INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND ANALYSIS

ISSN(print): 2643-9840, ISSN(online): 2643-9875

Volume 05 Issue 06 June 2022

DOI: 10.47191/ijmra/v5-i6-31, Impact Factor: 6.261

Page No. 1437-1455

"Catching Errors in Cyclic Codes"
"Detection of Errors in Cyclic Codes"
« Error - Trapping in Cyclic Codes »

Tran Van Anh[1], Nguyen Van Hoang[2], Huynh Chi Nhan[3], Tran Thi Yen Nhi[4], Châu Quang Vũ[5]

[1,2,3,4,5] Faculty of Fundamental Science, Van Lang University, 69/68 Dang Thuy Tram Street, Ward 13, Binh Thanh District, Ho Chi Minh City, Vietnam.

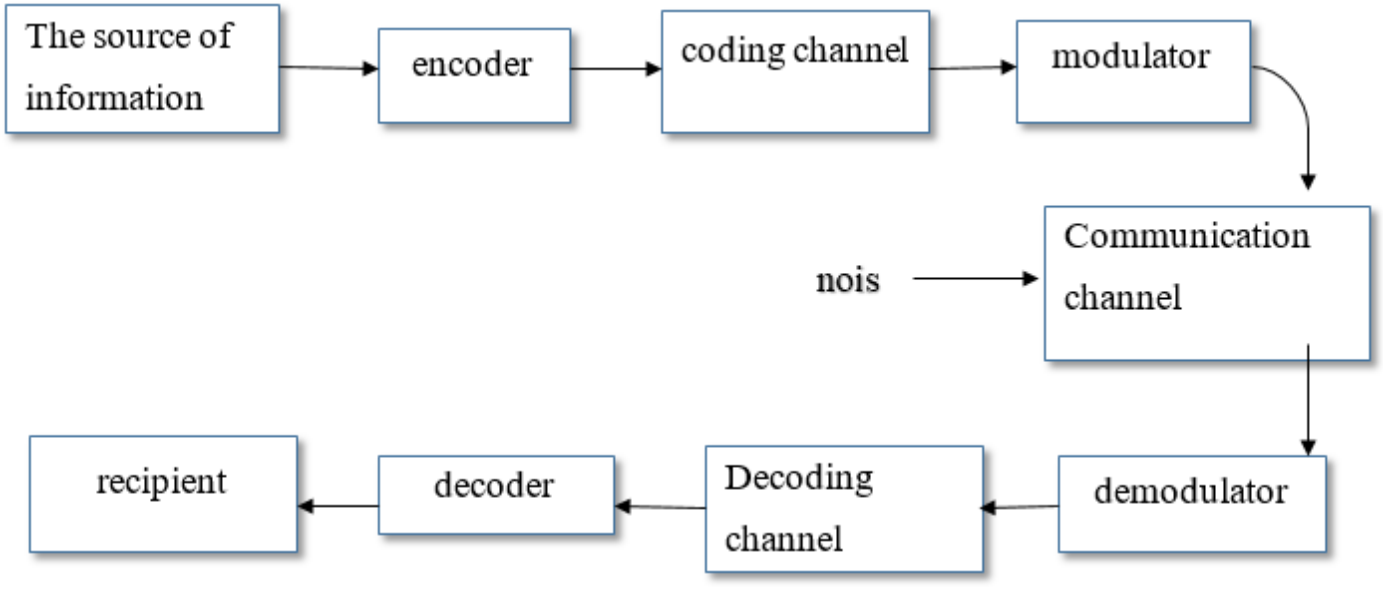
ABSTRACT: Coding theory (English: coding theory) is a branch of mathematics (mathematics) and computer science (computer science) that deals with error-prone situations in the process of transmitting data over communication channels. noise channels), using sophisticated methods that make the majority of errors that occur can be corrected. It also deals with the characteristics of the codes (codes), and thus makes them suitable for specific applications.

One of the most important scientific and technical problems at the present stage is the creation of automated control and monitoring systems for performing various tasks. In the process of automated control and monitoring, there is an intensive exchange of information between the individual parts of the systems, while the volume of information, as well as the speed of processing and transmitting it, are constantly growing. Increasingly high requirements are imposed on the reliability of transmitted messages, which leads to the need to apply special measures that reduce the frequency of errors to a certain acceptable level.

1. INTRODUCTION

One of the most effective measures is the use of error-correcting coding.The purpose of my work was to describe the properties of linear and cyclic codes, solve the problems of encoding and decoding linear (including cyclic) codes using syndromes (syndromic polynomials), which allow detecting and correcting errors. The term catching errors arose from the fact that we described a procedure (algorithm) by which we find and correct errors in received messages (n, k)- code, if these errors are located on the section of length $\leq m$, where $m = n - k$.

The beginnings of mathematical coding theory go back to the seminal paper by Claude Shannon (1948) and the invention of Hamming codes (1950). Thus began the history of error-correcting coding. The source and recipient of information are interconnected by the following scheme of the canonical infra communication system.



"Catching Errors in Cyclic Codes"
"Detection of Errors in Cyclic Codes"
« Error - Trapping in Cyclic Codes »

In the encoder, the input input message is written into a digital sequence a (information word) in alphabetical $\boldsymbol{F_q}$order (most often in bits). In the encoder channel , the information word is converted (encoded) into a code word c, written in the same alphabet $\boldsymbol{F_q}$. This is the most important part of coding. Since discrete symbols are not suitable for transmission over physical communication channels, a modulator is used (recording of alphabet units), through which the converted message enters the communication channel, which can be affected by noise , and then this physical message (generally speaking, distorted) enters demodulator, where it is converted into a digital sequence $\overline{v}$. Errors are corrected in the decoder channel (if it can be done) and a digital sequence is obtained $\hat{\bar{c}}$ (usually $\hat{\bar{c}} = \overline{c}$). In the decoder $\hat{\bar{c}}$, it is converted into an information word $\hat{\bar{a}}$and sent to the recipient. Ideally $\hat{\bar{a}} = \overline{a}$, otherwise a decoding error occurs .

2. LITERATURE REVIEW HISTORY

2.1 Block Line Codes

Data entering the communication system from the source of information is first of all processed by the source encoder, which represents this data in some digital alphabet (for example, as a sequence of characters from the final field $\boldsymbol{F_q}$). Thus, the input signal from the information source is written as an **information word** $\overline{a}$(this is the final sequence of characters in the chosen alphabet $\boldsymbol{F_q} = \{0, 1, \ldots, q-1\}$). The channel encoder converts the information word a into a longer word c (usually written in the same alphabet $\boldsymbol{F_q}$), called **a codeword** . The modulator then converts each codeword symbol into a corresponding analog symbol from a finite set of valid analog symbols. A sequence of analog symbols is transmitted over a channel, which can be subject to various noise, distortion, and interference. Therefore, the output of the channel, generally speaking, differs from the input. The demodulator converts each symbol received at the input of the channel into a sequence of symbols of the selected digital alphabet. The demodulated sequence is called **the received word**, we will denote it by the symbol $\overline{v}$. Due to potential errors, generally speaking, $\overline{v} \neq \overline{c}$. The task of the decoder, using the redundant record $\overline{v}$for the information word $\overline{a}$, is to recover the corresponding code word $\overline{c}$, and then $\overline{a}$.

Example 1 . Let $\overline{a} = (110)$ in the alphabet $F_2 = \{0,1\}$. Let's denote $\overline{c} = (111111111100000)$.

The word $\overline{c}$is obtained from $\overline{a}$repeating each character $\overline{a}$five times. As a result of channel noise, the word$\overline{v}$ =(110110011100011). We see that in each consecutive five characters

$$11011'00111'00011$$

more often, respectively, symbols 1,1,0,and therefore we conclude that the code word

$$\overline{c} = (111111111100000).$$

The code used is called a **repeat code** . It is reliable but very slow.

Example 2 . Each information word $\overline{a}$of length$k$,

$\overline{a} = (a_1, \ldots, a_k)$in the alphabet we F_2match the code word c of length

$k+1$, $\overline{c} = (c1, \ldots, ck, ck+1)$,

where $c_i = a_i$, if $1 \leq i \leq k$, and $c_{k+1} = a_1 + \cdots + a_k$. Such a code is called **a parity-check code** . If no more than one error was made in the code word during the passage of the communication channel, then it can be found out from the received word v, although the distorted symbol cannot be determined.

It can be seen from the above examples that an increase in the length of code words, generally speaking, increases the ability of the code to recover the code word after the distortion of its symbols in the communication channel.

In the future, we will study the so-called block codes. This means that we consider informational words in the alphabet $\mathbf{F_q}$of the same length k. It is clear that the set of informational words is the space $\mathbf{F_q^k} = \{(a_1, \ldots, a_k) \mid a_i \in F_q, i = 1, \ldots, k\}$. The power of this set is q^k. Each information word a is associated with the word $\overline{c} \in F_q^n$. Since different information words must correspond to different code words, the cardinality of the set of C code words is also equal to q^k, and therefore for we have $\mid C \mid = q^k < q^n$, and hence $C$is a proper subset in $\mathbf{F_q^k}$.

Thus, to specify the encoding of information words of length $k$into code words of length n, it is sufficient to specify a one-to-one mapping $\mathbf{F_q^k}$in $\mathbf{F_q^n}$. Since the most natural (and easily implemented) mappings of a linear space $\mathbf{F_q^k}$to $\mathbf{F_q^n}$ are linear mappings, the images of such mappings are called **linear codes** . Therefore, a linear code for information words from $\mathbf{F_q^k}$is any subspace of $C \subset \mathbf{F_q^n}$dimension k. We denote $C(n,k)$the linear length code n with the length of information words as k-code.

So, to construct a linear (n,k)-code in the alphabet $\mathbf{F}_q$and describe the encoding procedure in this code, we proceed as follows:

“Catching Errors in Cyclic Codes"
"Detection of Errors in Cyclic Codes"
« Error - Trapping in Cyclic Codes »

We choose $k$linearly independent vectors in the space $\mathbf{F_q^n}$: $g_1, g_2, \dots, g_k$. Their linear span

$$L(\overline{g}_1, \overline{g}_2, \dots \overline{g}_k) := \{\sum_{i=1}^{k} \alpha_i\, g_i \mid . \alpha_1 \dots, \alpha_k \in \mathbf{F_q}\}$$

forms a linear (n, k)-code.

