Belarusian State Technological University
Department of Information Systems and Technology

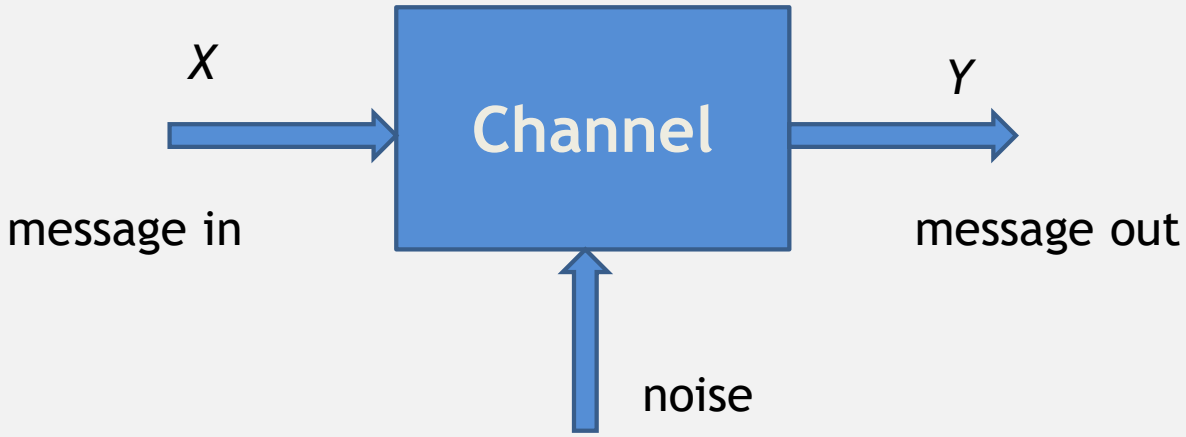
Pavel Urbanovich

INFORMATION PROTECTION.

Part 5:ERROR CORRECTING CODES

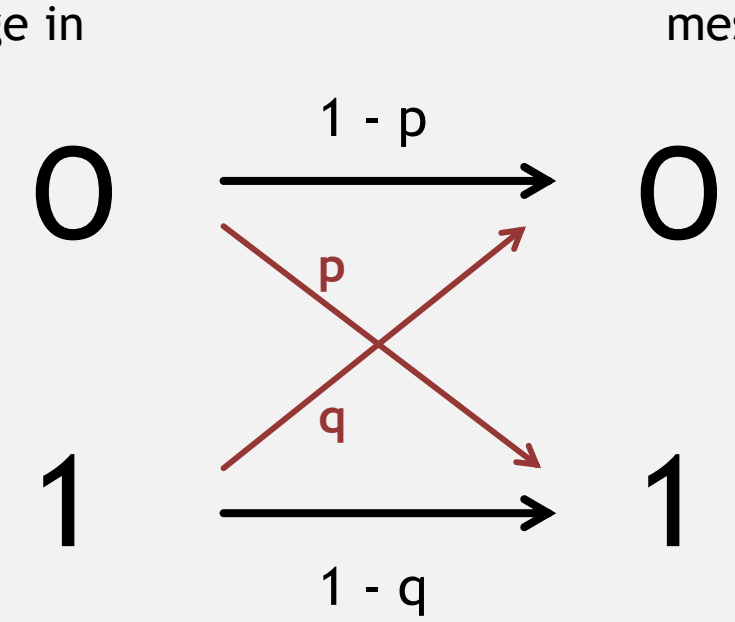
pav.urb@yandex.by, p.urbanovich@belstu.by

Bit-In, Bit-Out Model of Overall Path: Binary Symmetric Channel



Suppose that during transmission a „0” is turned into a „1” or a „1” is turned into a „0” with probability p, independently of transmissions at other times.

This is a ***Binary Symmetric Chanel*** (**BSC**) – a useful and widely used abstraction.

