Error correcting codes

2SN T+R

Marie-Laure Boucheret

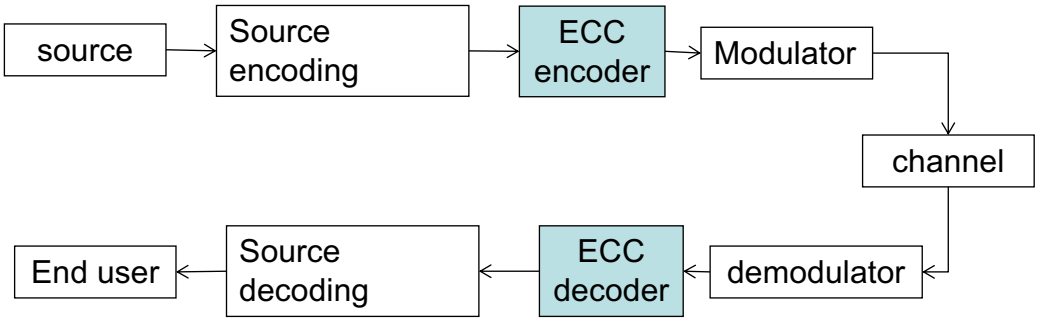
INP-ENSEEIHT

Marie-Laure.Boucheret@enseeiht.fr

Version : 09/2021

Contents

Generalities on coding (1)



Forward Error Correction (FEC) for physical layer

Generalities on coding (2)

Example : (3,1) repetition code Code rate R=1/3

Information bits : 0 1 1 0 1

Coded bits :000 111 111 000 111

2 Code words: 000 and 111

Demodulated bits :010 110 010 000 000

Received words : 8 words (vs 2 codewords)

Detected erroneous words

Undetected erroneous words

Decoded bits : 0 1 0 0 0

Decision rule: Majority bit

Erroneous decoded bits

Repetition code (3,1)

Error detection capacity : maximal number of erors which can be detected in one codeword (here : 2)

Error correction capacity : maximal number of errors which can be corrected in one code word (here : 1)

Repetition code (5,1)

Error detection capacity = ?

Error correction capacity = ?

Code Rate= ? BW increaded by ?

Repetition code (5,1)

Error detection capacity = ?

Error correction capacity = ?

Code Rate= ? BW increaded by ?

Code rate : R=1/5
BW increase by a factor 5 (1/R)

0 => 00000 1 = > 11111 : 2 code words 00000 and 11111

Received words : 5 bits => 32 received words (only 2 are codewords)

Error detection capacity :

Error detected = received word is NOT a codeword

Ex : 4 errors 000000 => 10111 is not a codeword => error detected !

5 errors 00000 => 11111 is a codeword ! error detection capacity is 4

Error correction capacity:

Hyp : 00000 code word

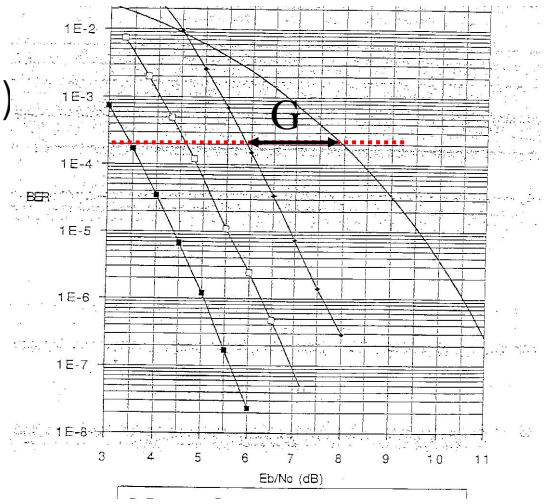
j errors => j « 1 » and 5-j « 0 » 0 is the majority bit if j<=2

error correction capacity is 2

Generalities on coding (4)

Code characteristics:

- coding gain : G (Error correction capacity)
- Spectral efficiency (code rate)
- Complexity (encoding/decoding)



3 code rates:
R=1/2
R=3/4
R=7/8

Generalities on coding (5)

Coding gain (for a given BER):

Difference (in dB) between the required Eb/N0 of uncoded (uc) and coded (c) systems.