Now the (n, k)-code itself as a set of code words is defined. But the coding procedure, that is, the rule for the transition of information words into code words, is determined by a specific isomorphic mapping

$$\mathbf{F_q^k} \overset{\varphi}{\to} L(\overline{g}_1, \dots \overline{g}_k).$$

Since the isomorphism of linear spaces is completely determined by the rule for comparing the basis vectors of these spaces, we fix a $\mathbf{F_q^k}$standard basis in the space of information words

$$\overline{f_1} = (1, 0, 0, \dots, 0),$$
$$\overline{f_2} = (0, 1, 0, \dots, 0),$$
$$\dots\dots\dots\dots\dots$$
$$\overline{f_k} = (0, 0, 0, \dots, 1).$$

vectors $\overline{g}_1$ in coordinate form for the standard space basis $\mathbf{F_q^n}$:

$\overline{g}_1 = (g_{11}, g_{12}, \dots, g_{1n})$,
$\overline{g}_2 = ((g_{11}, g_{12}, \dots, g_{1n})$,

$$\dots\dots\dots\dots\dots$$

$\overline{g}_k = ((g_{11}, g_{12}, \dots, g_{1n})$,

The mapping $\overline{f}_j \rightleftharpoons \overline{g}_j, j = 1, \dots, k$, gives the encoding procedure for the linear-th (n, k)-code.

Choosing in $L(g1, \dots, gk)$ another basis $\overline{g}'_1$,..., $\overline{g}'_k$, we get the same code C, but with a different rule for converting information words into code words.

Denote by the Gmatrix composed of the coordinate rows of the basis vectors $\overline{g}_1, \dots, \overline{g}_k$. Then the comparison $\overline{a} \to \overline{c}$given by the equality

$$\overline{c} = \overline{a}G,$$

defines the rule for constructing code words of a linear (n, k)-code C. The matrix $G$is called **the generator matrix of the** linear (n, k)-code C.

The matrix G has rank equal to k, and therefore among its columns there are k linearly independent.

Let, for definiteness, these be the first k columns. It follows from the linear algebra course that by elementary transformations over rows, a matrix Gcan be reduced to the form

$$G_0 = \begin{pmatrix} g_{11} & \dots & g_{1n-k} & 1 & 0 & \dots & 0 \\ g_{21} & \dots & g_{2n-k} & 0 & 1 & \dots & 0 \\ \vdots & \dots & \vdots & \vdots & \vdots & \dots & \vdots \\ g_{k1} & \dots & g_{k,n-k} & 0 & 0 & \dots & 1 \end{pmatrix}$$

and therefore the rows of this matrix determine the coordinate rows of the new basis of the subspace of the same code C, and the coding rule

$$\overline{a} \to \overline{c} = \overline{a}G_0$$

we will call the systematic (or canonical) form of the C code.

With systematic coding, the last $k$symbols of the code word $\overline{c}$corresponding to the information word $\overline{a}$coincide with the coordinates of the vector $\overline{a}$. We will write the $G = (AI_k)$matrix in the form G_0, where $I_k$is the unit matrix of order k, and $A$is the matrix of size $k \times (n - k)$, notation (AI_k), means the concatenation of matrices $A$and I_k.

For a linear (n, k)-code $C$with a generating matrix G, denote by the Hmatrix whose rows form the basis of the solution space of the system of linear homogeneous equations over the field $\mathbf{F}_q$with the matrix G:

$$GX = 0, X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, 0 = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Obviously, $GH^T = 0$where the sign " T " means the transposition of the corresponding matrix, 0 is a zero matrix of size $k \times (n - k)$. We have: rank $H = n - k$. Let's denote $m = n - k$. Obviously, it $m$characterizes the number of redundant (check) code symbols C.

follows $HG^T = 0$from equality $GH^T = 0$.

The matrix H is called the code C **check matrix** .

It is easy to check that if G = G 0=(AI_k), then

H =(($I_m(-A)^T$).

From the above reasoning, it can be seen that with each linear (n,k)- code $C$with a generator matrix $G$and a check matrix H, it is possible to associate a linear (n,m) - code $C_g$with a generator matrix $H$and a check matrix G. Codes $C$and $C_g$are called **dual** to each other, and any code vector from $C$is orthogonal (in the sense of the standard scalar product) to each vector from C_g, and vice versa.

Example 3 . Build Linear$(7,4)$ - code Cabove F_2. We choose a 4×7 matrix of rank 4:

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

Display $\overline{a} \to \overline{a}G = \overline{c}$sets code

$$(0\,0\,0\,0) \to (0\,0\,0\,0\,0\,0\,0)$$
$$(1\,0\,0\,0) \to (1\,10\,1\ 0\,0\,0)$$
$$(0\,1\,0\,0) \to (0\,1\,1\,0\,1\,0\,0)$$
$$(0\,0\,1\,0) \to (0\,0\,1\,1\,0\,1\,0)$$
$$(0\,0\,0\,1) \to (0\,0\,0\,1\,1\,0\,1)$$
$$(1\,1\,0\,0) \to (1\,0\,1\,1\,1\,0\,0)$$
$$(0\,1\,1\,0) \to (0\,1\,0\,1\,1\,1\,0)$$
$$(0\,0\,1\,1) \to (0\,0\,1\,0\,1\,0\,1)$$
$$(1\,0\,1\,0) \to (1\,1\,1\,0\,0\,1\,0)$$
$$(0\,1\,0\,1) \to (0\,1\,1\,1\,0\,0\,1)$$
$$(1\,0\,0\,1) \to (1\,1\,0\,0\,1\,0\,1)$$
$$(1\,1\,1\,0) \to (1\,0\,0\,0\,1\,1\,0)$$
$$(0\,1\,1\,1) \to (0\,1\,0\,0\,0\,1\,1)$$
$$(1\,1\,0\,1) \to (1\,0\,1\,0\,0\,0\,1)$$
$$(1\,0\,1\,1) \to (1\,1\,1\,1\,1\,1\,1)$$
$$(1\,1\,1\,1) \to (1\,0\,0\,1\,0\,1\,1)$$

We find the check matrix H. To do this, consider a system of linear homogeneous equations with a matrix G:

$$\begin{cases} x_1 + x_1 + x_1 = 0 \\ x_2 + x_3 + x_5 = 0 \\ x_3 + x_4 + x_6 = 0 \\ x_4 + x_5 + x_7 = 0 \end{cases}$$

It has a trapezoidal form (see matrix G), therefore x_5, x_6, are $x_7$its free unknowns.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
f_1	one	0	one	one	one	0	0
f_2	one	one	one	0	0	one	0
f_3	0	one	one	one	0	0	one

Consequently,

$$H = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

This code is not systematic. However, from the matrix, by G elementary row transformations, we arrive at the matrix

“Catching Errors in Cyclic Codes"

"Detection of Errors in Cyclic Codes"

« Error - Trapping in Cyclic Codes »

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

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

$$=G_0=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}A,\ \text{where}A=\begin{pmatrix}1&1&0\\1&1&0\\0&1&0\\1&0&0\end{pmatrix}$$

That's why

$$H_0=((-A)^TI_{n-k})=\begin{pmatrix}1&1&0&1&1&0&0\\1&1&1&0&0&1&0\\0&0&0&0&0&0&1\end{pmatrix}$$

Consider the information word $\overline{a}=(0110)$. When encoding with a matrix G, we have:

$$\overline{a}\to\overline{c}=\overline{a}G=(0\,1\,0\,1\,1\,1\,0),$$

and when encoding with matrix G_0:

$$\overline{a}\to\overline{c}=\overline{a}G_0=(0\,1\,1\,0\,1\,0\,0).$$

Example 4 . Construct a systematic (7,3) -code in the alphabet F_3. To solve the problem, you need to find a matrix of size 3 × 7 and rank 3 over the field

$F_3=\{0,1,2\}$. We choose two non-proportional vectors of length 7:

$$g_1=(1\,0\,2\,1\,1\,0\,0),\quad g_2=(2\,1\,1\,0\,2\,1\,0).$$

It is clear that any vector g3 with a nonzero last coordinate forms, together with $\overline{g}_1$, $\overline{g}_2$, a system of three linearly independent vectors.

Let $g_3=(0\,1\,2\,1\,2\,0\,1)$. Then matrices

$$G=\begin{pmatrix}\overline{g}_1\\\overline{g}_2\\\overline{g}_3\end{pmatrix}=\begin{pmatrix}1&0&2&1&1&0&0\\2&1&1&0&2&1&0\\0&1&2&1&2&0&1\end{pmatrix}.$$

Since over the field the F_3determinant

$$\begin{vmatrix}1&0&2\\2&1&1\\0&1&2\end{vmatrix}=2\neq0,$$

then we bring the matrix G to a systematic form:

$$G=\begin{pmatrix}1&0&2&1&1&0&0\\2&1&1&0&2&1&0\\0&1&2&1&2&0&0\end{pmatrix}\sim\sim\begin{pmatrix}1&0&2&1&1&0&0\\0&1&0&1&0&1&0\\0&1&2&1&2&0&1\end{pmatrix}\_$$

$$\sim\sim\sim\begin{pmatrix}1&0&2&1&1&0&0\\0&1&0&1&0&1&0\\0&0&2&0&2&2&1\end{pmatrix}\_\begin{pmatrix}1&0&0&1&2&1&2\\0&1&0&1&0&1&0\\0&0&2&0&2&2&1\end{pmatrix}\_$$

$$==\begin{pmatrix}1&0&0&1&2&1&2\\0&1&0&1&0&1&0\\0&0&1&0&1&1&2\end{pmatrix},\ \_G_0=(I_3\ \ A)$$

Where

$$A=\begin{pmatrix}1&2&1&2\\1&0&1&0\\0&1&1&1\end{pmatrix}.$$

From here,

$$H_0=((-A)^TI_4)=\begin{pmatrix}2&2&0&1&0&0&0\\1&0&2&0&1&0&0\\2&2&2&0&0&1&0\\1&0&1&0&0&0&1\end{pmatrix}$$

We are now able to encode the information word a =(210) in a systematic code form. We have

$$\overline{a}\to\overline{a}G_0=(2\,1\,0\,0\,1\,0\,1).$$

“Catching Errors in Cyclic Codes"
"Detection of Errors in Cyclic Codes"
« Error - Trapping in Cyclic Codes »

Consider a block linear (n,k)- code Cin the alphabet F_q. Let be $G$the generating matrix of this code. Let us fix the check matrix $H$of this code. Then $H \cdot G^T = 0$, where $k \times n, (m \times n), (m \times k)$are the dimensions of the matrices , respectively G, H, O. Fixing the matrix Hmeans that only codewords $\overline{c}$(i.e., vectors from the space of Ccodewords) satisfy the equality $H\overline{c}^T = 0$. Therefore, if for the received word $\overline{v}$we have $H\overline{v}^T = 0.$, we conclude that the code word has been distorted in the channel c, and therefore $\overline{v} = \overline{c} + \overline{e}$.