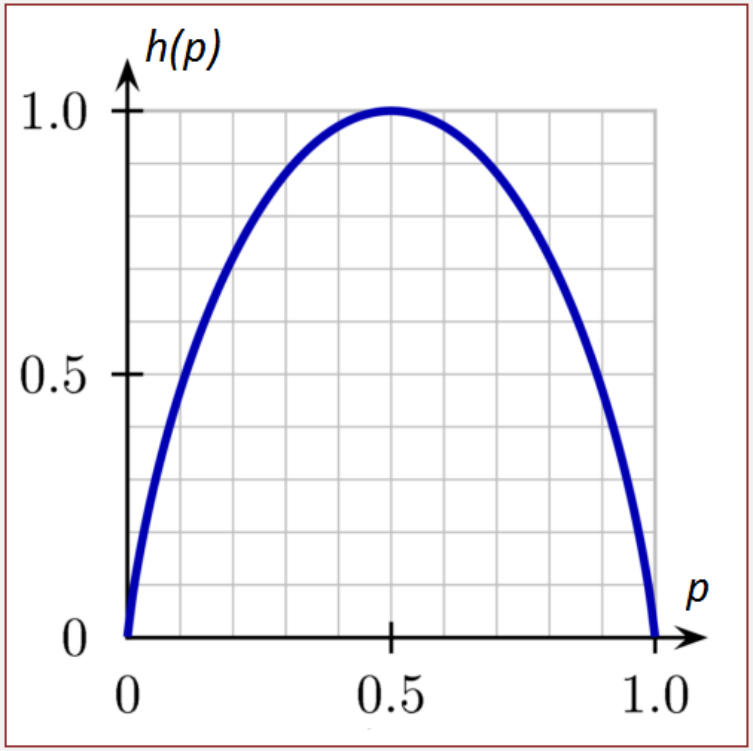


$$P(0 \mid 0) = 1 - p,$$
$$P(1 \mid 0) = p,$$
$$P(0 \mid 1) = q,$$
$$P(1 \mid 1) = 1 - q.$$