In linear : $G=(Eb/N0)_{uc}/(Eb/N0)_c$

$(Eb/N0)_{uc}=P_{uc}Tb/N0$ and $(Eb/N0)_c=P_cTb/N0$

So $G=P_{uc}/P_c$ in dB : $G_{dB}=P_{uc\ dB}-P_{c\ dB}$

Question : What is the coding gain for R=1/2 for the code in the previous slide?

Spectral efficiency

By definition : η_c=Rb/B

Rb : Information bit rate Ru (not coded bit rate Rc !)

B : occupied bandwith (=> related to coded bit rate !)

R : code rate R=k/n=Ru/Rc => Rc=Ru/R

B=k Rs (k: depends on shaping filter, Rs : symbol rate)

Rs=Rc/log2(M) (not Rb!) => B=k Rc /log2(M)=k Ru/R 1/log2(M)

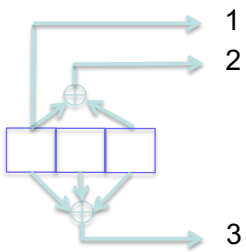
η_c =Ru/B=R log2(M)/k (k:shaping filter, M : modulation, R : ECC)

Recall : spectral efficiency (uncoded system)

η_{uc} =Rb/B=log2(M)/k => η_c = η_{uc} R

Two types of error correcting codes:

- Convolutional codes (with memory)



k_0=1, n_0=3

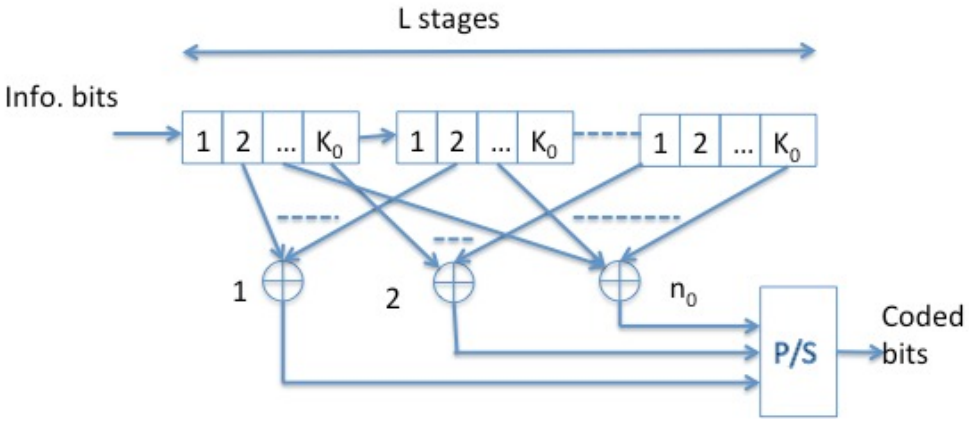
Code rate :R=1/3

Constraint length : L=3

- Block codes (memoriless encoder)

Convolutional Codes

Binary convolutional encoder(1)



Length of information word : k0 bits

Length of coded word : n0 bits

Code rate : R=k0/n0

Constraint length : L stages (=L information words)

Binary convolutional encoder(2)

X(n) : information word at time t=nT

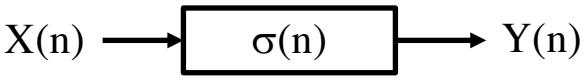
Y(n) : coded word at time t=nT

Content of the L stages : X(n), X(n-1),…,X(n-L+1)

[X(n),X(n-1),…,X(n-L+1)]=[X(n),σ(n)]

