v-1$$

The $s_i^{(l)}$ can be calculated based on the Eq 9 and 10 in the following recursion Eq:

$$s_i^{(0)} = N_i,\ i = 1, 2,..., v$$

$$s_i^{(l)} = MF(s_i^{(l-1)}, d_1 * s_{i-1}^{(l-1)}),\ i = l+1,..., ?;\ l = 1,..., v-1$$

By Eq. 11 we obtain:

$$y_v = (q^m - 1 - (\theta_v^{(v-1)} * d_v)) * s_v^{(v-1)}$$

In the right side of the Eq. 11 the expression:

$$J_l = \sum_{\sigma=l+1}^{v} B_\sigma^{(l-1)} X_\sigma Y_\sigma$$

can be calculated recurrently as follows:

$$J_1 := 0;\ J_l := J_l + B_\sigma^{(l-1)} X_\sigma Y_\sigma\ \sigma = l+1,...,v$$

Therefore if $J_1 = a^\eta$ then the exponent γ_1 can also be defined recurrently as follows:

$$\gamma_1 := 0;$$

$$\gamma_1 := MC(\gamma_1, \theta_s^{(l-1)} * d_s * y_s)\ s = l+1,...,v$$

Thus:

$$y_l = (q^m - 1 - q_l^{(l-1)} * d_l) * MF(s_1^{(l-1)}, g_l),\ l = n-2, n-3, ..., 1$$

Description of the algorithm to detect and correct errors in the received polynomial: Let's assume that the massive (tables) M1, M2, MP, MF, MC are pre-compiled. Then the decoding algorithm can be described in detail in the follows view:

Step 0: Take $\upsilon_{n-1}, \upsilon_{n-2}, ..., \upsilon_1, \upsilon_0$. Accept $\beta = 1$
Step 1: $N_\beta = M1(\upsilon_{n-1}, \beta, \upsilon_{n-2})$ $\gamma = 1$
Step 2: $N_\beta := M2((N_\beta * \beta), \upsilon_{n-2-\gamma})$
Step 3: $\gamma := \gamma+1$ If $n-2-\gamma \geq 0$ then go to step 2, else to step 4
Step 4: $\beta := \beta+1$. If $\beta \leq$ then go to step 1, else to step 5
Step 5: If numbers $N_1, N_2, ..., N_{2t}$ are equal -1, then go to step 53, else accept v = t and go to step 6
Step 6: Construct a matrix $Z = (z_{\rho\beta})\ \rho = \overline{1,v} \beta = \overline{1,v}$ where $z_{\rho\beta} = N_{\gamma-1+\beta}\ \beta = \overline{1,v}$, Construct a vector $\eta = (\eta_1, \eta_2, ..., \eta_v)$ where $\eta_\rho = MP(N_{\rho+v})\ \rho = \overline{1,v}$. If in the matrix A have zero rows and columns then go to step 20, else accept l = 1 and go to step 7
Step 7: If l+1> then go to step 9, else go to step 8
Step 8: Find the $\sigma = \min\{\xi | \xi \in \{\ell,...,1\}, z_{\xi\ell} \neq -1\}$. If $\sigma \neq$ l then go to step 9, else to step 12