Binary Entropy Function, $h(p)$

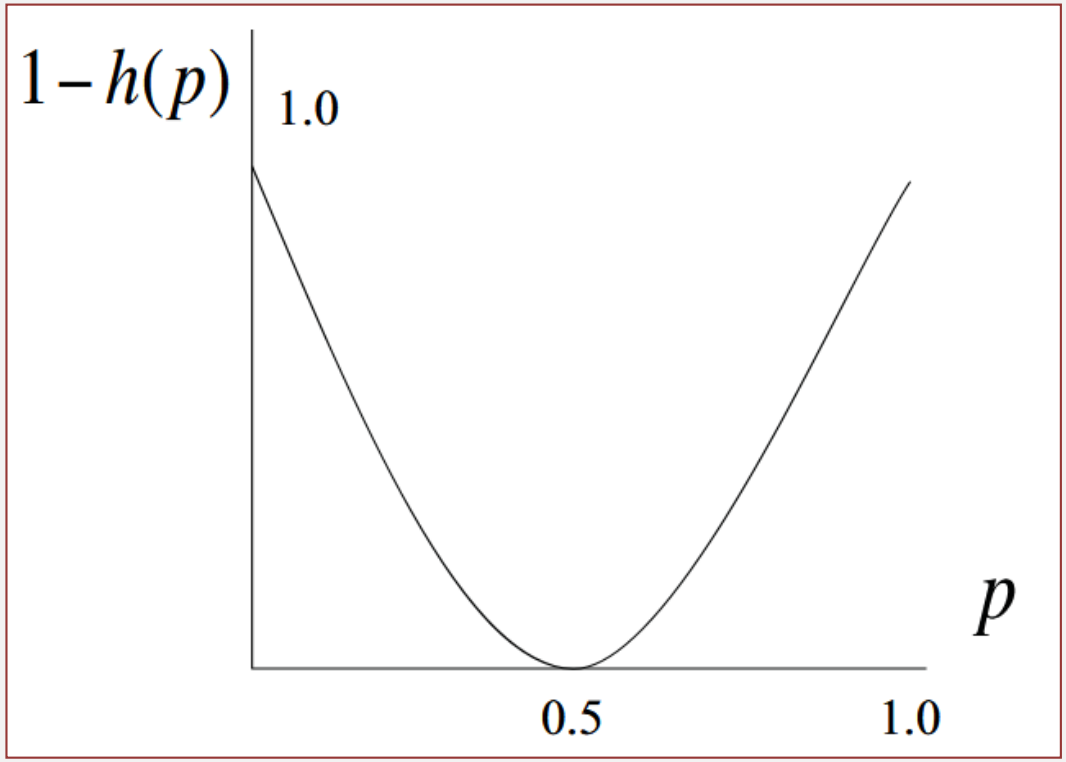
Heads (or C=1) with probability p

Tails (or C=0) with probability $1 - p$



$$H(C) = - p\log_2 p - (1-p)\log_2 (1-p) = h(p)$$

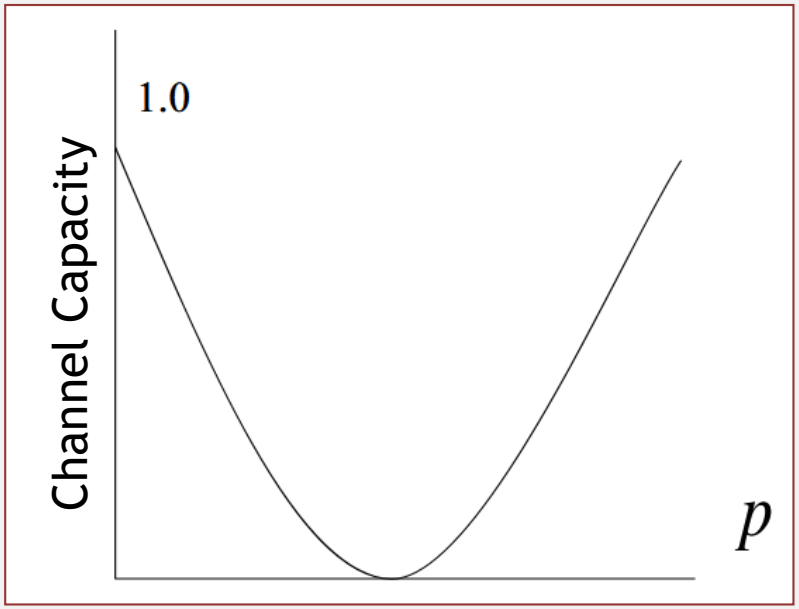
So mutual information between input and output of the BSC with equally likely inputs looks like this:



Easiest to compute as $C = max\ \{H(Y) - H(Y|X)\}$, still over all possible probability distributions for X.

The second term doesn't depend on this distributions, and the first term is maximized when **0** and **1** are equally likely at the input. So invoking our mutual information example earlier:

➡ $C = 1 - h(p)$



What channel capacity tells us about **how fast** and **how accurately** we can communicate?

The magic of asymptotically error-free transmission at any rate $R < C$

Shannon showed that one can theoretically transmit information (i.e., message bits) at an average rate $R < C$ per use of the channel, **with arbitrarily low error**.

(He also showed the converse, that transmission at an average rate $R \geq C$ incurs an error probability that is lower-bounded by some positive number.)

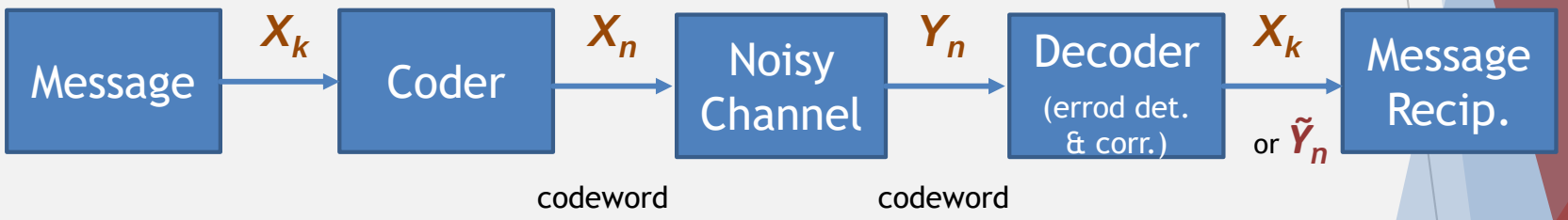
The secret: Encode blocks of k message bits into n-bits codewords, so $R = k/n$, **with k and n large**.

Encoding block of k message bits into n-bits codewords to protects against channel errors in an example of **channel coding**.

General Model

Errors introduced by the **noisy channel**:
- changed fields in the codeword (e.g. a flipped bit)
- missing fields in the codeword (e.g. a lost byte). Called erasures

How the decoder deals with errors:
- error detection vs. error correction



Hamming Distance

The number of bit positions in witch the corresponding bits of two encodings of the same length are different.

The Hamming Distance (HD) between a valid binary codeword and the same codeword with ***e*** errors is **e**.

The problem with no coding is that the two valid codewords („0" and „1") also have a Hamming Distance of 1. So single-bit error changes a valid codeword into another valid codeword…

The Hamming Distance between two words ***u*** and ***v***:

$$d_H(u, v) = w_H(u+ v)$$

Hamming Weight of vector *v* $W_H(v)$ is the non-zero number of coordinates of this vector.

Minimum Hamming Distance of Code vs. Detection & Correction Capabilities

If ***d*** is the minimum Hamming distance between codewords, we can **detect** **all** patterns of <= (d – 1) bit errors.

If d is the minimum Hamming distance between codewords, we can **corrects** **all** patterns of $\left[\frac{d-1}{2}\right]$ or fewer bit errors.

If d is the minimum Hamming distance between codewords, we can:

- detect all patterns of up to t bit errors if and only if

$$d \geq t+1$$

- correct all patterns of up to t bit errors if and only if

$$d \geq 2t+1$$

- detect all patterns of up to t_d bit errors while correcting all patterns of t_c ($<t_d$) errors if and only if

$$d \geq t_c+t_d+1$$

Binary Arithmetic

Computations with binary numbers in code construction will involve Boolean algebra, or algebra in “GF(2)” (Galois field of order 2), or modulo-2 algebra:

0+0=0,
1+0=0+1=1,
1+1=0
0*0=0*1=1*0 =0,
1*1=1

A Simple Code: Parity Checks

• Add a parity bit to message of length k to make the total number of „1” bits even (aka „even parity”).