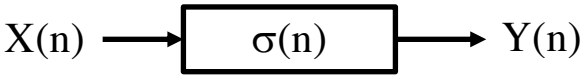
σ(n)=[X(n-1) X(n-2),…,X(n-L+1)] **encoder state** at t=nT

σ(n) : encoder memory

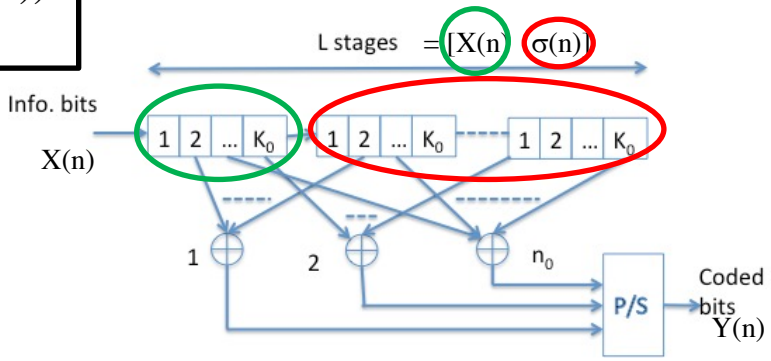


Binary convolutional encoder(3)

σ(n)=[X(n-1) X(n-2),…,X(n-L+1)]

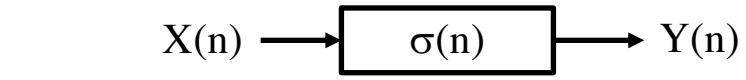


$$Y(n)=f(X(n), \sigma(n))$$
$$\sigma(n+1)=g(X(n), \sigma(n))$$

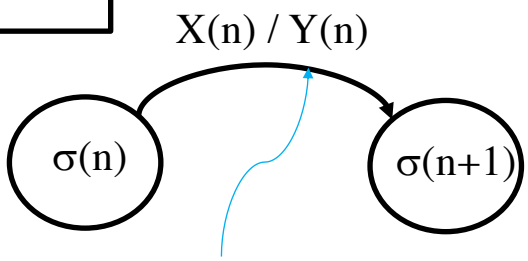


Binary convolutional encoder(4)

$\sigma(n)=[X(n-1)\ X(n-2),\ldots,X(n-L+1)]$



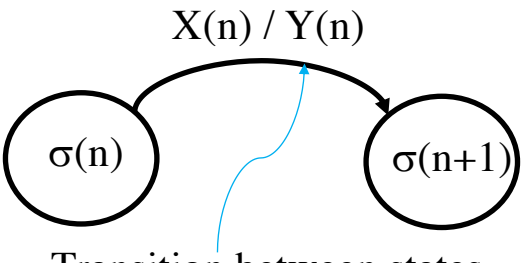
$Y(n)=f(X(n), \sigma(n))$
$\sigma(n+1)=g(X(n), \sigma(n))$



Transition between states

Binary convolutional encoder(5)

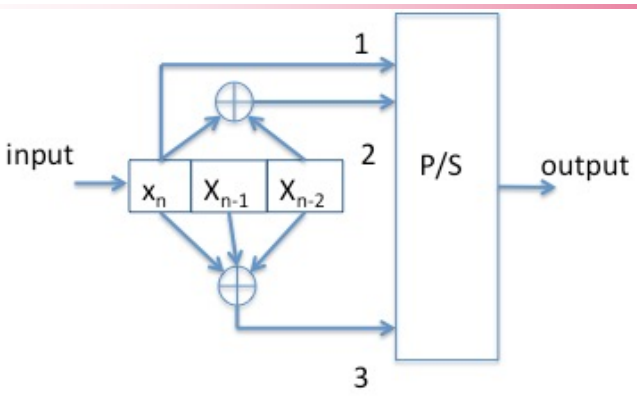
$\sigma(n)=[X(n-1)\ X(n-2),\ldots,X(n-L+1)]$



Transition between states

Number of states : $2^{(L-1)k0}$
Number of outgoing transitions for one state : 2^{k0}
Total number of transition : 2^{Lk0}

