**Basic principles of block coding**

**Redundancy and its role**

**Redundancy** is the addition of additional symbols (bits) to the original data, which at first glance may seem "useless". After all, from the point of view of information compression, we want to minimize the amount of transmitted data. However, in the conditions of data transmission over a noisy channel with possible errors, this redundancy becomes the most important element for ensuring reliability.

**Why are redundant characters added?**

* **Error detection** : Redundant bits allow the receiver to check whether there was an error in the transmission of data. Using special decoding rules, it is possible to determine whether the received code word has been corrupted.
* **Error correction** : In some schemes, such as the Hamming code, redundancy not only allows the error to be detected, but also corrected. This capability makes systems more reliable, since errors caused by noise in the communication channel can be automatically corrected.

Thus, adding redundant symbols increases the amount of transmitted data, but at the same time increases the reliability of transmission, which is critical for digital communications.

When we transmit data over a communication channel such as the Internet or wireless communication, the problem of errors inevitably arises. These errors can occur in the form of corrupted bits, which changes the meaning of the transmitted data. Error-Correcting Codes (ECC) use redundancy to counteract these errors.

**Error correction mechanism:**

* Redundant bits are added so that when corrupted data is received, it is possible to reconstruct the original message.
* At the receiver side, using decoding rules, it is possible to calculate which bits were corrupted and hence reconstruct the original message without the need for retransmission.

**Example: Hamming code (7,4) (see previous lectures)**

Let us consider the process of encoding and error correction using the example of the Hamming code with parameters (7,4), which encodes 4 information bits into a 7-bit code word, adding 3 control (redundant) bits.

**1. Example of coding:** Let us have the original message X=1011 (4 information bits).

* Step 1: Assign bits d1,d2,d3,d4 as message information bits.
* Step 2: Calculate 3 control bits p1,p2,p3​ based on the parity of the combinations of information bits:
  + p1​ checks bits d1,d2,d4​
  + p2​ checks bits d1,d3,d4​
  + p3​ checks bits d2,d3,d4​
* Let the information bits be X=1011. We calculate the control bits:
  + p 1= d 1 ⊕ d 2 ⊕ d 4=1 ⊕ 0 ⊕ 1=0
  + p 2= d 1 ⊕ d 3 ⊕ d 4=1 ⊕ 1 ⊕ 1=1
  + p3 = d2⊕d3⊕d4 = 0​​​​​

Now the code word is: p1,p2,d1,p3,d2,d3,d4 ​, i.e. 0110101.

**2. Example of error detection and correction:** Let us assume that an error occurred during data transmission, and instead of the code word 0110101, 0010101 was received, where the second bit was transmitted erroneously (instead of 1, 0 was transmitted).

* Step 1: The receiver checks the check bits with the same combinations:
  + We check p1=d1 ⊕ d2 ⊕ d4=1 ⊕ 0 ⊕ 1=0 (no error).
  + We check p2=d1 ⊕ d3 ⊕ d4=1 (there is no error).
  + We check p3=d2 ⊕ d3 ⊕ d4=0 (there is no error).
* Step 2: Based on these checks, it can be found that the error occurred in the second bit and restored to the correct value.

Thus, the Hamming code allows not only to detect a single error, but also to correct it. For students, this example will be a clear confirmation of how adding just a few control bits can significantly increase the reliability of data transmission.

**Fundamentals of error-correcting coding**

The task of the source coder is *to present the data to be transmitted in the most compact and, if possible, undistorted form* .

When transmitting information over a communication channel with interference, errors may occur in the received data. If such errors are small or occur rarely enough, the information can be used by the consumer. If there are a large number of errors, the information received cannot be used.

To reduce the number of errors that occur when transmitting information over a noisy channel, *channel coding* , or *error-correcting coding , can be used* .

The possibility of using coding to reduce the number of errors in a channel was theoretically demonstrated by K. Shannon in 1948 in his work "Mathematical Theory of Communication". It was stated that *if the rate at which a source creates messages (source throughput) does not exceed a certain value, called the channel capacity, then with appropriate coding and decoding it is possible to reduce the probability of errors in the channel to zero* .

It soon became clear, however, that the actual limits on transmission speed are set not by the channel capacity, but by the complexity of the coding and decoding schemes. Therefore, the efforts of developers and researchers in recent decades have been directed at finding effective codes, creating practically feasible coding and decoding schemes that would approach those predicted theoretically in their characteristics.

Error-correcting coding is *a message processing technique designed to improve the reliability of transmission over digital channels.* Although the various coding schemes are very different from each other and are based on different mathematical theories, they all have two common properties.