• If the number of „1” s in the received word is *odd*, there has been an error.

0 1 1 0 0 1 0 1 0 0 1 1 → original word with parity bit
0 1 1 0 0 **0** 0 1 0 0 1 1 → single-bit error (detected)
0 1 1 0 0 **0 1** 1 0 0 1 1 → 2-bit error (not detected)

• Minimum Hamming distance of parity check code is 2:
- can detect all single-bit errors,
- in fact, can detect all odd number of errors,
- but cannot detect even number of errors,
- and cannot correct any errors.

Binary Redundant Codes

1. The use of the redundancy code involves attaching (concatenating) **additional data (X_r)** to the **transmitted message (X_k)**. These redundant data are used to control the accuracy of the transmission (improve transmission reliability).

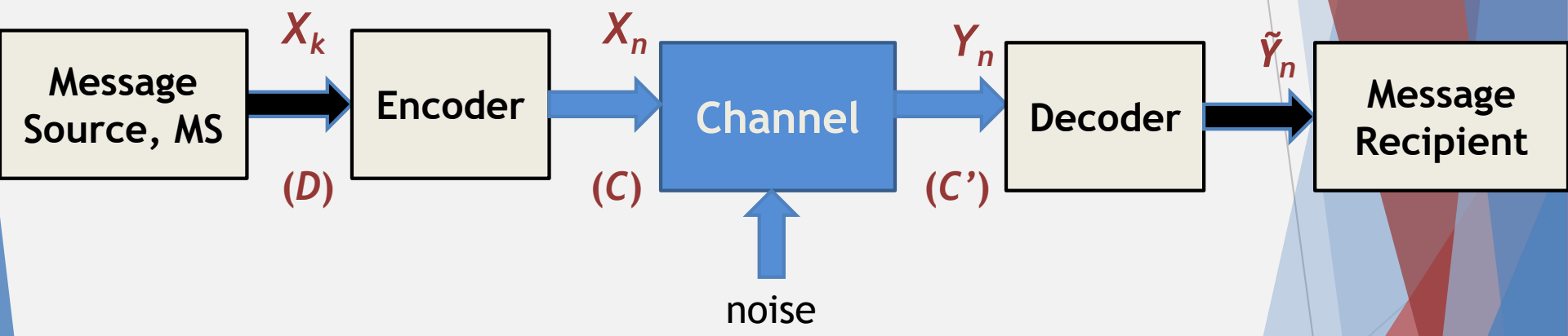
2. Due to the way to add this information codes are divided into **block** and **recursive**.

3. Block codes require sharing of input information to blocks of equal, fixed length (***k***). Then the redundant elements (***r***) are calculated for each block separately and added (concatenated) to it: $k + r = n$.

4. Recurrence codes do not require division of information - input data is processed on-line.

•**Encoding** means that the message ***D*** (X_k) (before channel) must be converted to ***C*** (X_n) (see (2)) based on the existing code in a certain time.

•**Decoding** means that the received message (Y_n) (after channel) must be converted to X_n ($\tilde{Y}_n$), taking into account transmission errors and algorithm features → X_n .



The scheme of information system with redundant encoding/decoding of transmitted messages

(compare with the picture on slide 8).

Linear Block Codes

Block code: ***k*** message bits encoded to ***n*** code bits, i.e., each of 2^k messages encoded into a unique n-bit combination via a *linear transformation*, using GF(2) operations:

$$C = D * G , \qquad (1)$$

C is an ***n***-element row vector containing the codeword
D is a ***k***-element row vector containing the message
G is the ***k x n*** *generator matrix* (or *generating matrix*)

Each codeword bit is a specified linear combination of message bits.

Key property:
Sum of any two codewords is also a codeword ⟶
necessary and sufficient for code to be **linear.**

Minimum HD of Linear Code

- (**n**,**k**)-code has rate **k**/**n**

- sometimes written as (**n**,**k**,**d**), where **d** is the minimum HD of the code

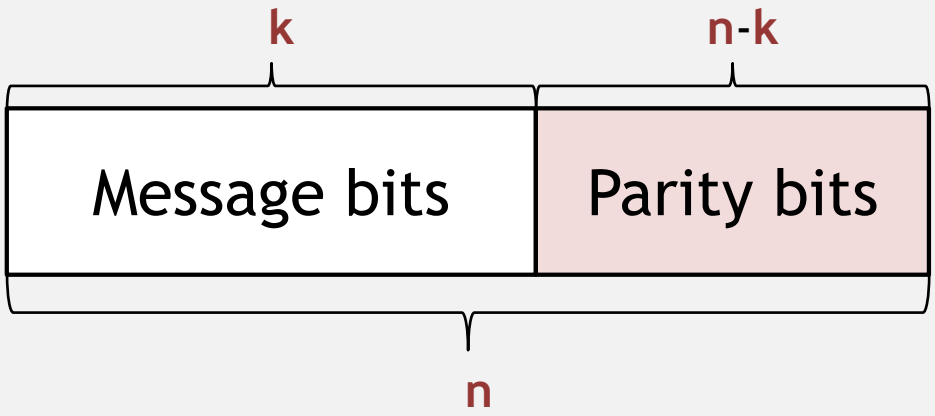
- the „weight" of a code word is the number of 1's in it