Since $H\overline{v}^T = H\overline{c}^T + H\overline{e}^T = H\overline{e}^T$, then the vector $H\overline{e}^T$ indicates the presence of errors in the received message and it is called the error sector **syndrome .** However, if the error vector coincides with some code vector, then we will get that $H\overline{e}^T = O$, and therefore we will not know about errors in the communication channel, and therefore we will perform incorrect decoding.

Definition 1. An arbitrary vector syndrome with $\overline{y} \in \mathbf{F}^{q^n}$respect to a linear (n,k)code C(or its check matrix H) is a vector $H\overline{y}^T \in \mathbf{F}_q^m$. Designated $S(\overline{y}) = (s_0, s_1, \ldots, s_{m-1})$.

Assuming that the communication channel is sufficiently "good, i.e., the number of distortions in one code word" is small , we can exclude the situation $S(\overline{e}) = 0$if code vectors are chosen knowingly different from possible error vectors.

Definition 2 . Let be $\overline{y} = (y_0, y_1, \ldots, y_{n-1})$an arbitrary vector from $\mathbf{F}_q^m$. **The Hamming weight of a** vector $\overline{y}$is the number of non-zero coordinates of this vector.

Definition 3 . **Hamming distance** between vectors
$\overline{x}$ и $\overline{y} \in \mathbf{F}_q^m$is the number of non-zero coordinates of the vector $\overline{x} - \overline{y}$.
If $w(\overline{x})$means the Hamming weight for $\overline{x}$, and $d(\overline{x}, \overline{y})$is the Hamming distance between $\overline{x}$and $\overline{y}$, then we have

$$d(\overline{x}, \overline{y}) = w(\overline{x} - \overline{y}).$$

It is easy to check that the Hamming distance on $\mathbf{F}_q^n$defines a matrix on the space $\mathbf{F}_q^n$, and the triangle inequality holds

$$d(\overline{x}, \overline{z}) + d(\overline{y}, \overline{z}) \geq d(\overline{x}, \overline{y})$$

for any $x, y, z \in \mathbf{F}_q^n$.

Definition 4 . Let a linear (n,k)-code over be given $\mathbf{F}_q$. The minimum code distance for C(denoted by d_c) is the minimum weight of non-zero code words; $d_c = min_{(0 \neq c \in C)} w(c)$.

Since the difference of code words is a code word (since is a Csubspace of $\mathbf{F}_q$.), we have
$d_c = min_{(c_1, c_2 \in C, c_1 \neq c_2)}(d(c_1, c_2))$.

For example, let $\overline{c}_1 = (101011)$, $\overline{c}_2$=(111001) $\in F_2^6$. Then $w(\overline{c}_1) = 4$, $w(\overline{c}_2) = 4$, $d(\overline{c}_1, \overline{c}_2) = w(\overline{c}_1 - \overline{c}_2) = 2$. This shows that if $d_c \geq 5$, and the communication channel is such that no more 3^xerrors can be made in each word passing through the channel, then for the received word v its syndrome (and hence the syndrome of the error vector) will be different from the zero vector.

In coding theory, the following principle of decoding in channels with noise is confessed: "the received word is $\overline{v}$decoded into the nearest (in the sense of the Hamming distance) code word. If for a given v there are at least two code words for which $d(v, \overline{c}_1) = d(v, \overline{c}_2) \geq d(v, \overline{c})$for all code words $\overline{c} \in C, c \neq \overline{c}_1, \overline{c} \neq \overline{c}_2$, then a collision occurs, leading to decoding failure".

Definition 5 . A sphere of radius $t, t \in N$centered at a point $\overline{y}_0 \in \mathbf{F}_q^n$is a set of vectors $\overline{y} \in \mathbf{F}_q^n$for which $d(\overline{y}_0, \overline{y}) \leq t$. (Designation: $B_t(y_0)$).

Definition 6 . A code is said $C$to correct $t$errors and less if each $y \in \mathbf{F}_q^n$sphere $B_t(y)$contains no more than one codeword.

Theorem 1 . Linear (n,k)-code Ccorrects $t$errors and less if and only if $d_C \geq 2t + 1$.

Proof . If the code C corrects $t$errors, then for any codewords $\overline{c}_1, \overline{c}_2$, $\overline{c}_1 \neq c2$the assumption $d(\overline{c}_1, \overline{c}_2) \leq 2t$follows $w(\overline{c}_2 - \overline{c}_1) \leq 2t$.. Take as y a vector whose first $t$non-zero coordinates are the same as those of the vector $\overline{c}_2 - \overline{c}_1$, and the remaining coordinates are equal to 0. For such a vector y we have $d(y, \overline{c}_2 - \overline{c}_1) = t, d(\overline{y}, 0) \leq t$, and therefore the sphere radius $t$centered at the point y contains two code words $\overline{0}$and $\overline{c}_2 - \overline{c}_1$, which is contradictory.

And vice versa, if $d_C \geq 2t + 1$, then for each $y \in F_q^*$we have

$$d(\overline{y}, \overline{c}_1) + d(\overline{y}, \overline{c}_2) \geq d(\overline{c}_1, \overline{c}_2) \geq 2t + 1$$

for any $\overline{c}_1, \overline{c}_2 \in C$, $\overline{c}_1 \neq \overline{c}_2$, which means that at least one of the inequalities

$$d(\overline{y}, \overline{c}_1) > t, d(\overline{y}, \overline{c}_2) > t$$

is performed, i.e. sphere $B_t(\overline{y})$contains at most one vector

$$\overline{c}_1, \overline{c}_1 \in C, \overline{c}_1 \neq \overline{c}_2.$$

Theorem 2 . Let a C-linear (n,k)-code over F_q. For each code word $\overline{c}$with Hamming weight $l, l \neq 0$, there are $l$columns of the check matrix $H$of this code, such that the linear combination of these columns, whose coefficients are non-zero coordinates

of the vector $\overline{c}$, is equal to zero. And vice versa, if there are $l$matrix columns $H$and non-zero , $l$field $F_q$elements $a_1,\dots,a_l$so that the linear combination of these columns with coefficients $a_1,\dots,a_l$is equal to zero, then there is a code vector whose non-zero components are equal $a_1,\dots,a_l$and located on the numbers of the selected columns of the matrix H.

Proof. Indeed, for any vector , $v=(v_1,\dots,v_n)$the product $\overline{v}H^T=v_1\overline{h}_1+\cdots+v_n\overline{h}_n$, where $\overline{h}_i$is the i-th column of the matrix H. Hence, the validity of the theorem is easily verified in both directions.

Corollary 1 . If the check matrix of a Hlinear (n,k)-code is such that any of its s columns are linearly independent over Fq, then the (n,k)-code Cin the alphabet $F_q$has a minimum code distance $\geq s+1$.

Consequence 5 . Let C-linear (n,k)-code over F_2. Then $d_C$it is equal to the least number of columns of the check matrix Hwhose sum is equal to $\overline{O}$.

Example 5 . Let $C-(7,4)$-code over F_2. with check matrix

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

The matrix Hhas no proportional columns (hence any two columns are linearly independent). But the sum of the second, third and fourth columns is zero. Therefore, $d_C=3$, and, by virtue of Theorem 1, this code corrects single errors.

From now until the end of this section, we will consider linear (n,k)codes in the alphabet F_2, although the theory presented is also valid over an arbitrary alphabet F_q.

If a code vector was sent over the communication channel $\overline{c}$AND there were $l$symbol distortions in the channel, then for the received word v we have $d(c,v)=l$, which means that at , the $l<d_C$word $\overline{v}$will not coincide with any code word. For such a code, a corruption of 1 d_C-1or fewer symbols will not result in a codeword. This means that a linear (n,k)-code $C$with a minimum code distance dC is able to detect the occurrence of errors in the amount d_C-1or less (but not necessarily correct them).

We will assume that the number of distortions in code words after passing through a noisy channel is less than d_C, i.e. the probability of the opposite event is quite small. Only under these assumptions does it make sense to use this code.

Let be Ca given linear (n,k)-code in the alphabet F_2. The code Ccontains 2^kwords, and in space F_2^nthere are 2^nvectors. We divide the set of vectors F_2^ninto classes as follows:

the class K_1consists of 2^kcode words;

the class K_2consists of vectors of the form $\overline{e}_2+\overline{c}$, where the $\overline{c}$entire set runs C, and the vector $\overline{e}_2$has the least weight among the vectors from $F_q\setminus C$;

the class K3 consists of vectors of the form $\overline{e}_3+\overline{c},\overline{c}\in C$, $\overline{e}_3$the vector of the smallest Hamming weight from $F_q\setminus(C\cup K_2)$, and so on.

Each class $K_i$contains exactly 2^kvectors, these vectors are different, and the classes $K_i$and $K_j$do not intersect at $i\neq j$(Indeed, from $K_i\cap K_j\neq\emptyset$ it follows that there are vectors $\overline{c}_l$and $\overline{c}_m$from Csuch that
$\overline{e}_i+\overline{c}_l=e_i+\overline{c}_m$.

Let for definiteness $j>i$. Then we have

$$\overline{e}_j=\overline{e}_i+(c_l+\overline{c}_m)=\overline{e}_i+\overline{c},\text{где }\overline{c}\in C,$$

but this contradicts the choice $\overline{e}_j$as the vector with the least weight and not lying in any of the previous classes.

The classes $K_i$will be called the cosets of the space $F_2$by the code C. It is clear that we will have a total of six $2^{n-k}=2^m$classes. Selected vectors $\overline{e}_i\in K_i,(\overline{e}_i=0)$, have the smallest Hamming weight in the class K_i, but in this class, in addition to the vector , there $\overline{e}_i$may be other vectors with a weight equal to the weight $\overline{e}_i$.

results $F_2^n$of splitting the set of space vectors into classes using the table

The allocated vectors $\overline{e}_i,i=2,3,\dots,2^m$,are called class leaders K_i. The leader of the class $K_1=C$is the null vector $\overline{e}_1$=(0,...,0).

