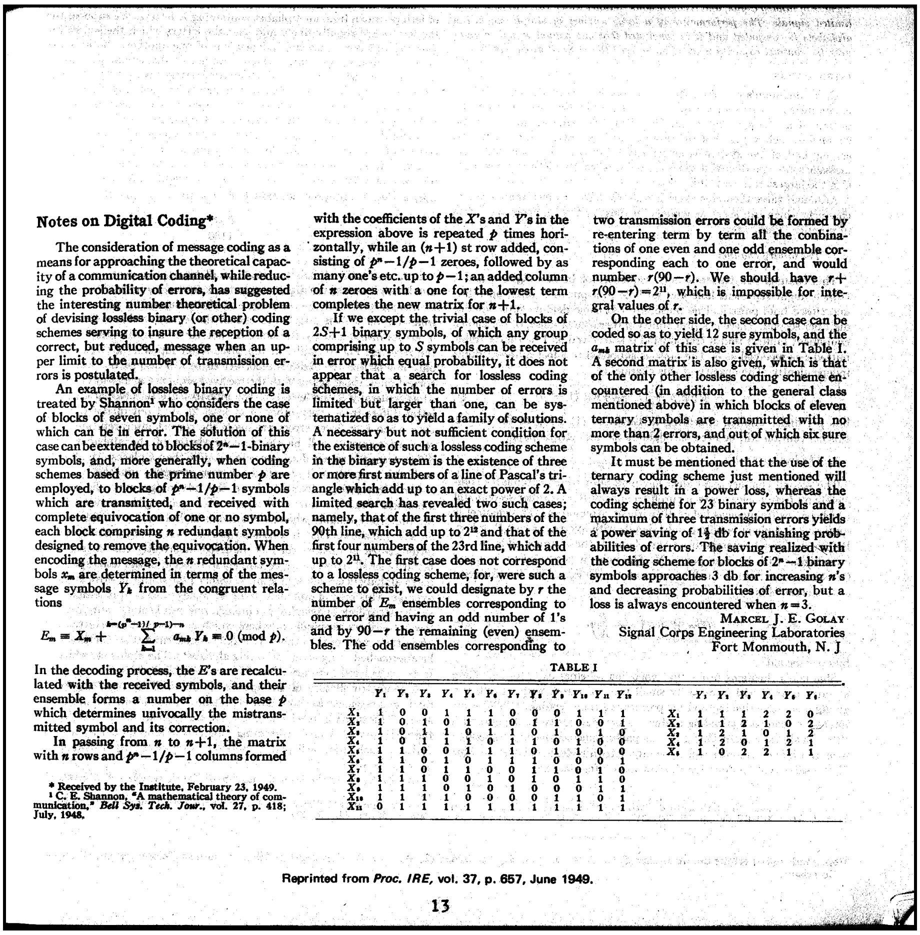
2.1. Introdução (com um pouco de história)

Códigos para controlo de erros

Um pouco de história

A investigação sobre códigos para controlo de erros, ou códigos de canal, tem seguido dois "percursos": um, com um "sabor" mais algébrico, refere-se principalmente aos *códigos de blocos*; o outro, mais probabilístico, diz especialmente respeito aos *códigos convolucionais*.

Os primeiros códigos de blocos foram introduzidos por *Richard Hamming* em 1950. Tratava-se de códigos correctores de erros simples, portanto com uma reduzida capacidade de correcção. Na mesma época *M. Golay* inventou um código corrector de erros triplos num artigo surpreendentemente curto (uma página, como se vê na figura seguinte).



Notes on Digital Coding*

The consideration of message coding as a means for approaching the theoretical capacity of a communication channel, while reducing the probability of errors, has suggested the interesting number theoretical problem of devising lossless binary (or other) coding schemes serving to insure the reception of a correct, but reduced, message when an upper limit to the number of transmission errors is postulated.

An example of lossless binary coding is treated by Shannon[1] who considers the case of blocks of seven symbols, one or none of which can be in error. The solution of this case can be extended to blocks of 2^n-1-binary symbols, and, more generally, when coding schemes based on the prime number p are employed, to blocks of $p^n-1/p-1$ symbols which are transmitted, and received with complete equivocation of one or no symbol, each block comprising n redundant symbols designed to remove the equivocation. When encoding the message, the n redundant symbols x_m are determined in terms of the message symbols Y_k from the congruent relations

$$E_m \equiv X_m + \sum_{k=1}^{k=(p^n-1)/(p-1)-n} a_{mk} Y_k \equiv 0 \pmod{p}.$$

In the decoding process, the E's are recalculated with the received symbols, and their ensemble forms a number on the base p which determines univocally the mistransmitted symbol and its correction.

In passing from n to $n+1$, the matrix with n rows and $p^n-1/p-1$ columns formed with the coefficients of the X's and Y's in the expression above is repeated p times horizontally, while an $(n+1)$ st row added, consisting of $p^n-1/p-1$ zeroes, followed by as many one's etc. up to $p-1$; an added column of n zeroes with a one for the lowest term completes the new matrix for $n+1$.