- the minimum HD of a linear code is the minimum weight found in its nonzero codewords

(n,k)-Systematic Linear Block Codes

• Split data into ***k***-bit blocks.

• Add (***n-k***) parity bits to each block using (***n-k***) linear equations, making each block n bits long.



The entire block is the called the „code word in systematic form”.

• Every linear code can be represented by an equivalent systematic form.

•Corresponds to choosing **G = [I|A]**, i.e., the identity matrix in the first ***k*** columns.

Matrix Notation

Task: given ***k-bit message***, compute ***n-bit codeword***.
We can use standard matrix arithmetic (modulo 2) to do the job.

Example, here's how we would describe the ***(9,4,4)-code*** that includes an overall parity bit.
Using (1):

$$D_{1k} = G_{kn} * C_{1n} \qquad (2)$$

$$[D_1\ D_2\ D_3\ D_4] \bullet \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = [D_1\ D_2\ D_3\ D_4\ P_1\ P_2\ P_3\ P_4\ P_5]$$

(1 x k)-Message-vector

k x n **generator matrix**

1 x n code word vector

The generator matrix, $G_{kxn} = [I_{kxk} \mid A_{kx(n-k)}]$

Generator Matrix and Parity-check Matrix

A **parity-check matrix (H)** of a linear block code is a matrix which describes the linear relations that the components of a codeword must satisfy.
It can be used to decide whether a particular vector is a codeword and is also used in decoding algorithms.

The **parity check matrix** for a given code can be derived from its **generator matrix** (and vice versa).
If the generator matrix for an [*n*,*k*]-code is in standard form

$$G = [\, I_k \mid P \,],$$

then the parity check matrix is given by

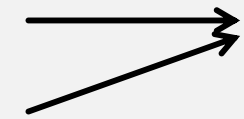
$$H = [-P^T \mid I_r],$$

because

$$G H^T = P - P = 0.$$

Example. If a binary (5,2)-code has the generator matrix

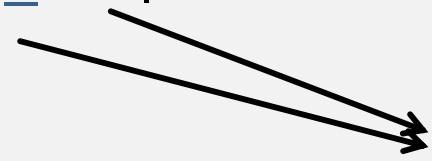
$$G = \begin{matrix} 1 & 0 & | & 1 & 0 & 1 \\ 0 & 1 & | & 1 & 1 & 0, \end{matrix}$$



$w() \geq 2$

then its parity check matrix is

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



$w() \geq 2$

$$H_{r \times n} = A_{r \times k} \mid I_{r \times r}$$

$$G_{k \times n} = I_{k \times k} \mid A_{k \times r}$$

Task. Create a generator and parity-check matrix for (7,4)-code

Hamming Codes

• Hamming codes correct single errors with the minimum number of the parity bits:

$$n = 2^{n-k} - 1 \qquad (3)$$

• (7,4,3)
•(15, 11, 3)

Using (3):