**The first** is *the use of redundancy* . Coded sequences always contain additional, or redundant, symbols. *The number of symbols in a code sequence* ***Y*** *always larger than necessary to uniquely represent any message* ***λ*** ***i*** *from the alphabet.*

**The second** is *the averaging property* , which means that *redundant symbols depend on several information symbols* , that is, the information contained in the code sequence ***X*** is also redistributed to redundant symbols.

There are two large classes of error-correcting codes: *block and convolutional.* The defining difference between these codes is the absence or presence of encoder memory.

**Introduction to Block Codes**

- **Definition of block code** : Each block of the original data of length *k* symbols is encoded into a codeword of length *n* , where redundant symbols are added for error correction. This is called an ( *n* , *k* ) -code.

- **Example** : If *k* =4, it means that we encode 4 bits of information into a 7-bit code word.

**The encoder for block codes** divides a continuous information sequence ***X*** into message blocks of length ***k*** symbols.

*The channel encoder* converts message blocks ***X*** into longer binary sequences ***Y*** consisting of ***n*** symbols and called *code words.* The symbols ( ***nk*** ) added to each message block by the encoder are called *redundant* . They do not carry any additional information, and *their function is to provide the ability to detect (or correct) errors* that occur during transmission.

***A k*** -bit binary word can represent ***2 k*** possible values from the source alphabet, which correspond to ***2 k*** code words at the encoder output.

*Such a set of* ***2 k*** *code words is called* ***a block code*** *.*

The term " *memoryless* " means that *each block of* ***n*** *symbols depends only on the corresponding information block of* ***k*** *symbols and does not depend on other blocks.*

**The encoder for convolutional codes** works with the information sequence without dividing it into independent blocks. At each moment of time, the encoder forms a block consisting of ***v code symbols (code block) from a small current block of information symbols of size b symbols*** (message block), where ***v > b.*** In this case, the code ***v -*** symbol block depends not only on the ***b-*** symbol block-message present at the input of the encoder at the current moment, but also on the preceding ***m*** message blocks. This is, in fact, what the presence of memory in the encoder consists of.

Block coding is convenient to use in cases where the source data is already grouped into some blocks or arrays by its nature.

When transmitting over radio channels, convolutional coding is more often used, which is better suited to bit-by-bit data transmission. In addition, with the same redundancy, convolutional codes, as a rule, have a better correcting ability.

**Classification of block codes**

1. **Linear block codes**

Linear block codes are a class of block codes that have linear properties that simplify their analysis and use in data transmission systems. They are based on the concepts of linear algebra, which allows the use of powerful methods for encoding and decoding.

**Definition of Linear Code**

**code is called a linear** code if any linear combination of its code words is also a code word. More formally, if *C* is a linear code of length *n* over *GF* (2) (the field of two elements 0 and 1), then for any two code words ***x*** , ***y*** ∈ *C* and any coefficients *a* , *b* ∈ *GF* (2), the linear combination *ax* + *by* also belongs to *C* . This means that the set of code words forms a vector space over GF(2).

**Linearity of Codes: Example**

Consider the (7,4) Hamming code. In the Hamming code, 4 information bits are converted into a 7-bit code word with the addition of three check bits. It is a linear code because if you add (modulo 2) any two code words, the result is a new code word.

**Example of code words** :

* The code words x1=0110101 and x2=1101100 are the code words of the Hamming code.
* Linear addition of these two code words modulo 2: x1 ⊕ x2 = (0110101) ⊕ (1101100) = 1011001

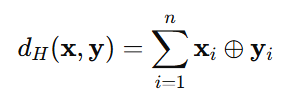
The result is also a code word, which confirms the linearity of the code.

**2. Codes with minimum Hamming distance**

The minimum Hamming distance is one of the key parameters of any error-correcting code. It determines how reliable the code will be in detecting and correcting errors.

**Determining the Hamming distance**

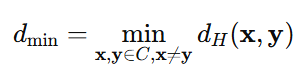
**The Hamming distance** between two codewords is the number of positions at which they differ. If two codewords *x* and *y* have the same length, then the Hamming distance between them, denoted as *dH* ( *x* , *y* ), is calculated as follows:



where ⊕ is addition modulo 2 (logical XOR) and *n* is the length of the codewords.

**Minimum code distance**

The minimum Hamming distance of a code *C* is the minimum Hamming distance between any two distinct codewords in the code *C* . It is denoted by *d min* :



The minimum code distance is important because:

* **Error detection** : The code can detect up to *d min* −1 errors in a code word.
* **Error correction** : The code can correct up to errors.

In the Hamming (7,4) code, the minimum Hamming distance is 3. This means that the code can:

* **Detect** up to 2 errors in any code word.
* **Correct** 1 error.