Leader	Class elements$K_j\{\overline{e}_j\}$					leader syndrome
$\overline{c}_1=\overline{0}$	$\overline{c}_2$	…	$\overline{c}_j$	…	$\overline{c}_{2^k}$	$S(\overline{0}=0)$
$\overline{e}_2$	$\overline{c}_2+\overline{e}_2$	…	$\overline{c}_1+\overline{e}_2$	…	$\overline{c}_{2^k}+\overline{e}_2$	$s(\overline{e}_2)$
$\overline{e}_3$	$\overline{c}_2+\overline{e}_3$	…	$\overline{c}_j+\overline{e}_3$	…	$\overline{c}_{2^k}+\overline{e}_3$	$s(\overline{e}_3)$
…..	…..	…	…..	…	…..	…….
$\overline{e}_{2^m}$	$\overline{c}_2+\overline{e}_{2^m}$	…	$\overline{c}_1+\overline{e}_{2^m}$	…	$\overline{c}_{2^k}+\overline{e}_{2^m}$	$s(\overline{e}_{2^m})$

“Catching Errors in Cyclic Codes"
"Detection of Errors in Cyclic Codes"
« Error - Trapping in Cyclic Codes »

Example 6 . Let C-linear (6,3)-code cby generating matrix

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

The generating matrix allows you to write all code words as all possible linear combinations of matrix rows $G$with coefficients from F_2. We have

$$\overline{c}_1 = (000000), \overline{c}_2 = (101010), \overline{c}_3 = (010110), \overline{c}_4 = (110001)$$
$$\overline{c}_5 = (111100), \overline{c}_6 = (011011), \overline{c}_7 = (100111), \overline{c}_8 = (001101)$$

Therefore, we have such a class table

(000000)	(101010)	(010110)	(110001)	(111100)	(011011)	(100111)	(001101)
(100000)	(001010)	(110110)	(010001)	(011100)	(111001)	(000111)	(101101)
(010000)	(111010)	(000110)	(100001)	(101100)	(001011)	(110111)	(011101)
(001000)	(100010)	(011110)	(111001)	(110100)	(010011)	(101111)	(000101)
(000100)	(101110)	(010010)	(110101)	(111000)	(011111)	(100011)	(001001)
(000010)	(101000)	(010100)	(110011)	(111110)	(011001)	(100101)	(001111)
(000001)	(101011)	(010111)	(110000)	(111101)	(011010)	(100110)	(001100)
(100100)	(001110)	(110010)	(010101)	(011000)	(111111)	(000011)	(101001)

Consider an arbitrary class

$$K_j = \{ \overline{e}_j + \overline{c}_j \mid \overline{c}_j \in C, i = 1,1, \dots, 2^k \}.$$

Then we have

$S(\overline{e}_j + \overline{c}_j) = S(\overline{e}_j) + S(\overline{c}_j) = S(\overline{e}_j), i = 1, \dots, 2^k.$

This means that elements of the same class have equal syndromes. Using the principle of choosing the coset leader, we, by virtue of the decoding method to the nearest code number, obtain the decoding algorithm by the coset leader:

Step I Calculate the received vector syndrome $\overline{v}$: $S(\overline{v}) = H\overline{v}^T$.

Step II In the table of syndromes, we find the line for which $S(\overline{v}) = S(\overline{e}_j)$.

Step III Decode the received word $\overline{v}$for word $\overline{c} = \overline{v} + \overline{e}_j$.

This decoding method minimizes the decoding error, since, according to our agreement, the probability of the word v appearing at the channel input decreases with an increase in the number of distorted symbols. This encoding method is also supported by the fact that the Hamming distance between the received word $\overline{v}$and the code word $\overline{c}$(see step III) is not greater than the distance between $\overline{v}$and another (other than $\overline{c}$) code word. Really,

$$d(\overline{v}, \overline{c}') = w(\overline{v} + \overline{c}') = w(\overline{e}_j + \overline{c} + \overline{c}') = w(\overline{e}_j + \overline{c}''), c'' = \overline{c} + \overline{c}'$$

And since the leader $\overline{e}_j$and the vector $\overline{e}_j + \overline{c}''$are in the same adjacent class, then, by virtue of the choice of the leader in the adjacent class, we conclude

$w(\overline{e}_j) \leq w(\overline{e}_j + \overline{c}'')$, and therefore $d(\overline{v}, \overline{c}) \leq d(\overline{v}, \overline{c}')$.

The above reasoning also shows that in cases where the coset leader can be chosen in a unique way, then decoding by the coset leader has a minimum decoding error. | indent Denote by the Anumber of coset leaders with weight j . The numbers $A_0, A_1, \dots, A_n$will be called **the weight distribution of leaders** .

In some cases, the weight distribution makes it possible to estimate the probability of a decoding error. Let us consider a binary symmetric channel (BSC) with the probability $p$of one symbol distortion, i.e.

$$P(0 \mid 1) = P(1 \mid 0) = p$$

(here $P(0 \mid 1)$ means the probability that the sent character 1 will go to the character 0).

Since a decoding error occurs if and only if the error vector is not the leader of an adjacent class, we get that the probability of incorrect decoding in such a channel is equal to

$$P(E) = 1 - \sum_{i=0}^{n} A_i p^i \ (1-p)^{n-i}.$$

In Example 15 considered above, for a linear (6,3)-code with generator matrix G, we have the following weight distribution:

$A_0 = 1, A_1 = 6, A_2 = 1, \ A_3 = A_4 = A_5 = A_6 = 1.$

That's why

$$P(E) = 1 - (1-p)^6 - 6p(1-p)^5 - p^2(1-p)^4.$$

For $p = 10^{-2}$we get $P(E) \approx 1.4 \cdot 10^{-3}$.

Theorem 3 . Let a C −linear (n,k)-code be the cminimum code distance d_C. Then all weight vectors $t = \frac{d_C - 1}{2}$ and less can be examined as coclass leaders, but at least one weight vector $t+1$cannot be a coset leader.

The assertion of the theorem follows from the definition of an error-correcting $t$code; a the considered example 16 illustrates what was said for $t = 1$. (For example, a vector (010001)cannot be a leader).

Example 7 . Consider the (7,4) -code c by the generating matrix

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

The check matrix Hhas size (7,3) so the number of cosets is 23=8 and it looks like

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

In a matrix H, no two columns are equal, but the first column is equal to the sum of the last two. Therefore, $d_C = 3$. Therefore, the code Ccorrects single errors. This is also evidenced by the fact that any two vectors with weight 1 cannot be in the same coset. There are only 7 such vectors, which, together with the zero row, give eight different leaders, i.e. each coset has a weight vector ≤ 1 as its leader, which makes it possible to uniquely correct single errors.

For example, if a message v=(0010010) is received, we compute $S\overline{v} = H\overline{v}^T = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$and compare with the weight vector syndromes of 1:

$$S(1000000) = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \qquad S(0100000) = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

$$S(0010000) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \qquad S(0001000) = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

$$S(0000100) = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \qquad S(0000010) = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

$$S(0000001) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

Where do we conclude that the error is in the fourth position, and therefore $\overline{c} = (0011010)$.

If there was a double error, for example, $\overline{e} = (0010010)$,and we got $\overline{v} = (1001110)$, then the calculations give $S(\overline{v} = S(\overline{e})) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$. Now, when decoding by the method of adjacent classes, we would have an erroneous result $\overline{e} = (0000001)$, $\overline{c} = (1001111)$instead of a sent message $\overline{c}_0 = (1011100)$.

Thus, exaggerating the possibilities of decoding using the coset method, we can come to an erroneous result.

2.2 Hamming Codes

The Hamming code was the first example of a linear (n,k)-code to detect and correct single errors. We fix a natural number $m \geq 3$and consider a matrix whose columns are the m-valued notation of numbers from 1 to 2^{m-1}. We will get a matrix H of size $2^{m-1} \times m$, which we will consider as a check matrix of a linear (n,k)-code, where $n = 2^m - 1, k = n - m = 2^m - m - 1$. This code is called the Hamming code. Obviously, the columns of the matrix H are different, but there is a column equal to the sum of the other two. Therefore, this code corrects single errors. There are $2^{n-k} = 2^m$cosets, and vectors

$$(000\cdots00), (100\cdots00), (010\cdots00), \ldots, (000\cdots10), (000\cdots01)$$

belong to different cosets.

Each sphere of radius 1 contains n + 1 points from F_2^n, and therefore non-intersecting spheres with centers in code words contain $(n+1)2^k = 2^m \cdot 2^k = 2^n$points from F_2^n, i.e. these spheres cover the entire space without intersections F_2^n.

Definition 7 . A code correcting t-errors and less is called **a perfect code** if all vectors with Hamming weight $\leq t$are coset leaders and there are no other leaders.

Therefore, the Hamming code is a perfect code that corrects single errors. In addition to the Hamming code, there is another binary perfect (23,12) Galey code. The (7,4)-code considered in Example 17 is a binary Hamming code. There are also perfect codes in the alphabet .F_q

Let us construct a generalized Hamming code. Let $m \geq 3$. We put n = $\frac{q^m-1}{q-1}$and denote by α the primitive element of the field F_{q^m}. Then the order of α in the group $F_{q^m}^*$is $q^m - 1$. Recall that each element of the field $F_{q^m}$is uniquely represented by a linear combination of elements $1, \alpha, \ldots, \alpha^{m-1}$with coefficients from F_q:

$$\alpha^j = m\sum_{i=0}^{m-1} a_{ij}\alpha^i, j = 0,1,\ldots,n-1.$$

We form a matrix H , whose columns are $\overline{h}_j = \begin{pmatrix} a_{0j} \\ \vdots \\ a_{m-1,j} \end{pmatrix}$, i.e.

$$H = \begin{pmatrix} a_{00} & a_{01} & \ldots & a_{0,n-1} \\ a_{10} & a_{11} & \cdots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1,0} & a_{m-1,1} & \cdots & a_{m-1.n-1} \end{pmatrix}$$

A generalized Hamming code is a linear (n,k)-code in the alphabet F_q, whose check matrix is equal to H(here $k = n - m$).

Theorem 4 . *The generalized Hamming code is a perfect single error correcting code* .

Proof . The different columns of the matrix represent different powers of α, and therefore their proportionality implies that $\alpha^i = a\alpha^j, a \in F_q, 0 \leq j < i \leq n-1$. But then $a = \alpha^{i-j}$, but because $1 = \alpha^{q-1} = \alpha^{(i-j)(q-1)}$, which is impossible, since $0 < (i-j)(q-1) < n(q-1) = q^{m-1}$, and the order of α is equal to q^{m-1}. Thus, for the considered code: $d_C \geq 3$, i.e. the code corrects single errors. Each sphere $B_1(\overline{y}), \overline{y} \in F_q$contains a point y and more $n(q-1)$points at a distance of 1 from y, i.e. $| B_1(\overline{y}) | = 1 + n(q-1) = q^m$. When the set of all $\overline{y}$code words runs through , then the set$q^k \cup_{\overline{c}\in C} B_1(\overline{c})$will contain $q^k \cdot | B_1(\overline{y}) | = q^{k+m} = q^n$vectors. Hence, these spheres cover the entire space F_q^n. Therefore, the generalized Hamming code is a perfect code.