If we except the trivial case of blocks of $2S+1$ binary symbols, of which any group comprising up to S symbols can be received in error which equal probability, it does not appear that a search for lossless coding schemes, in which the number of errors is limited but larger than one, can be systematized so as to yield a family of solutions. A necessary but not sufficient condition for the existence of such a lossless coding scheme in the binary system is the existence of three or more first numbers of a line of Pascal's triangle which add up to an exact power of 2. A limited search has revealed two such cases; namely, that of the first three numbers of the 90th line, which add up to 2^{12} and that of the first four numbers of the 23rd line, which add up to 2^{11}. The first case does not correspond to a lossless coding scheme, for, were such a scheme to exist, we could designate by r the number of E_m ensembles corresponding to one error and having an odd number of 1's and by $90-r$ the remaining (even) ensembles. The odd ensembles corresponding to two transmission errors could be formed by re-entering term by term all the conbinations of one even and one odd ensemble corresponding each to one error, and would number $r(90-r)$. We should have $r+r(90-r)=2^{11}$, which is impossible for integral values of r.

On the other side, the second case can be coded so as to yield 12 sure symbols, and the a_{mk} matrix of this case is given in Table I. A second matrix is also given, which is that of the only other lossless coding scheme encountered (in addition to the general class mentioned above) in which blocks of eleven ternary symbols are transmitted with no more than 2 errors, and out of which six sure symbols can be obtained.

It must be mentioned that the use of the ternary coding scheme just mentioned will always result in a power loss, whereas the coding scheme for 23 binary symbols and a maximum of three transmission errors yields a power saving of $1\frac{1}{2}$ db for vanishing probabilities of errors. The saving realized with the coding scheme for blocks of 2^n-1 binary symbols approaches 3 db for increasing n's and decreasing probabilities of error, but a loss is always encountered when $n=3$.

Marcel J. E. Golay
Signal Corps Engineering Laboratories
Fort Monmouth, N. J

TABLE I

	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	Y_{10}	Y_{11}	Y_{12}
X_1	1	0	0	1	1	1	0	0	0	1	1	1
X_2	1	0	1	0	1	1	0	1	1	0	0	1
X_3	1	0	1	1	0	1	1	0	1	0	1	0
X_4	1	0	1	1	1	0	1	1	0	1	0	0
X_5	1	1	0	0	1	1	1	0	1	1	0	0
X_6	1	1	0	1	0	1	1	1	0	0	0	1
X_7	1	1	0	1	1	0	0	1	1	0	1	0
X_8	1	1	1	0	0	1	0	1	0	1	1	0
X_9	1	1	1	0	1	0	1	0	0	0	1	1
X_{10}	1	1	1	1	0	0	0	0	1	1	0	1
X_{11}	0	1	1	1	1	1	1	1	1	1	1	1

	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6
X_1	1	1	1	2	2	0
X_2	1	1	2	1	0	2
X_3	1	2	1	0	1	2
X_4	1	2	0	1	2	1
X_5	1	0	2	2	1	1

* Received by the Institute, February 23, 1949.
[1] C. E. Shannon, "A mathematical theory of communication," *Bell Sys. Tech. Jour.*, vol. 27, p. 418; July, 1948.

Reprinted from *Proc. IRE*, vol. 37, p. 657, June 1949.

13

O artigo em que Golay apresentou o seu código.

Só anos mais tarde surgiriam melhores códigos: o principal avanço surgiu quando *R. C. Bose* e *D. K. Ray-Chaudhuri* (em 1960) e *A. Hocquenghem* (em 1959) descobriram independentemente uma classe alargada de códigos correctores de erros múltiplos (os *códigos BCH*), e *I. Reed* e *G. Solomon* (1960) descobriram uma classe, relacionada com a primeira, de códigos para canais não binários.

Estes códigos permanecem entre as classes mais importantes de códigos; mesmo assim, desde então a teoria dos códigos tem-se desenvolvido imenso e novos códigos vão sendo descobertos de tempos a tempos.