**Geometric interpretation of the Hamming distance**

The Hamming distance can be interpreted geometrically: each code can be represented as a set of code words in a vector space, where the code words are the vertices of a hypercube. The minimum Hamming distance determines how far apart these vertices are. The larger the minimum distance, the better the code is protected from errors.

**Linear block codes**

For a block code with ***2 k*** code words of length ***n*** symbols, unless it has a special structure, the coding and decoding apparatus is very complex. Therefore, we will limit our consideration to codes that can be implemented in practice.

One of the conditions for the feasibility of block codes for large ***k*** is the condition of their linearity.

*What is a line code?*

A block code of length ***n*** symbols consisting of ***2 k*** code words is called a linear ( ***n, k*** )-code provided that all its ***2 k*** code words form ***a k*** -dimensional subspace of the vector space ***of n-*** sequences of the binary field ***GF(2).***

*To put it simply, a binary code is linear if the sum modulo 2 (* ***mod2*** *) of two code words is also a code word of that code.*

*When working with binary codes, we will constantly encounter elements of binary arithmetic, so let's define the basic concepts.*

*A field is a set of mathematical objects that can be added, subtracted, multiplied, and divided.*

*Let's take the simplest field consisting of two elements - zero - 0 and one - 1. Let's define the operations of addition and multiplication for it:*

*0+0=0, 0× 0=0;   
0+1=1, 0× 1=0;1+0=1, 1× 0=0;1+1=0, 1× 1=1.*

*The operations of addition and multiplication defined in this way are called addition modulo 2 (* ***mod2*** *) and multiplication modulo 2.*

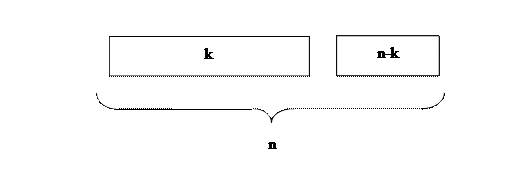
*Note that from the equality* ***1+1 = 0*** *it follows that* ***-1 = 1*** *and, accordingly,* ***1+1=1-1*** *, and from the equality* ***1×1=1*** *it follows that* ***1:1=1*** *.*

*An alphabet of two symbols 0 and 1 together with addition and multiplication* ***mod2*** *is called a field of two elements and is denoted as* ***GF(2). All methods of linear algebra are applicable*** *to the field* ***GF(2)*** *, including matrix operations.*

*Let us once again draw attention to the fact that all operations on symbols in binary codes are performed modulo 2.*

*A desirable quality of linear block codes is systematicity.*

*The systematic code has the format shown in Fig. i.e. it contains an invariable information part of length* ***k*** *symbols and a redundant (check) part of length* ***n – k*** *symbols.*



*A block code that has the properties of linearity and systematicity is called a linear block systematic (* ***n, k*** *) code.*

**Matrices for encoding and decoding**

**Generator matrix**

For a linear block code, the coding process can be efficiently described using **a generator matrix** *G* . This matrix transforms the information bits into code words with the addition of redundancy, which is necessary for error detection and correction. The main task of the generator matrix is the linear transformation of the input data (information bits) into code words.

**Formal definition**

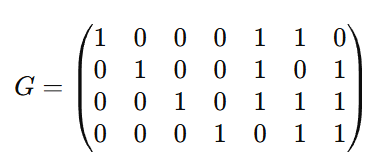
Let the original message be a vector *x* of length *k* (where *k* is the number of information bits). To encode this message, we multiply it by the generator matrix *G* , which has size *k* × *n* (where *n* is the length of the code word):

*y* = *xG*

where *y* is a code word of length *n* . Each row of the generator matrix represents one code word corresponding to a specific combination of information bits.

**For the Hamming (7,4) code** , we have 4 information bits and a 7-bit code word, of which 3 bits are redundant.

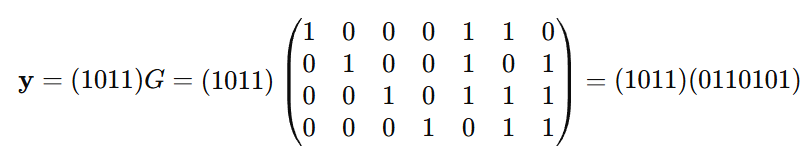
The generator matrix for the Hamming (7,4) code is as follows:



- The first 4 columns form the identity matrix, which is responsible for the original information bits.

- The last 3 columns represent the check bits that are calculated to check and correct errors.

Now, to encode a message, for example, x=(1011), we multiply this vector by the generator matrix:



So, the code word for message 1011 is 0110101.