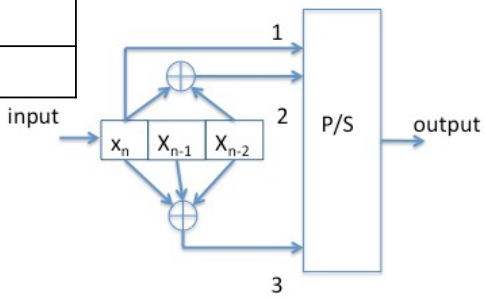
Example 1 (1)



$k_0=1$, $n_0=3$, R=1/3, L=3

σ_n	state
(0,0)	a
(0,1)	b
(1,0)	c
(1,1)	d

Example 1 (2)



σ_n=a

$X_n=0 \Rightarrow Y_n=000, \sigma_n=00=a$

$X_n=1 \Rightarrow Y_n=111, \sigma_n=10=c$

0	0	0

1	0	0

σ_n=c

$X_n=0 \Rightarrow Y_n=001, \sigma_n=01=b$

$X_n=1 \Rightarrow Y_n=110, \sigma_n=11=d$

0	1	0

1	1	0

σ_n=b

$X_n=0 \Rightarrow Y_n=011, \sigma_n=00=a$

$X_n=1 \Rightarrow Y_n=100, \sigma_n=10=c$

0	0	1

1	0	1

σ_n=d

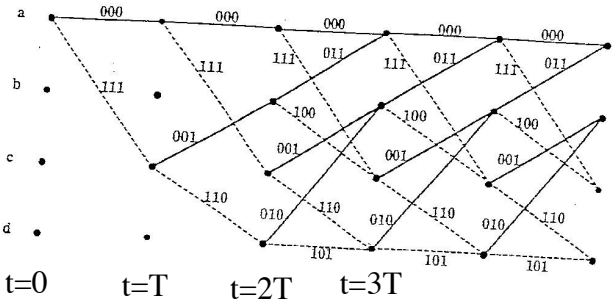
$X_n=0 \Rightarrow Y_n=010, \sigma_n=01=b$

$X_n=1 \Rightarrow Y_n=101, \sigma_n=11=d$

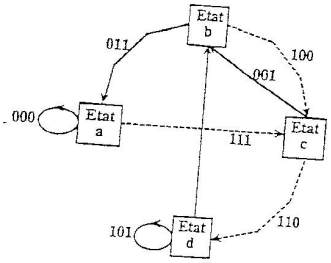
0	1	1

1	1	1

Example 1 (2)