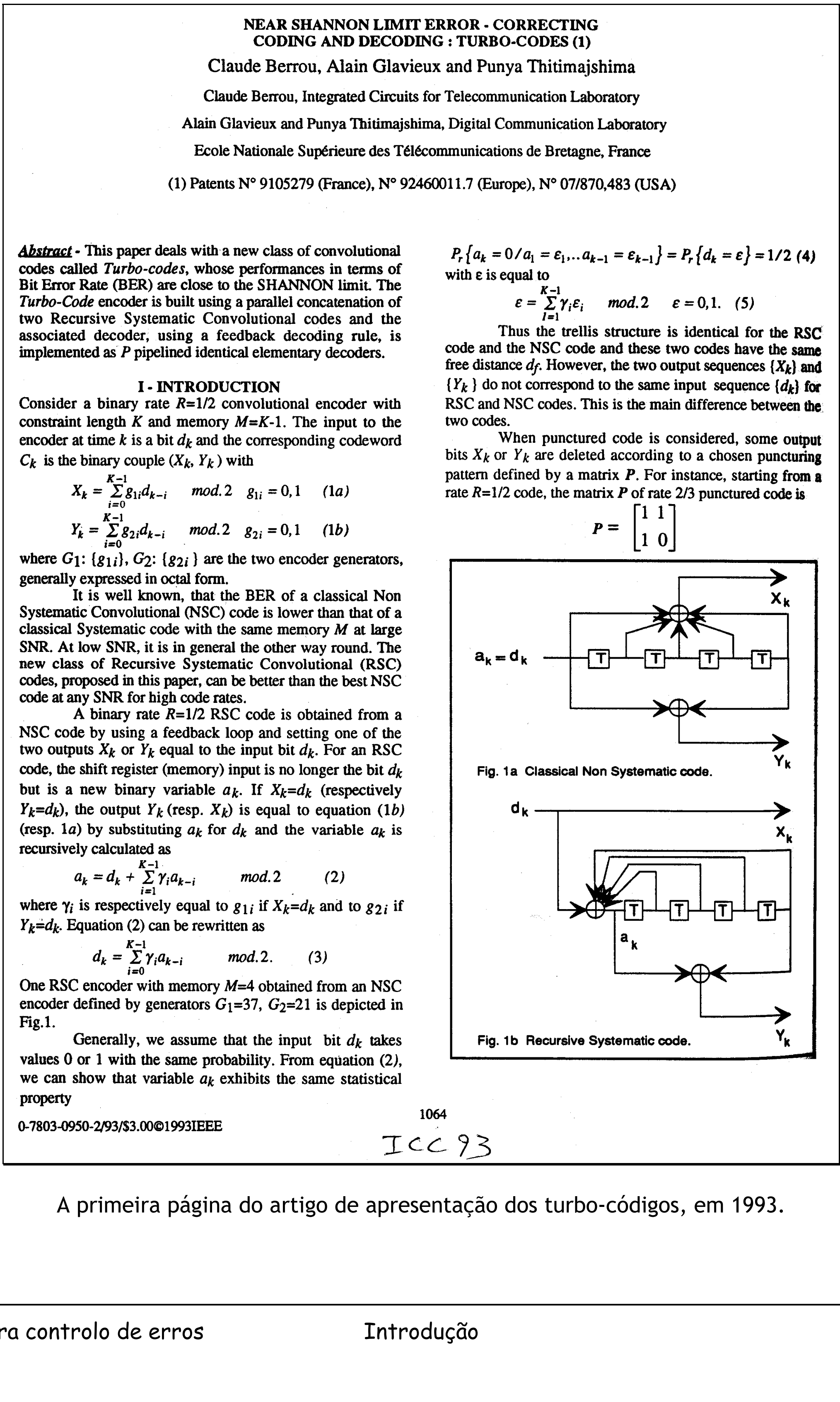
A descoberta dos códigos BCH levou à pesquisa de métodos práticos (em hardware e software) de implementar o codificador e o descodificador. O primeiro algoritmo "bom" foi proposto por *W. Peterson*. Mais tarde *E. Berlekamp* e *J. Massey* descobriram um algoritmo poderoso para efectuar os cálculos de Peterson, cuja implementação foi tornada possível através de tecnologia digital.

O outro percurso de investigação em códigos tem um "sabor" mais probabilístico, como se disse. Nos primeiros tempos os investigadores procuravam estimar a probabilidade de erro das melhores famílias de códigos de blocos, apesar de esses códigos "melhores" não serem conhecidos. Procurava-se compreender também a codificação e a descodificação de um ponto de vista probabilístico, o que levou à noção de *descodificação sequencial*. A descodificação sequencial exigiu uma nova classe de códigos sem blocos, de comprimento indefinido, representáveis por uma árvore e que pudessem ser descodificados percorrendo essa árvore. Os códigos arborescentes mais úteis são códigos altamente estruturados chamados *códigos convolucionais*, que podem ser gerados por um circuito com registos de deslocamento lineares que efectua a operação de convolução sobre a sequência de informação que se deseja codificar.

No fim dos anos 50 os códigos convolucionais foram descodificados com sucesso com algoritmos de descodificação sequencial. Um algoritmo de descodificação muito mais simples — o *algoritmo de Viterbi* — foi proposto apenas muito mais tarde (1967). Este algoritmo goza de grande popularidade em códigos convolucionais de complexidade modesta mas é impraticável com códigos mais complexos.

Na década de 70 estas duas vias de investigação foram-se aproximando uma da outra: a teoria dos códigos convolucionais foi sendo estudada pelos matemáticos, com outros pontos de vista, e também se fizeram avanços nos códigos de blocos em direcção ao prometido por *Shannon* anos antes (Teorema da Codificação de Canal: *para* $R \le C$ *existe um código…*), especialmente quando foram propostos dois novos esquemas de codificação, um proposto por *Justesen* e o outro por *Goppa*. O objectivo comum era o de criar famílias de códigos que tivessem ao mesmo tempo um comprimento de palavra longo e uma "performance" muito boa. No entanto ambos os esquemas sofrem de algumas limitações práticas.

Mais recentemente, na IEEE International Conference on Communications (ICC93), em Junho de 1993, *C. Berrou*, *A. Glavieux* e *P. Thitimajshima* apresentaram pela primeira vez os *turbo-códigos*. De certo modo pertencentes à família dos códigos convolucionais, são realizados por *concatenação em paralelo* com *entrelaçamento* e descodificados usando a técnica da *descodificação iterativa*. É uma área contemporânea de intensa actividade de investigação e que provocou o renascimento de uns outros códigos surgidos cerca de trinta anos antes, em 1963: os *códigos LDPC* ("low density parity check"), de *R. Gallager*.