2.3 Decoding algorithms based on algebraic features of cyclic codes.

We considered decoding methods based mainly on the property of linear cyclic codes that these codes are subspaces of a linear n −dimensional space over a field $GF(q)$. The cyclicity of the code is only a circumstance that allows the most simple implementation of these algorithms. In this section, we consider another class of decoding algorithms based on the finer algebraic structure of cyclic codes. This class is based on a simple algebraic technique for finding the erroneous components of the received sequence, which was first used by Peterson to decode binary codes and by Zierler and Hornstein to decode non-binary Bose-Chowdhury-Hokvinhem codes. For ease of reference, we will call the method of Peterson and Gornstein-Zierler the direct decoding method.

Empty Kis the finite extension of the field $GF(q)$containing all the roots of the polynomial $g(x)$, with coefficient $GF(q)$and m −degree Kover $GF(q)$. Let be$g(x)$ belongs to the exponent n, then n divides $q^m - 1$. It is known that in $K = GF(q^m)$there is an element $\mu$of order n, in the form of powers of which it is possible to represent all elements of the cyclic subgroup of the order of the $n$multiplicative group of the field K, including all the roots of the polynomial$g(x)$. Let's pretend that$g(x)$ generates a cyclic code A. Denote the set of roots of the polynomial$g(x)$ across
$\{\theta_1 = \mu^{l_1}, \theta_2 = \mu^{l_2}, \ldots, \theta_r = \mu^{l_r}\}$.

Since each code word $a(x)$from the cyclic code A is divided without remainder by the generating polynomial $g(x)$, then the roots listed above must also be the roots of any polynomial $a(x) \in A$.

Let us assume that the errors in the communication channel have been translated
$a(x)$ в $b(x) = a(x) + e(x)$.

“Catching Errors in Cyclic Codes"
"Detection of Errors in Cyclic Codes"
« Error - Trapping in Cyclic Codes »

If errors can in principle be corrected, then the syndrome $S(x) = b(x)\ mod\ g(x)$carries all the information about the errors that have occurred. Knowing this information is enough for correction, for example, using a filter and a selector. On the other hand, all information about errors is contained in the set of the following $r$elements of the field K: $s_{11} = b(\theta_1), s_{12} = b(\theta_2), \dots, s_{lr} = b(\theta_r)$, which can also be calculated using multi-cycle linear filters. Indeed, by supplying quantities instead of in comparison.

$$S(x) = c_0 + c_1 x + \cdots + c_{r-1}x^{r-1} \equiv b(x)\ mod\ g(x)$$

We get a system of equations

$$c_0 + c_1\theta_j + \cdots + c_{r-1}\theta_j^{r-1} = s_{r_j}, j = 1,2 \dots, r$$

Relatively $c_0, c_1 \dots, c_{r-1}$. The coefficient matrix of this system

$$\begin{bmatrix} 1 & \theta_1 & \theta_1^2 & \dots & \theta_1^{r-1} \\ 1 & \theta_2 & \theta_2^2 & \dots & \theta_2^{r-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \theta_r & \theta_r^2 & \dots & \theta_r^{r-1} \end{bmatrix}$$

Has a non-zero determinant since the latter is a Vandermonde determinant and θ_j, different roots of the polynomial $g(x)$.

Therefore, the system of linear equations given above has exactly one solution and the elements $S_{lj} = b(\theta_j)$, где o $= 1,2, \dots r$, completely define the syndrome.

From $S_{lj} = b(\theta_j)$it follows that $S_{lj} = e(\theta_j)$, since $a(\theta_j) = 0$. Since $e(x) = \sum_{i=0}^{n-1} \varepsilon_i\, x^i$, then the equalities

$s_{lj} = \sum_{i=0}^{n-1} \varepsilon_i\, (x^{lj})^l,\ j = 1,2, \dots r$ (3.0)

It can be considered as a system $r$of equations with respect to $n$variables, $\varepsilon_0, \varepsilon_1 \dots, \varepsilon_{n-1}$and any way to solve this system as a method that can be the basis of some decoding algorithm.

Each error vector is completely specified by listing the values of non-zero components and indicating the places where these non-zero components are located. Since $e(\mu) = \sum_{i=1}^{n-1} \varepsilon_i \mu^i$, the position of non-zero components can be specified using $t$elements $\mu^{i1}, \mu^{i2}, \dots, \mu^{it}$and the value of these components using elements $\varepsilon_{i1}, \varepsilon_{i2}, \dots, \varepsilon_{it}$.

Such a representation of the error vector in terms $t$of pairs of values is convenient for decoding purposes, since it is possible to reduce $\mu^{iu}, \varepsilon_{iu}$the problem of error correction to first determining the position of the erroneous components, and then calculating the values of these components.

2.3 Direct method for decoding cyclic codes

Let us introduce the notation $\mu^{i1} = \mu_1, \mu^{i2} = \mu_2, \dots, \mu^{it} = \mu_t$ and construct a polynomial $I(x) = x^t + \sigma_1 x^{t-1} + \cdots + \sigma_t$whose roots are the quantities $\mu_u, u = 1,2, \dots, t$.

We will assume that the degree of this polynomial is t, even if the true number of errors is $v < t$, i.e. let 's put $\varepsilon_{iu} = \mu_u = 0$at $v < u \le t$. The coefficients of $\sigma_1, \sigma_2, \dots \sigma_t$the polynomial $l(x)$are determined by the relation.

$$x^t + \sigma_1 x^{t-1} + \cdots + \sigma_1 = \prod_{u-1}^{t} x - \mu_u$$

Substituting into this relation $\mu_u$instead of x, multiplying the left and right sides of the resulting expressions by $\varepsilon_{iu}$and summing over u, we obtain

$$\sum_{u=1}^{t} \varepsilon_{iu}\, \mu_u^{(1+p)} + \sigma_1 \sum_{u-1}^{t} \varepsilon_{iu}\, \mu_u^{(1+p-1)} + \cdots + \sigma_1 \sum_{u=1}^{t} \varepsilon_{iu}\, \mu_{iu}^{p}$$

Introducing the notation

$\sum_{u=1}^{t} \varepsilon_{iu}\, \mu_u^j = S_j$ (3.1)

We get a system of equations

$S_{t+p} = \sigma_1 S_{1+p-1} + \cdots + \sigma_1 S_p,\ p = 0,1,2 \dots,$ (3.2)

which must be fulfilled for all values of p. If now it $j$is equal to one of the numbers $\{l_1, l_2, \dots\ l_r\}$, then from expressions (3.0) and (3.1) we get

$S_j = s_l\ \ j \in \{l_1, l_2, \dots\ l_r\}$.

This circumstance can be used to determine those coefficients of system (3.2) for which $j \in \{l_1, l_2, \dots\ l_r\}$.

Let us assume that in this way it was possible to determine all the coefficients of $v$linearly independent equations from the system (3.2). Then, by solving the system of these equations, we can find the quantities $\sigma_1, \sigma_2, \dots\ \sigma_v$. Now the position of

erroneous characters (ie numbers $i_1, i_2, \dots i_v$) will be determined by substituting all the elements $\mu^2, i = 0,1, \dots, n-1$one by one into the polynomial $I(x)$.

If, when setting the element , the $\mu_u = \mu^{iu}$equality is fulfilled $I(\mu_u) = 0$, then the corresponding component of the ε_{iu}error vector is different from zero. This follows from the fact that, by construction, the polynomial $l(x)$vanishes for all $\mu_u \in \delta\{\mu_1, \mu_2, \dots \mu_v\}$and is nonzero for all the others μ_u. Therefore, in this way, the position numbers of the vector that contain non-zero components can be found . eTo find the components themselves, $\varepsilon_{i1}, \varepsilon_{i2}, \dots \varepsilon_{iu}$it suffices to solve the system of linear equations (3.0), in which $\varepsilon_i, i = i_1, i_2, \dots i_v$, are set equal to zero.

In the case of decoding binary codes, there is no need to determine the values of the erroneous components, since all non-zero components of the error vector are equal to one. The problem of finding the error vector with the help of elements $S_j = b(\theta_j), j = 1,2, \dots r$,, will be completely solved if we establish the possibility of a unique solution to systems (3.0) and (3.2).

Consider the decoding of Bose-Chowdhury-Hokvinhem codes. Recall that cyclic BCH codes that correct independent errors whose multiplicity does not exceed t are specified using 2t nonzero elements

$$\mu^{m_0}, \mu^{m_0+1}, \dots, \mu^{m_0+2t-1}$$

Which by definition are the roots of all code words. As before, $\mu \in k$and has order n. The code generator is a polynomial.

$$g(x) = HOK[f_0(x), f_1(x), \dots, f_{2t-1}(x)].$$

Where $f_i(x), i = 1,2, \dots 2t-1$, are irreducible over $GF(q)$polynomials whose roots are μ^{m_0+i}. In this case (3.0) becomes

$s_j = \sum_{i=0}^{n-1} \varepsilon_i (\mu^j)^i, m_0 \le j \le m_0 + 2t - 1$.(3.0')

And $S_j$can be calculated using appropriate linear filters. Thus, equations (3.0') determine $2t$the coefficients in $t$the equations of system (3.2) for $p = m_0, m_0+1, \dots, m_0+t-1$. System (3.2) is a linear inhomogeneous system with a matrix of coefficients.

$$M_t = \begin{bmatrix} S_{m_0+t-1} & S_{m_0+t-2} & \dots & S_{m_0} \\ S_{m_0+t} & S_{m_0+t-1} & \dots & S_{m_0+1} \\ \dots & \dots & \dots & \dots \\ S_{m_0+2t} & S_{m_0+2t-1} & \dots & S_{m_0+t-1} \end{bmatrix}$$

Where $S_j = s_j = b(\mu^j), j = m_0, m_0+1, \dots m_0+t-1$. The conditions under which this system is solvable with respect $\sigma_1, \sigma_2, \dots \sigma_t$to are given by the following theorem.

Theorem 1 . The matrix determinant is $M_t$non-zero if the values are S_j[см. формулу (3.1)]formed from exactly $t$different non-zero pairs $\varepsilon_{iu}, \mu_u$and u is equal to zero if S_j, are formed less than from $t$non-zero pairs $(\varepsilon_{iu}, \mu_u)$.