$$k + r = 2^r - 1$$

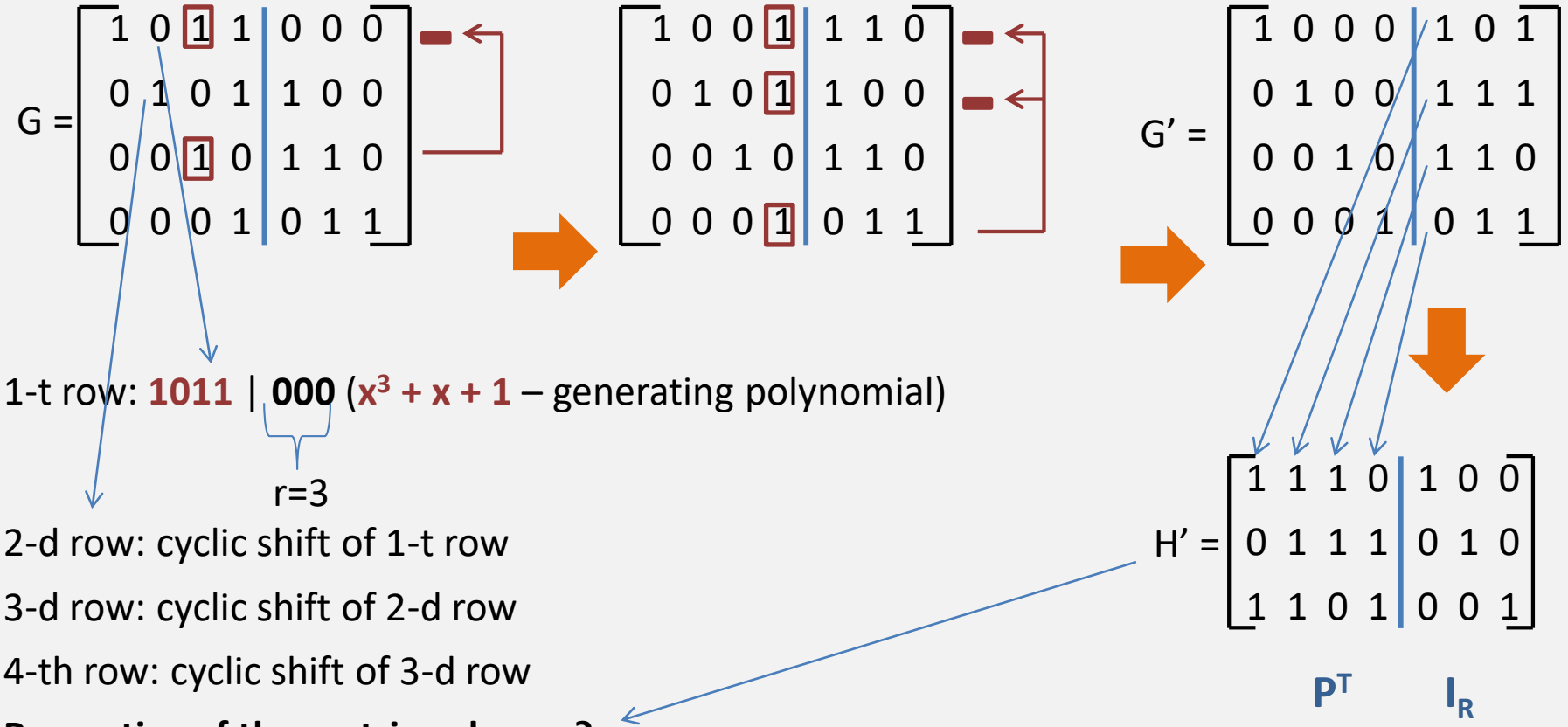
and

$$r \approx \log_2 (k+1) \qquad (4)$$

(7,4) Binary Hamming Code Systematic Encoding Matrix (k=4, r=3)

based on the **generating polynomial $x^3 + x + 1$**

Every linear code is equivalent to a *systematic* code:



1-t row: **1011** | **000** (**$x^3 + x + 1$** – generating polynomial)

r=3

2-d row: cyclic shift of 1-t row

3-d row: cyclic shift of 2-d row

4-th row: cyclic shift of 3-d row

Properties of the matrix columns?

Task. Create a generator and parity-check matrixes for (7,4)-code, if generating polynomial is **1101 (x3+x2+1).**

Encoding: The codeword X_n *can be obtained on the basis of the following identity:*

$$H * (X_n)^T = 0 \qquad (5)$$

$$X_n = \underbrace{x_1, x_2, \ldots, x_k}_{X_k}, \underbrace{x_{r1}, x_{r2}, \ldots, x_{rr}}_{X_r}$$

X_k message bits

X_r parity bits

$r1 = k+1,\ r2 = k+2, \ldots,\ rr = k+r = n$

$$A = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1k} \\ h_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ h_{r1} & \cdots & \cdots & h_{rk} \end{bmatrix}$$

then

$$x_{ri} = \sum h_{ij} * x_j \ (\text{mod}2) \qquad (6)$$

i = 1,2,…r; j=1,2,…,k

Decoding: In the case of a linear code C of length n, it is assumed that the data transmission channel adds an error vector (e) to codeword (transmitted message X_n) .

•Let the codeword X_n be transmitted and the word Y_n be accepted; then the noise in the channel gives the effect of adding to X_n the error vector e:

$$e_n = Y_n - X_n,$$

so that $Y_n = X_n + e_n$ and $X_n = Y_n + e_n$ (7)

•The decoding problem consists in calculating the vectors X_n and e by given Y_n (one of).

•The weight of the vector enis equal to the number of positions in which Y_n and X_n are differ.

Def. The ***r***-bit vector $S\ (Y_n)$ is called the **syndrome** of Y_n and is defined by the equation:

$$S\ (Y_n) = H\ (Y_n)^T \qquad (8)$$

•Two vectors, Y_n and e_n, which differ in the code word X_n, $(Y_n - e_n = X_n)$ have the same syndrome, since

$$H * (Y_n)^T = H * (X_n + e_n)^T = H * (X_n)^T + H * (e_n)^T = 0 + H * (e_n)^T = H * (e_n)^T$$

see (5)

- It follows from the latter that the syndrome of the vector X_n equals 0.