**Verification matrix**

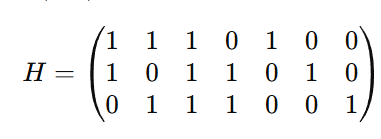
In order to detect errors on the receiver side, **a check matrix is used.** *H* , which helps to determine where exactly the error occurred, if it occurred. The check matrix allows to calculate **the error syndrome** .

**Definition of a check matrix**

The parity-check matrix *H* for a linear code is constructed in such a way that the following condition is satisfied:

Hy T =0

The matrix *H* for the Hamming (7,4) code is as follows:



This matrix has size ( *n* − *k* )× *n* , that is, 3×7, and is used for error detection.

**Error detection using a check matrix**

When a codeword *y* is received by the receiver, it multiplies it by the transposed parity-check matrix *H T* and obtains **the error syndrome** s:

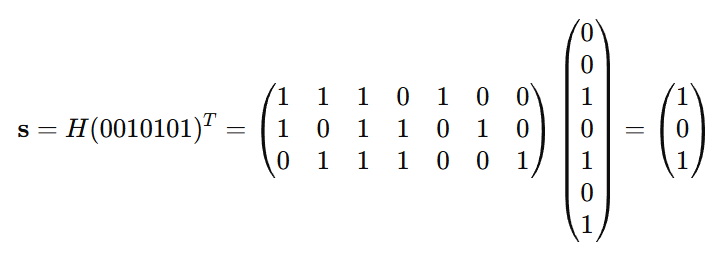
*s* = *Hy T*

* If syndrome s=0, it means that there are no errors and the code word was received correctly.
* If the syndrome is not zero (s≠0), this indicates the presence of an error, and the syndrome helps to determine its position.

**Example of error detection**

Suppose that an error occurred while transmitting the code word 0110101, and the receiver received 0010101 (the second bit was changed).

To detect an error, we multiply the resulting code word by the check matrix:



Syndrome (101) indicates the position of the second error in the code word. We can correct this error by changing the second bit back to 1, getting the correct code word 0110101.

**Real life applications of block codes**

Block codes are widely used in various areas of digital communications and data storage to ensure the reliability of information during its transmission or storage. They help correct errors that occur due to interference, equipment failures or external factors. Let's consider several real-life applications of block codes where they play a key role.

**1. RAID disk arrays**

**RAID (Redundant Array of Independent Disks)** is a technology used to improve the reliability of data storage and increase the performance of disk systems. The technology is based on the principle of using redundancy to restore data in the event of damage to one of the disks.

**How are block codes used in RAID?**

* **RAID 5** and **RAID 6** use redundant data (similar to redundant bits in block codes) that is distributed across all disks. This allows data to be recovered if one or two disks fail.
* An example of a block code is a simple scheme with the addition of control bits, which are used to recover lost information from a damaged disk.

**Example:**

Imagine an array of 4 disks where data is divided into blocks. For each block, a checksum (or check bit) is calculated and stored on a separate disk. If one disk fails, these checksums allow its contents to be reconstructed.

**2. Wi-Fi (wireless networks)**

In wireless data networks such as **Wi-Fi** , block codes play an important role in protecting information from errors caused by interference in the transmission channel (e.g. from walls, electromagnetic noise, etc.).

**How do block codes work in Wi-Fi?**

**802.11** standard uses error-correcting codes, such as **Reed-Solomon** or **convolutional codes** , to minimize the impact of errors that occur when transmitting packets over wireless channels.

* **Error correction coding** adds redundant bits to each data packet, allowing the receiver to detect and correct errors.

**Example:**

When you transmit a file over Wi-Fi, it is divided into packets, and control bits are added to each packet. If part of a packet is lost due to interference, the receiver can correct the errors using block codes.

**3. Mobile networks**

Mobile networks such as **3G** , **4G** , and **5G** use block codes to ensure reliable data transmission in changing communication channels. Due to the characteristics of mobile devices and the movement of users (for example, when driving a car), data transmission can be unstable.

**How are block codes used in mobile networks?**

Mobile networks actively use **codes (LDPC)** and **turbo codes** , which are powerful block codes for error correction. These codes provide correction of a significant number of errors during data transmission, which is especially important for streaming video, calls and the Internet.

* These codes reduce the number of repeated data transmissions and increase the reliability of communication.

**Example:**

When you watch videos on your phone over a mobile network, block codes correct errors caused by signal fluctuations or data loss to ensure smooth playback.

**Using Block Codes in RAID**

1. Simulate a RAID 5 system using redundant block code.
2. Divide data into blocks, add control bits to protect against failure of one of the disks.
3. Simulate a situation where one of the blocks is damaged and recover its data.