Since the actual number of errors by assumption is equal to v, then the largest order of the matrix (3.3), for which its determinant is still nonzero, is equal to v. This means that in system (3.2) there are only $v$linearly independent equations obtained for $p = m_0, m_0+1, \dots m_0+v-1$, which can be solved with respect to $\sigma_1, \sigma_2, \dots \sigma_v$. This, in turn, implies that all non-zero roots of the polynomial can be found $l(x)$, i.e. the positions of all components of the error vector are determined. The values of the non-zero error vector can be found from the first $v$equations of the system (3.0'). The existence of a unique solution simply follows from the fact that the limiter of the coefficient matrix is the Vandermonde determinant. This determinant is different from zero, since $\mu^{iu}, u = 1,2, \dots v$, are different values.

So, the direct decoding algorithm can be formulated as follows:

1. Calculate the values of the quantities $S_j = b(\mu^j), j = m_0, m_0+1, \dots m_0+2t-1$. This operation can be carried out using linear multicycle filters.
2. Determine the largest number v of linearly independent equations in the system (3.2), looking for the largest v for which the determinant $|M_v|$is still different from zero. This number is equal to the number of errors that occurred.
3. Solve system (3.2) relatively $\sigma_1, \sigma_2, \dots \sigma_v$assuming that $\sigma_{v+1} = \dots = \sigma_t = 0$.
4. Alternately substitute the values $\mu^i, i = 0,1, \dots n-1$, into the equation $x^v + \sigma_1 x^{v+1} + \dots + \sigma_v = 0$, looking for those that turn it into an identity. If μ^uthere is one of these values, then in the received sequence of characters at the position $u$is erroneous.
5. The found values $\mu_u, u = 1,2, \dots v$, put into the equation (3.0') and solve them with respect to ε_{iu}. This operation is not needed in case of decoding binary codes.
6. Make corrections of errors by scaling $b(x)u - c(x) = -\sum_{u=1}^{i} \varepsilon_{i_u} x^{i_u}$

Some control can be obtained by restricting to decoding only binary codes and $m_0 = 1$. In this case, all are equal to one and

$$S_j = \sum_{u=1}^{v} (\mu^i)^{iu}, j = 1,2, \dots ,2t, v \le t.$$

Therefore, $S_j$are the power-law noises of the quantities v^{iu}.

In a polynomial $l(x)$whose roots are the $\mu^{iu}, u = 1,2, \dots , v$coefficients $\sigma_u$associated with c S_jNewton's formulas. Considering the coefficients in Newton's formulas as $a_u$variables, we can compose a system of equations

$$N = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_v \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{2v-1} \end{bmatrix} \text{(3.4)}$$

Where

$$N = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ S_2 & S_1 & 1 & \dots & 0 \\ S_4 & S_3 & S_2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ S_{2v-2} & S_{2v-3} & S_{2v-4} & \dots & S_{v-1} \end{bmatrix} \text{(3.5)}$$

The conditions under which the system of equations (3.4) is solvable are given by the following theorem.

Theorem 2 . The determinant of a matrix $N_v$is non-zero if the quantities $S_j, j = 1,2, \dots , 2t$, are formed by power sums $v$or $v - 1$different elements n^uu is equal to zero if the quantities are $S_j$formed by power sums of fewer than $v - 1$elements.

The direct algorithm for decoding binary BCH codes almost does not differ from the general algorithm given above, although the practical implementation of the decoding method is simplified due to the simpler calculation of the determinant of the matrix (3.5) compared to the determinant of the matrix (3.3). In addition, when finding the largest number $v$for which the determinant $N_v$is still different from zero, one can use the solvability of the system of equation (3.4), even if the desired number v is equal to or one more than the actual number of errors that have occurred. This allows us to construct a calculation scheme as follows. First, it is assumed that $v = 2$, then $v = 4$, etc., are increased by two units. For each value , the determinant is calculated $N_v$and choose for $v$the largest value at which $|N_v| \neq 0$. Thereafter, decoding is continued in the manner described. If it turns out that the found value of v exceeds the number of errors by one, then one of the solutions (3.4) is zero.

The complexity of the described direct decoding method depends to a large extent on how easy it is to calculate the determinant of the matrix $N_v$and solve the system of equations (3.4). One of the possible methods for calculating the determinant and solving the system of equations is such a linear transformation of the system, in which the matrix $N_v$takes a triangular shape. However, a more efficient transformation can be proposed, which is based on the special structure of the matrix $N_v$and on the properties of the elements of finite fields. Such a transformation was described by E. Berlekamp.

Let us write the system of equations (3.4) again:

Recall that S , are the power sums of the roots of the polynomial

$$l(x) = \prod_{u=1}^{t} (x - \mu^{i_u}), \text{где } i_1, i_2 \dots i_t$$

- numbers of erroneous characters.

Let us transform this system by introducing new variables R_u, which are related to the quantities by $S_j$the relation

$$R_u = \sum_i A_j S_{u-2j} .$$

Where $A_0 = S_0 = 1, A_{-1} = S_{-1} = 0$for everyone $i > 0$ и $A_j \in K, i > 0$. Let us now add to each ith equation of the system (3.4) $(i - 1)$- e equation multiplied by A_1, $(i - 1)$- e equation multiplied by A_2, etc. As a result, we obtain a new system of equations

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ R_2 & R_1 & 0 & \dots & 0 \\ R_4 & R_3 & R_2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ R_{2v-2} & R_{2v-3} & R_{2v-4} & \dots & R_{v-1} \end{bmatrix} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \vdots \\ \sigma_v \end{bmatrix} = \begin{bmatrix} R_1 \\ R_3 \\ R_5 \\ \vdots \\ R_{2v-1} \end{bmatrix} \qquad (3.6)$$

Let us choose the coefficients $A_j$in such a way that $R_{2i} = 0, i \neq 0$i.e.

$$0 = \sum_{j=0}^{i} A_j S_{2(i-j)}$$

From here $A = \sum_{j=0}^{i-1} A_j S_{2(i-j)}$and under the initial condition $\mathrm{A}_0 = 1$, all other A can be determined. It can be proved that $\mathrm{A}_{2i} = 0$.

Using the fact that $R_{2i} = 0$, we rewrite system (3.6) in the form of two systems of linear equations (for convenience, we assume that $v = 5$).

The first system consists of $[v/2]$($[x]$is the integer part of x; for $v = 5$we have $\left[\frac{5}{2}\right] = 2$the last equations in system (3.6):

$$\begin{bmatrix} R_5 & R_3 \\ R_7 & R_9 \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \sigma_4 \end{bmatrix} = \begin{bmatrix} R_7 \\ R_9 \end{bmatrix} \qquad (3.7a)$$

The second system consists of the remaining $v - [v/2]$equations, which can be transformed as follows:

$$\begin{bmatrix} \sigma_2 \\ \sigma_3 \\ \sigma_5 \end{bmatrix} = \begin{bmatrix} R_1 & 0 & 0 \\ R_3 & R_1 & 0 \\ R_5 & R_3 & R_1 \end{bmatrix} \begin{bmatrix} 1 \\ \sigma_2 \\ \sigma_4 \end{bmatrix} \qquad (3.7b)$$

Therefore, instead of solving the system of $v$equations (3.4), it $\sigma_1, \sigma_2, \dots \sigma_v$suffices to solve the system $[v/2]$of equations (3.7 a) and find the rest $\sigma_i$using equality (3.7 b). The advantages of such a transformation can be explained by the following rough argument.

The number of operations required to perform the transformation and solve equation (3.7 b) is proportional to v^2. At the same time, the number of operations required to solve the system of equations (3.4) or (3.7 a) is proportional to the cube of the number of equations. By halving the number of equations, we have reduced the number of operations by about 8 times, which is quite large v. The external memory required to perform the transformation and solve equations (3.4) and (3.7 a) depends on the square of the number of equations.

Therefore, the introduction of a transformation reduces the external memory by about 4 times.

2.4 Incremental decading of cyclic codes

Let us consider a simple modification of the direct decoding method, which leads to a significant reduction in the decoding algorithm. Note that attempts to simplify decoding in some particular cases were found in the works of R. Banerji, E.L. , not yet fully disclosed by the possibilities of the approach described above. It is as the implementation of one of these possibilities that one should consider phased decadence, which was described in a general form by R.T. Jiang.

The considered decoding method directly follows from the property of the matrix M_v и N_v. It is most convenient to start with a matrix M_v. Its properties are formulated in Theorem 1. It allows detecting a change in the number of errors in the received sequence and, consequently, performing step-by-step decoding at $t \le (d_0 - 1)/2$. Indeed, if $v$there is the largest number for which $|M_v| \neq 0$, then by successively changing the symbols of the received sequence $b = (\beta_1, \beta_1, \dots \beta_{n-1})$and calculating the matrix determinant.

$$M'_v = \begin{bmatrix} S'_{m_0+v-1} & S'_{m_0+v-2} & \dots & S'_{m_0} \\ S'_{m_0+v} & S'_{m_0+v-1} & \dots & S'_{m_0+1} \\ \dots & \dots & \dots & \dots \\ S'_{m_0+2v-2} & S'_{m_0+v-3} & \dots & S'_{m_0+v-1} \end{bmatrix}$$

Where

$S'_j = b'(\mu^j). J = m_0, m_0 + 1, \dots, m_0 + 2t - 1, v \le t, b'(x) = b(x) - \varepsilon_i x^i,$

you can set the moment when changing some character of the sequence $b$leads to the correction of the error. Since the values $S'_j$when correcting the error turn out to be formed from $v - 1$non-zero pairs $(\varepsilon_{iu}, \mu^{iu})$, the determinant of the matrix M'_vvanishes. A value of zero $|M'_v|$indicates that the error has been fixed.

Let us assume that cyclic shifts of the received sequence are performed and after each i-th shift the values $S_j^i = b^{(i)}(\mu^j), j = m_0, m_0 + 1, \dots, m_0 + 2t - 1$,where are calculated $b^{(i)}(x) \equiv x^{-i} b(x) mod\ x^n - 1, i = 0,1, \dots, n - 1$. The symbol at the zero position of the vector $b^{(i)} = (\beta_i, \beta_{i+1}, \dots, \beta_0, \dots, \beta_{i-1})$is erroneous and the error value is ε, if the determinant of the matrix.

$$M_v^i = \begin{bmatrix} S^{(i)}_{m_0+v-1} - \varepsilon & S^{(i)}_{m_0+v-2} - \varepsilon & \dots & S^{(i)}_{m_0} - \varepsilon \\ S^{(i)}_{m_0+v} - \varepsilon & S^{(i)}_{m_0+v-1} - \varepsilon & \dots & S^{(i)}_{m_0+1} - \varepsilon \\ \dots & \dots & \dots & \dots \\ S^{(i)}_{m_0+2v-2} - \varepsilon & S^{(i)}_{m_0+2v-3} - \varepsilon & \dots & S^{(i)}_{m_0+v-1} - \varepsilon \end{bmatrix}$$

Equal to zero. Ratio

$\left|\mathrm{M}_{v}^{(i)}=0\right|$ (3.8)

can be considered as a control condition, the fulfillment of which indicates the need for correction of the corresponding character.