- **This means that if the message is transmitted without errors ($X_n = Y_n$), then the syndrome will have zero weight;**
If $X_n \neq Y_n$ then the syndrome will have not zero weight.

Syndrome calculation.

$$H * (Y_n)^T = S(Y_n) = Y_r + (Y_r)'; \qquad (9)$$

$Y_n = \underbrace{y_1, y_2, \ldots, y_k}_{Y_k}, \underbrace{y_{r1}, y_{r2}, \ldots, y_{rr}}_{Y_r}$

Y_k message bits

Y_r parity bits

$$(y_{ri})' = \sum h_{ij} * y_j \ (\text{mod}2) \qquad (10)$$

$i=1,2,\ldots r;\ j=1,2,\ldots,k$

Syndrome decoding allows to calculate e_n and, thus, to locate the erroneous bit: the symbols '1' in the vector e_n are located at the positions of the erroneous bits in the received message Y_n

With a single error in the message Y_n, its syndrome corresponds to the vector-column of the matrix H, whose number corresponds to the number of the erroneous bit.

Error correction.

We use (7):

$$X_n = Y_n + e_n \qquad (11)$$

What will be syndrome, if 2, 3, ... errors will appear in a message Y_n?

Example.

Let $X_k = X_4 = 1010$; $x_1=1$, $x_2=0$, $x_3=1$ $x_4=0$

Let's analyze the algorithm of encoding /decoding this message.

We see – $k=4$, according to (4) – $r=3$.

We can use matrix H on slide 22:

$$H' = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

Encoding. We use (5) and (6):

$$x_{r1} = h_{11} * x_1 + h_{12} * x_2 + h_{13} * x_3 + h_{14} * x_4 = 1*1+1*0+1*1+0*0 = 0$$

$$x_{r2} = h_{21} * x_1 + h_{22} * x_2 + h_{23} * x_3 + h_{24} * x_4 = 0*1+1*0+1*1+1*0 = 1$$

$$x_{r3} = h_{31} * x_1 + h_{32} * x_2 + h_{33} * x_3 + h_{34} * x_4 = 1*1+1*0+0*1+1*0 = 0,$$

So, $X_r = X_3 = 010$ and codeword $X_n = X_7 = 1010\ 010$.

Decoding.

1.Let $Y_n = Y_7 = 1010\ 010$:

$y_1=1$, $y_2=0$, $y_3=1$ $y_4=0$; $y_{r1}=0$, $y_{r2}=1$, $y_{r3}=0$

Syndrome calculation. We use (9) and (10):

$(y_{r1})' = h_{11} * y_1 + h_{12} * y_2 + h_{13} * y_3 + h_{14} * y_4 = 1*1+1*0+1*1+0*0 = 0$

$(y_{r2})' = h_{21} * y_1 + h_{22} * y_2 + h_{23} * y_3 + h_{24} * y_4 = 0*1+1*0+1*1+1*0 = 1$

$(y_{r3})' = h_{31} * y_1 + h_{32} * y_2 + h_{33} * y_3 + h_{34} * y_4 = 1*1+1*0+0*1+1*0 = 0,$

So, $(Y_r)' = (Y_3)' = 010$ and Syndrome $S(Y_n) =$ 010 + 010 = 000

Syndrome analysis.

Syndrome Wight $w(S)=0$ it means – message $Y_n = Y_7 = 1010\ 010$ is free from errors, $e_n = e_7 = 0000\ 000$

$X_n = Y_n + e_n = 1010\ 010 + 0000\ 000 = 1010\ 010$ (this operation – error correction -may not be performed in the absence of errors)

Decoding.

2.Let $Y_n = Y_7 = 1011\ 010$:

$y_1=1,\ y_2=0,\ y_3=1\ y_4=1;\ y_{r1}=0,\ y_{r2}=1,\ y_{r3}=0$

Syndrome calculation. We use (9) and (10):

$(y_{r1})' = h_{11} * y_1 + h_{12} * y_2 + h_{13} * y_3 + h_{14} * y_4 = 1*1+1*0+1*1+1*0 = 0$

$(y_{r2})' = h_{21} * y_1 + h_{22} * y_2 + h_{23} * y_3 + h_{24} * y_4 = 0*1+1*0+1*1+1*1 = 0$

$(y_{r3})' = h_{31} * y_1 + h_{32} * y_2 + h_{33} * y_3 + h_{34} * y_4 = 1*1+1*0+0*1+1*1 = 1,$

So, $(Y_r)' = 001$ and Syndrome $S(Y_n) = 010+001 = 011$

Syndrome analysis.