NEAR SHANNON LIMIT ERROR - CORRECTING CODING AND DECODING : TURBO-CODES (1)

Claude Berrou, Alain Glavieux and Punya Thitimajshima

Claude Berrou, Integrated Circuits for Telecommunication Laboratory

Alain Glavieux and Punya Thitimajshima, Digital Communication Laboratory

Ecole Nationale Supérieure des Télécommunications de Bretagne, France

(1) Patents N° 9105279 (France), N° 92460011.7 (Europe), N° 07/870,483 (USA)

Abstract - This paper deals with a new class of convolutional codes called *Turbo-codes*, whose performances in terms of Bit Error Rate (BER) are close to the SHANNON limit. The *Turbo-Code* encoder is built using a parallel concatenation of two Recursive Systematic Convolutional codes and the associated decoder, using a feedback decoding rule, is implemented as *P* pipelined identical elementary decoders.

I - INTRODUCTION

Consider a binary rate R=1/2 convolutional encoder with constraint length K and memory M=K-1. The input to the encoder at time k is a bit d_k and the corresponding codeword C_k is the binary couple (X_k, Y_k) with

$$X_k = \sum_{i=0}^{K-1} g_{1i} d_{k-i} \quad mod.2 \quad g_{1i} = 0,1 \qquad (1a)$$

$$Y_k = \sum_{i=0}^{K-1} g_{2i} d_{k-i} \quad mod.2 \quad g_{2i} = 0,1 \qquad (1b)$$

where G_1: $\{g_{1i}\}$, G_2: $\{g_{2i}\}$ are the two encoder generators, generally expressed in octal form.

It is well known, that the BER of a classical Non Systematic Convolutional (NSC) code is lower than that of a classical Systematic code with the same memory M at large SNR. At low SNR, it is in general the other way round. The new class of Recursive Systematic Convolutional (RSC) codes, proposed in this paper, can be better than the best NSC code at any SNR for high code rates.

A binary rate R=1/2 RSC code is obtained from a NSC code by using a feedback loop and setting one of the two outputs X_k or Y_k equal to the input bit d_k. For an RSC code, the shift register (memory) input is no longer the bit d_k but is a new binary variable a_k. If X_k=d_k (respectively Y_k=d_k), the output Y_k (resp. X_k) is equal to equation (1*b*) (resp. 1*a*) by substituting a_k for d_k and the variable a_k is recursively calculated as

$$a_k = d_k + \sum_{i=1}^{K-1} \gamma_i a_{k-i} \quad mod.2 \qquad (2)$$

where γ_i is respectively equal to g_{1i} if X_k=d_k and to g_{2i} if Y_k=d_k. Equation (2) can be rewritten as

$$d_k = \sum_{i=0}^{K-1} \gamma_i a_{k-i} \quad mod.2. \qquad (3)$$

One RSC encoder with memory M=4 obtained from an NSC encoder defined by generators G_1=37, G_2=21 is depicted in Fig.1.

Generally, we assume that the input bit d_k takes values 0 or 1 with the same probability. From equation (2), we can show that variable a_k exhibits the same statistical property

$$P_r\{a_k = 0/a_1 = \varepsilon_1,..a_{k-1} = \varepsilon_{k-1}\} = P_r\{d_k = \varepsilon\} = 1/2 \quad (4)$$

with ε is equal to

$$\varepsilon = \sum_{i=1}^{K-1} \gamma_i \varepsilon_i \quad mod.2 \quad \varepsilon = 0,1. \quad (5)$$

Thus the trellis structure is identical for the RSC code and the NSC code and these two codes have the same free distance d_f. However, the two output sequences $\{X_k\}$ and $\{Y_k\}$ do not correspond to the same input sequence $\{d_k\}$ for RSC and NSC codes. This is the main difference between the two codes.

When punctured code is considered, some output bits X_k or Y_k are deleted according to a chosen puncturing pattern defined by a matrix P. For instance, starting from a rate R=1/2 code, the matrix P of rate 2/3 punctured code is

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

X_k $a_k = d_k$ T T T T Y_k

Fig. 1a Classical Non Systematic code.

d_k X_k T T T T a_k Y_k

Fig. 1b Recursive Systematic code.

1064

0-7803-0950-2/93/$3.00©1993IEEE

ICC 93

A primeira página do artigo de apresentação dos turbo-códigos, em 1993.

É cada vez mais frequente encontrar codificadores e descodificadores em novos sistemas quer de comunicações digitais quer de armazenamento digital.

Da história à prática

A primeira aplicação bem sucedida da codificação sucedeu nas comunicações espaciais. Só assim tem sido possível, por exemplo, a recepção de imagens e outra informação proveniente da sonda Voyager I, dos confins do Sistema Solar, para lá de Plutão, para a Terra.

Hoje em dia é muito comum encontrar sistemas de codificação para controlo de erros em qualquer sistema de comunicações moderno (veja-se a televisão digital, por exemplo, ou "ouçam-se" as comunicações móveis celulares).