When decoding binary codes, the control condition can be simplified by passing to a matrix $N_v$whose determinant is easier to calculate. The property of the matrix N_v, formulated in Theorem 2, also admits stepwise decoding. Let $v$there be the largest number for which the determinant of the matrix $N_v$is still different from zero. Then the number of errors is exactly $v-1$. Indeed, according to Theorem 2 for , the $|N_v|\neq 0$number of errors is $v$or $v-1$. But the number of errors cannot be equal v, because in this case $|N_{v+1}|\neq 0$, which would contradict the assumption of maximality v. Therefore, it is clear that the matrix determinant $N_v$becomes equal to zero if a change in some symbol in the received sequence leads to an error correction. This fact can serve as a starting point for obtaining a control condition similar to (3.8) in the case of binary codes.

Let, as before, cyclic shifts of the received sequence are performed and $b$after $i$each i-th shift the values are calculated $S_j^i=b^{(i)}(\mu^j), j=1,2,\dots,2t$, where$b^{(i)}(x)\equiv x^{-i}b(x) mod\ x^n-1 t=0,1,\dots,n-1$

$$N_v^i=\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ S_2^{(i)}+1 & S_1^{(i)}+1 & 1 & \dots & 0 \\ S_4^{(i)}+1 & S_3^{(i)}+1 & S_2^{(i)}+1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ S_{2v-2}^{(i)}+1 & S_{2v-3}^{(i)}+1 & \dots & \dots & S_{v-t}^{(i)}+1 \end{bmatrix}$$

Will be equal to zero. Ratio

$\left|N_v^{(i)}\right|=0$ (3.9)

It can again be considered as a control condition, the fulfillment of which indicates the need to invert the i- ith symbol in the received sequence.

Let us now formulate an algorithm for step-by-step decoding, limiting ourselves only to binary codes and test relations in the form (3.9). The general case will differ in minor details, which the reader can recover for himself. For decoding, you must perform the following operations:

1. Calculate the value of the quantities $S_j^0=b^0(\mu^j) j=1,2,\dots 2t$. Index (0) indicates that the sequence $b$has not yet undergone cyclic shifts.
2. Find the largest value $v$for which the determinant of the matrix $N_v^0$is still different from zero. Then the actual number of errors in the sequence is $b=v-1$.
3. Perform cyclic shifts, each time calculating the values S_j^i. If the first K symbols of code words are informational, then $i=0,1,\dots,k-1$. Otherwise $i=0,1,\dots,n-1$. For each i, which varies within the limits indicated above, calculate the determinant $\left|N_v^{(i)}\right|$. Form a noise sequence eby writing at the position with the number $i$of the noise sequence one if $\left|N_v^{(i)}\right|=0$and zero if $\left|N_v^{(i)}\right|\neq 0$.
4. Restore the given sequence by adding b и echaracter by character.

Thus, when switching to step-by-step decoding, we managed to get rid of such time-consuming operations as solving a system of equations over a finite field and finding the roots of the polynomial $l(x)$. Now the complexity of decoding is determined only by the complexity of calculating the determinant of the size matrix $v*v$, whose elements are the values of the final field K.

Example.

Let us illustrate the above algorithm with an example of decoding with a binary BCH code with binary independent error correction. Note that step-by-step decoding provides the simplest implementation of decoding for k cycles of binary error-correcting codes compared to all other algorithms already described. This simplification is possible due to the simplicity of the single control condition.

Consider two situations

1. Two errors occurred during transmission. In this case, the largest v for which the determinant of the matrix n is still non-zero is v =3. Therefore, omitting the index, but bearing in mind that the values change at each cycle, we get

$$N_3^i=\begin{bmatrix} 1 & 0 & 0 \\ S_2+1 & S_2+1 & 1 \\ S_4+1 & S_3+1 & S_2+1 \end{bmatrix}$$

Calculating the determinant $N_3^i$and equating it to zero, we obtain the control condition, $S_1 + S_1^2 + S_1^3 + S_3 = 0$ (3.10)

which must be executed whenever one of the two errors occupies the zero position during the shifts of the received message.

2. A single error occurred during transmission. Then the largest $v = 2$.

Revealing the Determinant of a Matrix $N_2^{(i)}$and equating it to zero, we obtain the second control condition

$S_1 = 1$ (3.11)

which must be executed when the erroneous symbol occupies the zero position during shifts. Since in the case of single errors $S_i = S_1^j$, it can be easily seen that the fulfillment of condition (3.11) also entails the fulfillment of (3.10). Therefore, to make a decision about correcting a certain symbol, it suffices to check the fulfillment of only one condition (3.10) for both single and double errors.

The absence of transmission errors must be checked in a special way. Such a check can be based on the fact that with error-free transmission $S_j = 0$for all $j = 1,2, \dots 2t$.

Such decoding itself is simpler than direct decoding. However, it allows even greater simplifications of the decoding device by eliminating the operation of multiplying field elements $k$and calculating the sum $S_1 + S_1^2$on a single shift register with feedback.

Let us consider the ways of obtaining each of the terms of the control condition (3.10), remembering that the $m_0 = 1$code is also given by the elements μ, μ^2, μ^3, μ^4, which, by definition, must be the roots of all code words.

1 . Calculation $S_1 + S_1^2$. Equations (3.0) and (3.1) imply that $S_1 = b(\mu)$, where b (x) is a polynomial corresponding to the accepted sequence $b = (\beta_0, \beta_1, \dots, \beta_{n-1})$. Let $g(x)$there be that irreducible over $GF(2)$factor of the generator polynomial $g(x)$whose root is μ. It was noted that the μ-element of the field K, which has the order n. We assume everywhere that μdoes not belong to any smaller subfield in K and, therefore, the degree of the polynomial $g_1(x)$is equal to m [m is the degree of the field $k$over $GF(2)$]

Each field element Kcan be represented as a linear combination of basic elements. As such basic elements can be chosen $1, \mu, \mu^2, \dots, \mu^{m-1}$. Then, introducing the notation for representing the element $r \in K$in the basis $1, \mu, \mu^2, \dots, \mu^{m-1}$, we can write

$$r(\mu) = r_0 + r_1\mu + \cdots + r_{m-1}\mu^{m-1}$$

Where $r_i \in GF(2), i = 0,1, .. m-1$.

On the other hand, $r(\mu)$is the element of the field $k$that is obtained by substituting into the polynomial $r(x)$instead $x$of the element μ. Hence the conclusion follows: in order to find the element of the field, which is obtained by setting instead in an arbitrary polynomial $f(x)$over the field $GF(2)$, it is necessary to find the remainder $r(x)$after dividing $f(x)$by $g_1(x)$and write $f(\mu)$it as a vector, the components of which are the coefficients of the polynomial $r(x)$. This also applies to the case when the main field contains elements.

Thus, to find S_1it is necessary to determine the remainder of the division $b(x)$by $g_1(x)$. The remainder can be computed using the multi-cycle linear filters described in Section 3. In offline mode, a linear filter with an initial state corresponding to the polynomial $r(x) mod\ g_1(x)$ will produce remainders from dividing the polynomials $b^i(x)\ mod\ x^n - 1, i = 0,1, \dots, n-1$, by the polynomial $g_1(x)$, i.e. Will comb out S_1^i. To find it $S_1^i$is enough to calculate the remainder of the division $[r(x)]^2 = r(x^2)$ на $g_1(x)$.

This operation could also be done with a linear filter. However, it is possible to construct a code filter that directly computes $r(x) + r(x^2)\ mod\ g_1(x)$, and hence also $S_1 + S_1^2$. In offline mode, the sequence of states of such a filter will correspond to $S_1^{(i)} + (S_1^{(i)})^2, i = 0,1, \dots, n-1$.

2. Calculation S_3. By definition $S_1^3 = b(\mu)$, therefore, it follows from the above that to find it $S_3$is enough to find the remainder after dividing the polynomial $b(x)^3$by the polynomial $g_1(x)$. For this, in turn, it is sufficient to have a code filter, the transition matrix of which is given by the generating element $c(x) \equiv x^{-3}\ mod\ g_1(x)$. It is easy to verify that such a filter will produce $S_3^i\ i = 0,1, \dots n-1$..

3. Calculation S_1^3. Since $S_3^i = [r(\mu)]^3$, where $r(x) \equiv b(x)\ mod\ g_1(x)$, then S_1^3it could be found using a special combinatorial scheme, calculating the representation coefficients $[r(\mu)]^3$in the basis $\{1, \mu, \dots, \mu^{m-1}\}$through the corresponding coefficients $r(\mu)$.

The combinatorial circuit is essentially an arithmetic device that performs addition and multiplication of elements of the Galois field in one cycle. In our case, the arithmetic device designed to calculate S_1^3, can be greatly simplified if we use the recurrence relation introduced below.