Syndrome Wight $w(S)>0$ it means – message

$Y_n = Y_7$ = 1011 *010* contains errors,

we see $S(Y_n) = \{h_4\}$, h_4 -the fourth vector – column of matrix H and $e_n = e_7 = 0001\ 000$

$X_n = Y_n + e_n$ = 1011 *010* + *0001 000* = 1010 *010*

The error is corrected.

3. What will happen if:

a) $Y_n = Y_7$= 1010 *011*,

b) $Y_n = Y_7$= 1011 *011*,

c) $Y_n = Y_7$= 1111 *111*.

References:

1.Hamming, R. W. Error Detecting and Error Correcting Codes, Bell System Technical Journal, April 1950, 29 (2), p.147–160.

2. URL: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science

3. URL: https://orion.math.iastate.edu/linglong/Math690F04/HammingCodes.pdf

4. URL: http://www.strongsec.com/zhw/EEC_4.pdf

5. Fan, John L. Constrained Coding and Soft Iterative Decoding. Kluwer International Series in Engineering and Computer Science, Boston: Kluwer Academic Publishers, 2001.

6. W. Wesley Peterson & E. J. Weldon, Error-Correcting Codes, Second Edition, MIT Press, 1972.

7. Урбанович, П. П. Информационная безопасность и надежность систем : учебно-методическое пособие по одноименному курсу для студентов специальности 1-40 01 02-03 "Информационные системы и технологии" / П. П. Урбанович, Д. М. Романенко, Е. В. Романцевич. - Минск : БГТУ, 2007. - 87 с. (URL: http://elib.belstu.by/handle/123456789/2937)

8. Урбанович, П. П. Защита информации и надежность информационных систем : пос. для студ. вузов спец. 1-40 05 01-03 «Информационные системы и технологии (издательско-полиграфический комплекс)» / П. П. Урбанович, Д. В. Шиман.- Минск : БГТУ, 2014. - 91 с. (URL: https://elib.belstu.by/handle/123456789/23761)

9. Gorbunova, Yu. W-cyclic method of interleaving of the data for communication systems / Yu. Gorbunova, P. Urbanovich // PRZEGLĄD ELEKTROTECHNICZNY. – 2012. - R. 88 NR 11b. – P. 344-345. (URL: https://elib.belstu.by/handle/123456789/3636)

10. Урбанович, П. П. Избыточность в полупроводниковых интегральных микросхемах памяти / П. П. Урбанович, В. Ф. Алексеев, Е. А. Верниковский. - Минск : Навука і тэхніка, 1995. - 262 с. (URL: https://elib.belstu.by/handle/123456789/24777)

11. Urbanovich, P. P. The algorithm for determining of the errors multiplicity by multithreshold decoding of iterative / Pavel P. Urbanovich, Marina F. Vitkova , Dmitri M. Romanenko // PRZEGLĄD ELEKTROTECHNICZNY. - 2014. - R. 90 № 3.
(URL: (URL: https://elib.belstu.by/handle/123456789/24777)

12. Urbanovich, P. Channel Adapted Decoding Algorithms for Correction Modular Errors and Erasures in Communication System / P. Urbanovich, N. Patsei, D. Romanenko, D. Shiman, Y. Bulova // 9th International Conference “New Electrical and Electronic Technologies and their Industrial Implementation” – NEET’2015, Zakopane, Poland, June 23 – 26, 2015. – P. 23.
(URL: https://elib.belstu.by/handle/123456789/25707)

13. Urbanovich, P. P. Multithreshold majority decoding of LDP-codes / P.P. Urbanovich, D.M. Romanenko, D.V. Shiman // 7th International Conference “New Electrical and Electronic Technologies and their Industrial Implementation” – NEET’2011, Zakopane, Poland, June 28–July 1, 2011. – P. 152. (URL: https://elib.belstu.by/handle/123456789/25720)

14. Исследование основных характеристик многоуровневых турбо кодов / П.П. Урбанович [и др.] // Автоматический контроль и автоматизация производственных процессов: материалы Международной научно-технической конференции, 6-8 июня 2006 г., Минск. – Мн.: БГТУ, 2006. – С. 199-202. (URL: https://elib.belstu.by/handle/123456789/25728)

15. Урбанович, П. П. Коррекция многократных ошибок в информационных словах итеративным кодом / П. П.Урбанович, Д. М. Романенко // Труды БГТУ. Сер. IV. Физико-математические науки и информатика. - Минск : БГТУ. - 1998. - Вып.VI.- С. 82-86.
(URL:https://elib.belstu.by/handle/123456789/26713)

16. Урбанович,П.П. Лабораторный практикум по дисциплинам «Защита информации и надежность информационных систем» и «Криптографические методы защиты информации». Ч.1: Кодирование информации: учебно-метод.пос./П.П. Урбанович, Д.В.Шиман, Н.П. Шутько. – Минск: БГТУ, 2019. – 95 с.