Uma outra aplicação é a dos sistemas de armazenamento em disco e em fita magnética, especialmente em equipamento de elevada capacidade e "performance". São usados códigos de blocos e códigos convolucionais, em esquemas concatenados com entrelaçamento.

Também nas memórias de semicondutores são usadas técnicas de codificação. O intuito é diminuir a taxa de falhas de várias falhas por hora para algumas poucas por ano. Como é preciso que os códigos sejam simples (para não ocuparem uma grande área no "chip") um dos códigos usados é o de Hamming.

A técnica de controlo de erros mais amplamente usada talvez seja a que detecta erros por "verificação cíclica" (CRC). É aplicada com sucesso em sistemas ARQ e em sistemas híbridos FEC-ARQ.

Por último, e não é demais repetir, também só com controlo de erros é possível não ouvir o efeito de riscos, arranhões e até buracos nos CDs, CD-ROMs e DVDs. A codificação da informação assenta num esquema concatenado de códigos Reed-Solomon (RS), com entrelaçamento duplo.

Também os sistemas de gravação DAT e as redes locais sem fios (WLAN) usam codificação de canal.

E assim se vai fazendo a História...

Códigos correctores de erros: introdução

- Os erros de transmissão em comunicações digitais dependem da relação sinal-ruído (S/N).
- Se S/N for fixo e a taxa de erros for muito elevada é preciso melhorar a fiabilidade por outro meio.
- Uma maneira possível é codificar a informação de modo a detectar ou a corrigir os erros (*codificação para controlo de erros)*.

Códigos para controlo de erros:	Envolvem a adição sistemática de dígitos *redundantes* os quais, só por si, não transportam informação mas tornam possível detectar e até corrigir alguns erros de transmissão.

Categorias de códigos com controlo de erros

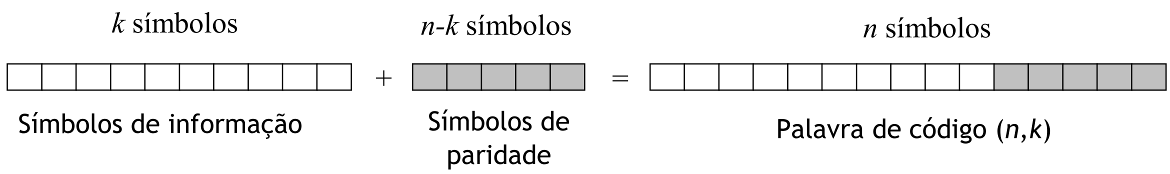
- Códigos de blocos
  - ARQ (**A**utomatic **R**epeat **R**equest)
  - FEC (**F**orward **E**rror **C**orrection)
- Códigos convolucionais

- Método ARQ: necessita de caminho de retorno e normalmente só detecta erros.
- Método FEC: detecta e corrige erros.
- A codificação para detectar erros, sem correcção, é mais simples que a codificação para os corrigir.

Códigos correctores de erros: introdução

Códigos de blocos

- • A fonte binária gera uma sequência de símbolos à taxa de r símbolos/s.
- • Estes símbolos são agrupados em *blocos* de k símbolos.
- • A cada um destes blocos de k símbolos são acrescentados n-k símbolos redundantes, produzindo uma *palavra de código* de n símbolos.
- A taxa de símbolos passa a ser $r_b = r\dfrac{n}{k}$ símbolos/s.



<u>Taxa do código</u>

$$R_c = \frac{k}{n}$$

<u>Codificação:</u>

A uma sequência de informação $\mathbf{X} = (x_1 x_2 ... x_k)$ faz-se corresponder uma palavra de código $\mathbf{Y} = (y_1 y_2 ... y_n)$ de acordo com regras bem definidas.

<u>Descodificação:</u>

A partir da sequência recebida $\mathbf{Z} = (z_1 z_2 ... z_n)$ determina-se a palavra de código mais provável, isto é, a palavra de código *mais próxima em distância de Hamming* da palavra **Z**.

$$\boxed{\text{Fonte}} \xrightarrow{X} \boxed{\text{Codificador}} \xrightarrow{Y} \boxed{\text{Canal}} \xrightarrow{Z} \boxed{\text{Descodificador}} \xrightarrow{\hat{Y} \to \hat{X}}$$

$$x_i,\ y_i,\ z_i \in \{0,1\}$$

Aplicações e tipos de codificação[1]

Aplicações de códigos para controlo de erros

Communication systems	**Control systems**	**Information security and retrieval systems**	**Information storage systems**	**Signal processing systems**
broadcasting HF, VHF, etc. line microwave military mobile networks optical power line satellite sonar space spread spectrum telemetry telex and fax troposcatter	aeronautical aerospace automobile naval railway RPV	account and card codes bar and dot codes crypto ISBN product identification codes	compact discs magnetic tapes and discs punched cards and tapes semiconductor memories	fault tolerance image coding speech coding