It was said above that S_1 can be found using the remainder calculator $b(x)\ mod\ g_1(x)$, where $b(x)$- corresponds to the input filter sequence. Then the subsequent state of $B_{i+l}(x)$such a filter can be determined in terms of the previous state $B_i(x)$using the comparison

$$B_{i+1}(x) \equiv [B_i(x) + B_{i+1}(x)]^{-1} mod\ g_1(x) \qquad (3.13)$$

Where $\beta_{i+1}$is the value of the output signal at the moment $i+1$. From this it follows that

$$B^3_{i+1}(x) \equiv [B^3_i(x) + B_{i+1}[B^2_i(x) + B_i(x) + 1]]x^{-3} mod\ g_1(x)\ ] \qquad (3.14)$$

Comparison (3.144) is a recursive relation that allows one to calculate $B^3_{i+1}(x)\ mod\ g_1(x)$ through $B^3_i(x), B^2_i(x) + B_i(x) mod\ g_1(x)$ and the current value of the input symbol β_{i+1}. Now noticing that

$$B_{n-1}(x) \equiv b(x) mod\ g_1(x)$$
$$B_{n-1+i}(x) \equiv b(x)x^{i-1} \equiv b(x)x^{i-1} mod\ g_1(x).$$

Moreover, the latter takes place, since $\beta_{n-1+i} = 0$for all $i > 0$, we obtain $B_{n-1+i}(\mu) = {S_1}^{(i)}$, and therefore

$$B_{n-1+i}(\mu) + [B_{n-1+i}(x)]^2 = S_1^{(i)} + (S_1^{(i)})^2. \qquad (3.15)$$

i.e. $B^3_{n-1+i}(\mu) = [S_1^{(i)}]^3$_

From (3.14) and (3.15) follows the recursive relation for $[S_1^{(i+1)}]^3$:

$$[S_1^{(i+)}]^3 = [S_1^{(i)}]^3\mu^{-3}$$

The operation of multiplication by $\mu^{-3}$is performed by the code filter. Thus, for decoding, it is necessary to have two code filters that calculate $S_1^{(i)}$ и $S_1^{(i)} + [S_1^{(i)}]^2$. The calculation $[S_1^{(i)}]^3$is carried out in two stages. The first stage, as a result of which the value is found $[B_{n-1}(\mu)]^3 = S_1^3$, ends after receiving the message $b = (\beta_1, \beta_2, \dots, \beta_{n-1})$in full. At this stage, the recursive relation (3.14) is used. At the second stage $[S_1^{(i)}]^3$, is calculated using relation (3.16). To perform the operation of multiplication by $x^{-3}\ mod\ g_1(x)$, or, what is the same, by μ^{-3}, a special code filter is used.

After receiving the message b, the states of these three filters will correspond to the values $S_1 + S_1^2, S_3$ и S_1^3 . Starting from this moment, the decoding device switches to autonomous mode and, using the control condition (3.10), generates a noise sequence.

On fig. 9 shows a decoder for a binary BCH code (15, 7) that corrects double independent errors. In this example, the polynomial , which belongs to the exponent 15, is $GF(2)$selected as a generator polynomial , with non-reducible over $g(x) = g_1(x), g_2(x)$factors and $g_1(x) = x^4 + x + 1$. One can check that the first of these polynomials is primitive, and the root of the second is the cube of the root of the first polynomial.

The filter that calculates $S_1 + S_1^2$can be constructed in accordance with the results of paragraph 3. The transition matrix $M$and and $F$the linear transformation matrix of the outputs are, respectively.

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \text{ и } F = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

The transition matrix of filters that calculate $S_3$and $S_1^3$is

The complexity of the decoding device, designed to correct single and double errors, with increasing code length increases in proportion to the binary logarithm n. This is explained by the fact that the block diagram of the decoding device does not change with increasing code length (the control condition is preserved), but the number of memory elements in the code filters changes.

If the number of errors that can be corrected by a given code exceeds two, then the structure of the decoder becomes more complicated due to the need to use several control conditions. Consider a BCH code that corrects triple errors. In the presence of three errors, this condition, in accordance with (3.8) and (3.9), will be

$$\left|N_4^{(i)}\right| = \begin{bmatrix} 0 & 1 & 0 & 0 \\ S_2 + 1 & S_2 + 1 & 1 & 0 \\ S_4 + 1 & S_3 + 1 & S_2 + 1 & S_1 + 1 \\ S_3 + 1 & S_5 + 1 & S_4 + 1 & S_3 + 1 \end{bmatrix}$$

$$= S_1^3(1 + S_1 + S_1^3 + S_3(1 + S_1 + S_1^2 + S_1^3 + S_3) + S_5(1 + S_1) = 0) \qquad (3.17)$$

For two or one errors, the control condition coincides with condition (3.10). However, a simplification is possible here, which follows from the fact that the determinant (3.17) remains non-zero even if the actual number of errors is equal to two. But not one of the errors did not take the zero position as a result of the shifts. Therefore, condition (3.17) can serve as a control for correcting double and triple errors. Only when the number of errors in the codeword is equal to one should the second condition be checked.

$S_1 = 1$ (3.18)

The transition from one condition to another should be carried out on the basis of knowledge of the actual number of errors that have occurred. To do this, you can use the following approach. It is easy to verify that, as $S_1 = 1$, equality (3.17) holds. Therefore, with single errors, condition (3.17) will be satisfied at each cycle and the appearance of more than three corrective signals will be sufficient reason to go to condition (3.18).

The stepwise decoding algorithm was derived from the assumption that only BCH codes are considered. However, a similar decoding procedure can be found for other codes from the cyclic class. Such codes are, for example, Melas and Megpit codes. These codes correct double independent errors and are given by the elementsμ, μ^{-1}поля $GF(2^m)$.

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

Before receiving a message, the position of the key corresponds to $D = 0$. This opens up free access to information in the buffer register and in all code filters.

Of the features of the filters, only the originality of the change of states in the filter should be noted S_1^3. Each state is set in two half cycles. From the first half cycle, the contents of all memory cells are rewritten to the adders C, while the filter memory cells are S_1^3brought to the zero state at the same time. During the second half cycle, the summation result is rewritten into this filter and a shift is made. Shifting multiplies by μ^{-3}.

After receiving the message completely, the keys are set to the position $D = 1$. The checksum value (3.10) appears at the outputs of the adders at each cycle. CIf the checksum is zero, then the character currently leaving the buffer register must be corrected. To disable the correction circuits in the case when the transmission is carried out without errors [in this case, condition (3.10) is also satisfied], the signal is used $E = 1$. The specified signal is formed after receiving the message in full and corresponds to $S_1 = 0$. It is clear that if only single or double errors are allowed, $S_1 = 0$it can only be possible with error-free transmission.

2.5 Catching Errors in Cyclic Codes

In this section, we consider a method for detecting and correcting errors in cyclic codes using shift registers.

Consider a cyclic (n, k)code with a generating polynomial $g(x)$. Suppose that the code polynomial $c(x)$acquired an error during transmission over the communication channel $e(x)$, so that we will receive a message in the form of a polynomial $v(x)$. We know that the syndromic polynomial $s(x)$is the remainder of division $v(x)$by $g(x)$. It is clear that if the acquisition of the error is in the first m positions (here , the $m$degree $g(x)$of), then this sum is equal to the error polynomial $e(x)$. But in general this is not the case. Actually $e(x) = a(x)\,(g(x) + s(x))$.

Let us assume that the errors are concentrated in the last m positions, where $m$is the degree of $g(x)$, i.e. $e(x) = e_k\, x^{\,k} + e_{k+1}\, x^{\,k+1} + \ldots + e_{n=1}\, x^{\,n-1}$. If $v(x)$cyclically shifted by m steps, then the errors will go to the first m positions of $x^0, x^1, \ldots, x^{m-1}$the polynomial $v^{(m)}(x)$($v^{(m)}(x)$means a cyclic shift by $m$positions of the vector of coefficients of the polynomial $v(x)$. The corresponding error vector for $\overline{v}^{\,(m)}$has the form

$$\overline{v}^{\,(m)} = (e_k, e_{k+1}, \ldots, e_{n-1}, 0, \ldots, 0)$$

now the syndromic polynomial $s^{(m)}(x)$for $v^{(m)}(x)$is equal to the remainder of division $v^{(m)}(x)$by $g(x)$, i.e. equals$s^{(m)}(x) = e_k + e_{k+1}\, x + \ldots + e_{n-1}\, x^{m-1}$

Multiplying $s^{(m)}(x)$by x^k, we will have

$$x^k s^{(m)}(x) = e(x) = e_k x^k + \ldots + e_{n-1}\, x^{n-1}$$

Therefore, it is said that if the errors are concentrated on the last $m$positions, then the error vector coincides with $x^k s^{(m)}(x)\ (mod(x^n - 1))$, here $s^{(m)}(x)$there is a syndromic polynomial for $v^{(m)}(x)$. In this case, we calculate $s^{(m)}(x)$and then $e(x) = v(x) + x^k s^{(m)}(x)\ (mod\ (x^n - 1))$.

Suppose that the errors are contained in the section of length m, then not necessarily in the last places, for example, starting from $x^i$the place to x^{m+i-1}. If we $v(x)$shift to the right by $n - i$positions, then the errors will be concentrated in the conditions of the previous case, and therefore the error polynomial can be identified with c $x^i s^{(n-i)}(x)$. But $i$we don't know the meaning. Therefore, we do a right shift operation until we catch (i.e. until we find the value $i$and the corresponding error vector). This procedure is called error trapping.

Let's assume that the code corrects $t$errors and less. To determine the event that errors are caught in the syndrome register, we can simply test the syndrome weight after each syndromic register shift. As soon as the weight in the syndromic register becomes

less than $t$or equal $t$to , we consider that the errors are caught in the syndromic mode. So, if the number of errors in $v(x)$does not exceed $t$and if they are concentrated in $m$consecutive positions, then they are caught in the syndromic register.

If during successive testing of shifts we get the weight of the syndrome $\leq t$, then we find the places of errors in the shifted message, which means that by giving the reverse shift, we correct the errors in the message.

If, for all successive shifts, we never get weights in the syndrome message register $\leq t$, then this means that either there are more than $t$errors, or they do not form a packet with length $\leq t$.

3. CONCLUSION AND SOLUTIONS

In today's Internet environment, there are many potential dangers and risks. Therefore, data encryption is necessary to protect user information well. Almost all applications support the full range of encryption methods. However, many users still use the normal (non-encrypted) protocol, which is easy to leak information. Hopefully through this article you will understand more about encryption and encryption methods. For businesses, data encryption should only be a backup solution when data is stolen. The more important thing to do is to manage network security in the enterprise. This prevents hackers from getting data, especially sensitive data, from businesses. If the hacker cannot access the data, they have no chance to decrypt it. Then businesses only need to encrypt the sent information or important information. The linear decoding methods described above are based on knowledge of error syndromes with numbers not exceeding the capacity of the selected linear code. These methods are completely possible with modern computers.

REFERENCES

1) Shu Lin, Daniel J. Costello, Jr., Error control coding. fundamentals and applications. - Prentice-Hall Inc. Englewood Cliffs, New Jersey. - 1983. 603 pp.
2) Berezyuk N. T., Andrushchenko A. G., Moschitsky S. S. et al., Coding information (binary codes. - Kharkov, publishing association "Vishcha school". - 1978. - 252 p.
3) Blahut R. Theory and practice of error control codes, Translated from English: I.I. Grushko, V.M. Blinovsky. Edited by: K.Sh. Zigangirov. — M.: Mir. - 1986. - 576 p.
4) R. Morelos-Zaragoza, The art of error-correcting coding. Methods, algorithms, application. - Moscow: Technosphere. - 2005. - 320s.