Step 9: $\beta = 1$
Step 10: Sequentially accept: $c = Z_{1\beta}$, $Z_{1\beta} = Z_{\sigma\beta}$, $Z_{\sigma\beta} = c$
Step 11: β: = β+1. If $\beta \le v$ then go to step 10, else sequentially accept $c = \eta_1$, $\eta_1 = \eta_\sigma$, $\eta_\sigma = c$ and go to step 12
Step 12: J: = l+1
Step 13: $\beta = 1$
Step 14: $z_{j\beta} := MF(z_{j\beta}, (q^m - 1 - z_{jl}) * z_{jl} * z_{l\beta})$
Step 15: $\beta := \beta+1$. If $\beta \le v$, then go to step 14, else accept $\eta_j := MF(\eta_j, (q^m - 1 - z_{jj}) * z_{j\ell} * \eta_\ell)$ and go to step 16
Step 16: j := j+1. If $j \le v$, then go to step 13, else go to step 17
Step 17: l := l+1. If $l \le v$ then go to step 18, else to step 19
Step 18: If in the matrix $Z=(z_{\rho\beta})$ $\rho=\overline{l,v}$ $\beta=\overline{l,v}$ have zero rows and columns, then go to step 20, else go to step 8
Step 19: If $a_{ll} \neq 0$ then go to step 21, else go to step 20
Step 20: V := v-1. Go to step 6
Step 21: $\lambda_1 = (q^m - 1 - z_{vv}) * \eta_v$
Step 22: $\rho = 2$ If $\rho > v$, then go to step 28, else go to step 23.
Step 23: $\gamma := -1$; σ 1
Step 24: $\gamma := MC(\gamma, z_{v-\rho+1, v-\rho+1+\sigma} * \lambda_{\rho-\sigma})$
Step 25: $\sigma := \sigma+1$. If $\sigma \le \rho-1$ then go to step 24, else to step 26
Step 26: $\lambda_\rho := (q^m - 1 - z_{v-\rho+1, v-\rho+1}) * MF(\eta_{v-\rho+1}, \gamma)$
Step 27: $\rho := \rho+1$. If $\rho \le v$ then go to step 23, else to step 28
Step 28: $\beta = -1$, $\lambda_0 = 0$, $\sigma = 0$
Step 29: Accept $\lambda(\beta) := MC((\lambda_v * \beta), \lambda_{v-1})$ and l = v-2. If l<0, then go to step 32, else step 30
Step 30: $\lambda(\beta) := MC((\lambda(\beta) * \beta), \lambda_\ell)$
Step 31: L ;+ l-1. If $l \le 0$, then go to step 30, else to step 32.
Step 32: If $\lambda(\beta) \neq -1$ then go to step 34, else to step 33.
Step 33: $\sigma := \sigma+1$, $x_\sigma = \beta$. If $\beta \ge v$, then go to step 35, else to step 34
Step 34: β := β+1. If $\beta \le q^m$ then go to step 29, else to step 35
Step 35: For any l = 1,..., v define p_l by Eq $p_l = q^m - 1 - x_l$
Step 36: I = 1
Step 37: $\theta_i^{(0)} = 0$, $s_i^{(0)} = N_i$.
Step 38: I = I+1. Ifi $\le v$ then go to step 37, else to step 39
Step 39: L = 1
Step 40: I = l+1
Step 41:. If $i \le v$ then go to step 42, else to step 43
Step 42: Accept: $\theta_i^{(\ell)} = \theta_i^{(\ell-1)} * MF(p_i, p_\ell)$, $s_i^{(\ell)} = MF(s_i^{(\ell-1)}, p_\ell * s_{i-1}^{(\ell-1)})$ After accept I= i+1 and go to step 41
Step 43: l := l+1 If $l \le v-1$, then go to step 40, else to step 44
Step 44: Accept: $y_v = (q^m - 1 - (\theta_v^{(v-1)} * p_v)) * s_v^{(v-1)}$ and l = v Go to step 45
Step 45: l := l-1 If l = 0, then go to step 50, else to step 46
Step 46: $\gamma_{1:=-1}$ $\sigma = l$ Go to step 47
Step 47: If $\sigma \le v$, then go to step 48, else to step 49
Step 48: Accept $\gamma_\ell := MC(\gamma_\ell, (\theta_\sigma^{(\ell-1)} * p_\sigma * y_\sigma))$ $\sigma := \sigma+1$ and go to step 47
Step 49: Accept $y_\ell := (q^m - 1 - (\theta_\ell^{(\ell-1)} * p_\ell)) * MF(s_\ell^{(\ell-1)}, \gamma_\ell)$ and go to step 45
Step 50: l = 1
Step 51: Accept: $\upsilon_{p_\beta} := \upsilon_{p_\beta} - \alpha^{\gamma_\ell}$, GF(q) ℓ =1,..., v
Step 52: l = l+1. If $l \le v$, then go to step 51, else to step 53.
Step 53: Divided polynomial v(x) by a polynomial $g(x) = g_{n-k}x^{n-k} + ... + g_1x + g_0$ on scheme (Feyziyev and Babavand, 2012):

$$\begin{cases} y_\alpha[0] = \upsilon_\alpha, \ \alpha = 0,1,...,n-1 \\ y_{n-\beta-\alpha}[\beta] = y_{n-\beta-\alpha}[\beta-1] - y_{n-\beta}[\beta-1]g_{n-k-\alpha}, \ \alpha = 1,...,n-k, \ GF(q) \\ y_{n-\beta-\alpha}[\beta] = y_{n-\beta-\alpha}[\beta-1], \ \alpha = n-k+1,...,n-\beta \\ I_{k-\beta}[\beta] = y_{n-\beta}[\beta-1], \ \beta = 1,2,...,k-1 \\ y_{n-k-\alpha}[k] = y_{n-k-\alpha}[k-1] - y_{n-k}[k-1]g_{n-k-\alpha}, \ \alpha = 1,...,n-k, \ GF(q) \\ I_o[k] = y_{n-k}[k-1] \end{cases}$$

Step 54: Certain components of the vector information from the formula $i_{k-\beta} = I_{k-\beta}[\beta]$, $\beta = 1,2,...,k$
Step 55: Stop

CONCLUSION

Thus, to speed up the execution of the PGZ algorithm for non-binary BCH codes the use of the Gauss method and to perform operations of summing, multiplication, etc., elements of finite field Gf(q^m) using special tables is proposed. On the bas of the above stated a detailed algorithm for detecting and correcting errors in the received polynomial is developed. This algorithm can be implemented in software of the assembler.

REFERENCES

Birkoff, G., T. Barti, 1970. Modern Applied Algebra. McGraw-Hill, New York, USA., Pages: 400.

Feyziyev, F.G. and A.M. Babavand, 2012. Description of decoding of cyclic codes in the class of sequential machines based on the Meggitt theorem. Autom. Control Comput. Sci., 46: 164-169.

Ivanov, M.A., 2001. Crypto Graphic Methods of Information Protection in Computer Systems and Networks. Kudits-Obraz Publisher, Moscow, Russia, Pages: 368.

Richard, E.B., 1983. Theory and Practice of Error Control Codes. Wesley Publishing Company, Addison Texas, ISBN:9780201101027, Pages: 500.

William, C.H., P. Vera, 2003. Fundamentals of Error-Correcting Codes. Cambridge University Press, Cambridge, England, Pages: 662.