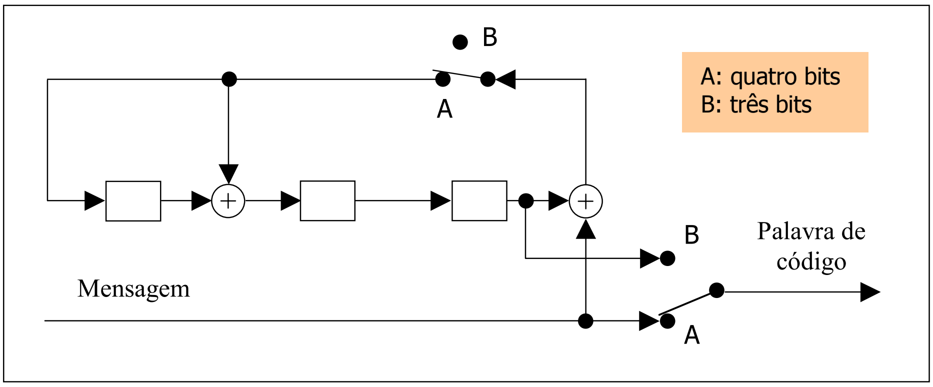
Tipos e classes de códigos vulgares

Blocos	**Convolucionais**	**Modulação codificada**	**Concatenados**
"Array"	Gerados em computador	BCM	Bloco-bloco (RS)
BCH	Dual-*k*	TCM	Convolucional-bloco (RS)
CRC	LDPC (Gallager)		Entrelaçamento cruzado (CIRC)
Fire	Hagelbarger		Turbo-códigos
Golay	Idaware		
Hamming	Paridade móvel		
Ortogonais	"random-error-correcting"		
Reed-Muller (RM)	Wyner-Ash		
Reed-Solomon (RS)			
Repetição			
SPC			

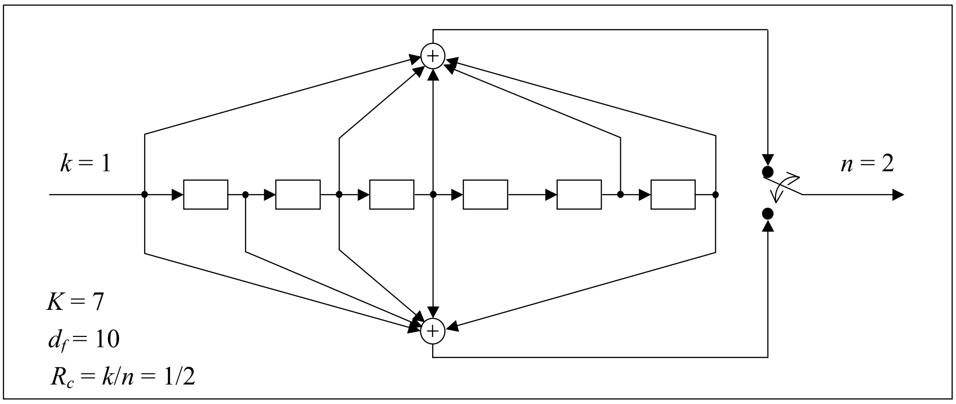
[1] In "Coding as a cure for communication calamities: the successes and failures of error control", P. G. Farrell.

Exemplos de codificação

Codificador cíclico



Codificador convolucional



International Standard Book Number (ISBN)

i	10 9 8 7 6 5 4 3 2	soma de controlo, x_1
ISBN	0 1 9 8 5 3 8 0 4	9

$$\sum_{i=1}^{10} i\, x_i = 0 \quad (\mathrm{mod}\, 11) \qquad \text{Se } x_1 = 10 \;\Rightarrow\; x_1 = X$$

Exemplos de codificação para detecção de erros

- **<u>ISBN</u>** (International Standard Book Number)

  É um conjunto de 10 caracteres identificativos de um livro. O último carácter serve para detectar um número ISBN errado.

$$x_{10}\ x_9\ x_8\ x_7\ x_6\ x_5\ x_4\ x_3\ x_2 \qquad \underbrace{x_1}_{\text{Quanto vale?}}$$

Os dígitos são tais que $\sum_{i=1}^{10} i\,x_i = 0 \pmod{11}$, isto é,

$$1\times x_1 + 2\times x_2 + 3\times x_3 + \ldots + 10\times x_{10} = 0 \pmod{11}$$

Se $x_1 = 10 \quad\Rightarrow\quad$ substitui-se pelo carácter *X*.

Exemplos: 0-89006-530-6 e 0-13-212713-X

Nota: A partir de 1-1-2007 passaram a ser usados 13 dígitos.

- **<u>Bilhetes de identidade portugueses</u>**

  Têm um número variável de caracteres (*n*). O último carácter à direita (que não faz parte do número!) serve para detectar um número de B. I. errado.

$$\underbrace{x_n\quad x_{n-1}\quad \cdots\quad x_3\quad x_2}_{\text{Nº de B.I.}} \qquad \underbrace{x_1}_{\text{Controlo}}$$

Deverá ser também $\sum_{i=1}^{n} i\,x_i = 0 \pmod{11}$, isto é, se dividirmos $\dfrac{2x_2 + 3x_3 + \ldots nx_n}{11}$, o seu resto somado a x_1 deve dar 11.

Se $x_1 = 10$ então x_1 deveria ser "X" mas alguém ignorante substituiu-o por "0"!!

Exemplos: 2967981 (nº de controlo: 8)

10060579 (nº de controlo: 6)

Exemplos de codificação para detecção de erros

- **NIF** (Número de Identificação Fiscal, ou Número de Contribuinte)

  É um conjunto de 9 caracteres servindo o último para detectar um NIF errado.

$$x_9 \;\; x_8 \;\; x_7 \;\; x_6 \;\; x_5 \;\; x_4 \;\; x_3 \;\; x_2 \qquad \underbrace{x_1}_{\text{Quanto vale?}}$$

  Faz-se como com os bilhetes de identidade: $\sum_{i=1}^{9} i\,x_i = 0 \pmod{11}$, isto é,

$$1 \times x_1 + 2 \times x_2 + 3 \times x_3 + \ldots + 9 \times x_9 = 0 \pmod{11}$$

  Se $x_1 = 10 \quad \Rightarrow \quad$ substitui-se pelo algarismo 0.

  Exemplos: 135796423 e 123456789

- **NIB** (Número de Identificação Bancária)

  Em Portugal tem 21 dígitos: os 4 primeiros representam o banco, os 4 seguintes a agência, os 11 seguintes a conta e os dois últimos os caracteres de controlo.

0123	4567	12345678987	xx
banco	balcão	conta	controlo

- Substituem-se os dois dígitos de controlo (desconhecidos) por 00 (ou seja, multiplica-se o número de 19 algarismos por 100).

- Acha-se o resto da divisão por 97.

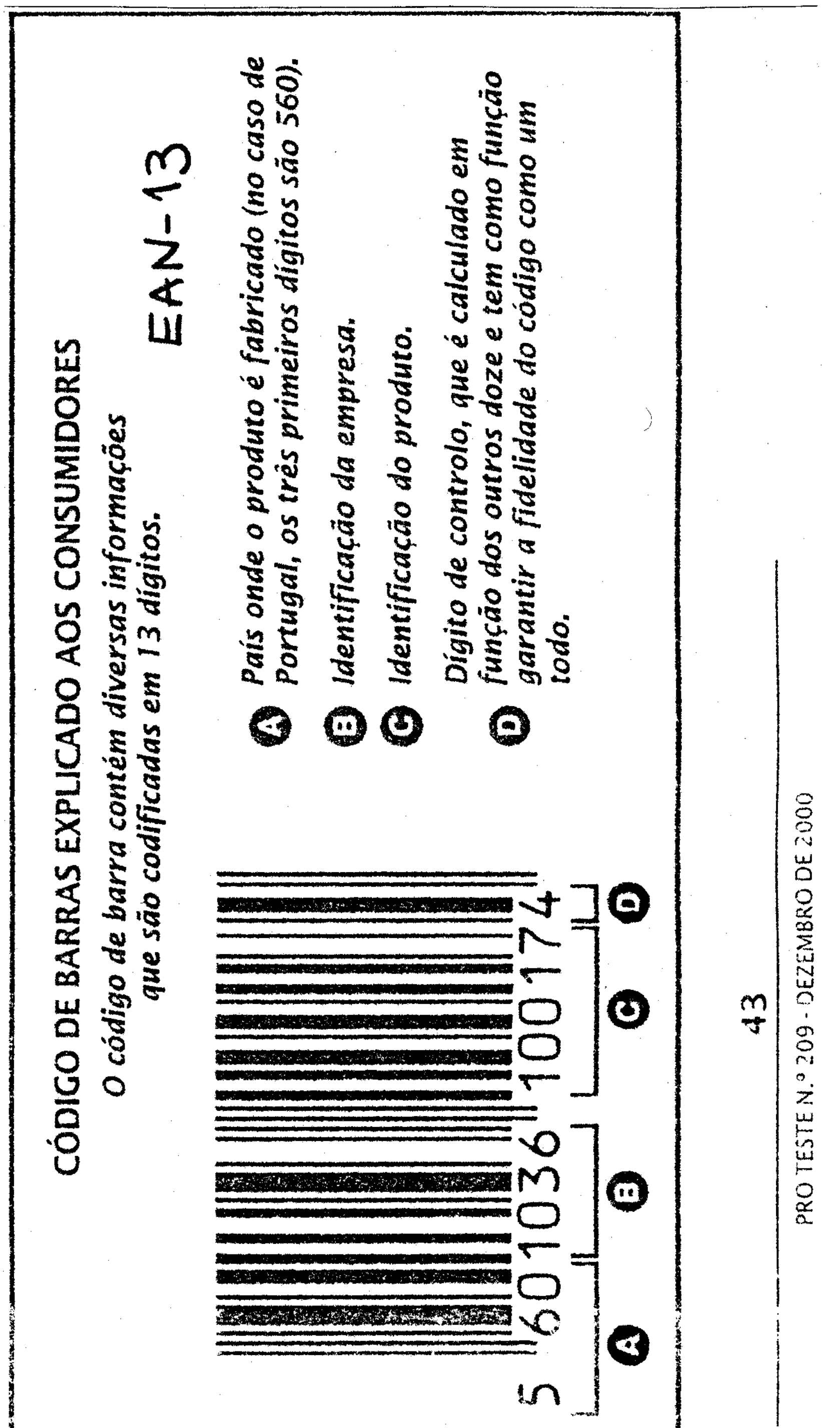
- Subtrai-se o resto a 98 $\Rightarrow$ obtêm-se os dois algarismos de controlo

  - Se a subtracção for ≤ 9 (um só algarismo) acrescenta-se um zero à esquerda.

  Exemplo de cima: $\quad 123456712345678987 00 \bmod 97 = 71 \;\Rightarrow\; xx = 98 - 71 = 27$

Exemplos de codificação para detecção de erros

- **<u>Códigos de barras</u>** (sistema EAN-13)



CÓDIGO DE BARRAS EXPLICADO AOS CONSUMIDORES

O código de barra contém diversas informações que são codificadas em 13 dígitos.

EAN-13

5 601036 100174

A B C D

A *País onde o produto é fabricado (no caso de Portugal, os três primeiros dígitos são 560).*

B *Identificação da empresa.*

C *Identificação do produto.*

D *Dígito de controlo, que é calculado em função dos outros doze e tem como função garantir a fidelidade do código como um todo.*

43

PRO TESTE N.º 209 - DEZEMBRO DE 2000

Exemplos de codificação para detecção de erros

- **<u>Códigos de barras EAN-13</u>** (continuação)

Neste sistema de numeração de artigos o dígito de controlo c é obtido assim:

$$a_{12}a_{11}a_{10} \quad a_9a_8a_7a_6 \quad a_5a_4a_3a_2a_1 \quad c$$

$$\underbrace{\left[3\sum_{i=1}^{6} a_{2i-1} + \sum_{i=1}^{6} a_{2i} + c\right]}_{\text{e' mu'ltiplo de10}} \bmod 10 = 0$$

Ou seja: a soma dos algarismos de ordem par (a contar da direita) é somada ao triplo da soma dos algarismos de ordem ímpar. O resultado, somado a c, deve ser um múltiplo de 10.

<u>Exemplo</u> (um produto de hipermercado)

12	11	10	9	8	7	6	5	4	3	2	1	
5	**6**	**0**	**1**	**3**	**7**	**7**	**3**	**3**	**1**	**0**	**8**	**4**
País			Empresa				Produto					Controlo

Ímpares: (8 + 1 + 3 + 7 + 1 + 6) x 3 = 78

Pares: 0 + 3 + 7 + 3 + 0 + 5 = 18

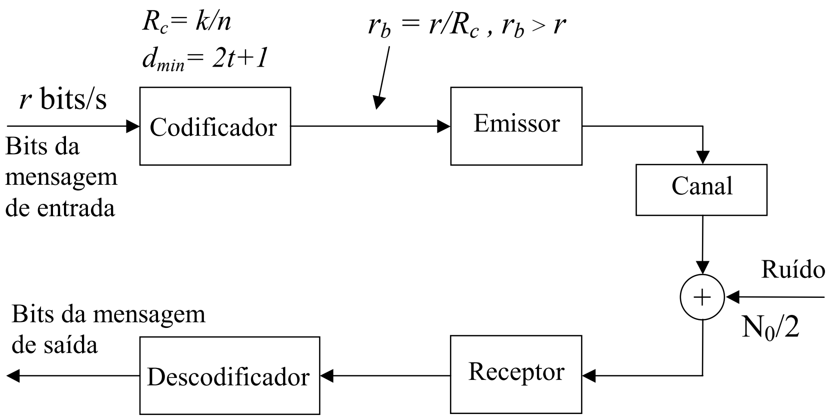
96

$\Rightarrow$ + 4

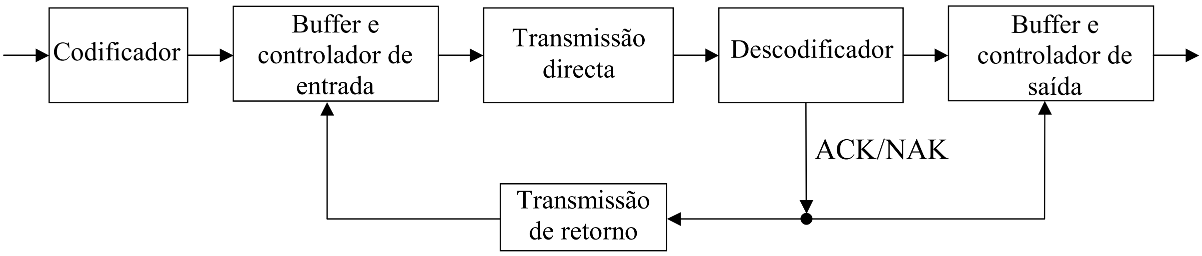
100

Códigos de blocos

FEC



ARQ



ACK (positive <u>ACK</u>nowledgement) — o descodificador não detectou nenhum erro.

NAK (<u>N</u>egative <u>A</u>c<u>K</u>nowledgement) — o descodificador detectou erro.

- ARQ exige menos bits de paridade e a taxa de código $R_c = \dfrac{k}{n}$ é mais elevada que num código FEC (ou seja, a largura de banda ocupada é menor).
- Requer um percurso de retorno e "hardware" para transmissão repetida de palavras de código, o que não acontece nos códigos FEC.
- A taxa de transmissão *directa* deve ter em conta as transmissões repetidas.