Trellis

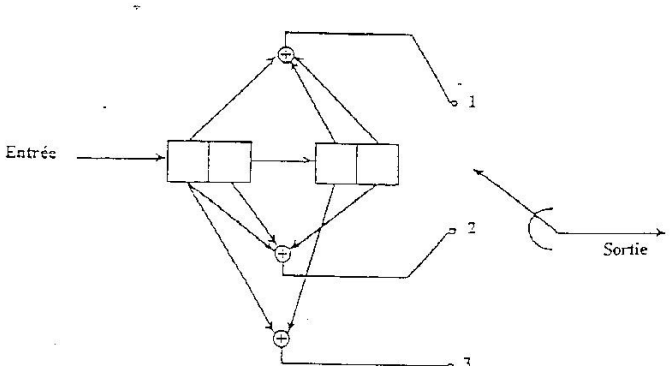


State diagram

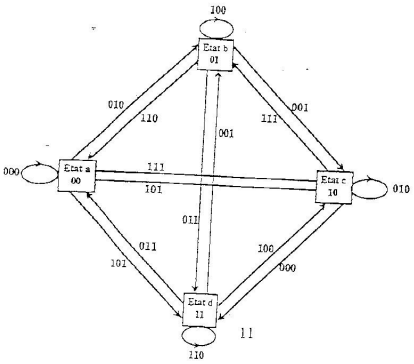
_____ : X_n=0

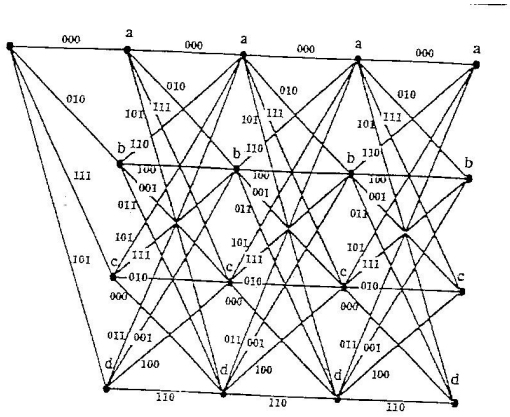
--------- : X_n=1

Example 2 (1)



Example 2 (2)





Free distance (1)

• Recall : Hamming distance

m1=00111 m2=11100 dH(m1,m2)=4

• dfree: free distance

Definition : It is the minimum Hamming distance between two coded sequences

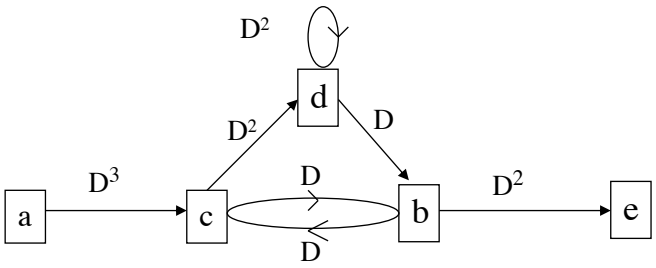
=> give information about the code performances @ high SNR

• For a **linear** code, the null sequence is taken as the reference sequence.

• d_{free} can be be determined with the code **transfer function**

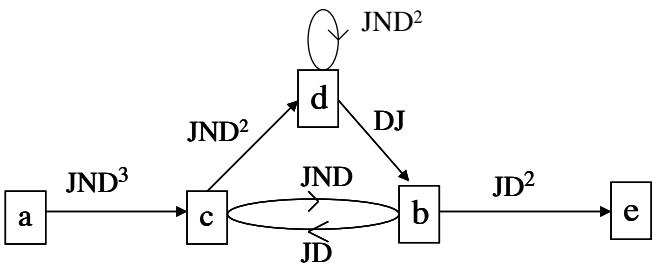
Free distance (2)

Transfert function of convolutional code p9



$$T(D)=\frac{X_e}{X_a}=\frac{D^6}{1-2D^2}=D^6+2D^8+4D^{10}+8D^{12}+...$$

Free distance (3)



N: input weigth

J= number of branches

$$T(D)=\frac{X_e}{X_a}=\frac{J^3ND^6}{1-JND^2(1+J)}=J^3ND^6+J^4N^2D^8+J^5N^3D^8+...$$

Hard Decoding (1)

Maximum likelihood decoding:

Y : received sequence

X: candidate emitted sequence

$\hat{X}$ **:the treillis sequence which minimizes $d_H(Y,X)$**

$\Rightarrow$ Problems : complexity, delay

$\Rightarrow$ Solution : use of the Truncated Viterbi algorithm

Hard Decoding (2)

Show that the estimated code word is the code word which minimizes the hamming distance between the received word and any code word.

Hard Decoding (3)

Example :

Problem :
calculate P(Y/X) where Y=(10010)=($y_4y_3y_2y_1y_0$) and X=(11000)=($x_4x_3x_2x_1x_0$)
p = bit error probability, errors are assumed independant.

Solution :
independant errors => $P(Y/X) = \prod_{i=0}^{4} P(y_i/x_i)$
P(Y/X)= (1-p)p(1-p)p(1-p) (P(0/1)=P(1/0)=p, P(0/0)=P(1/1)=1-p)
$P(Y/X)=p^2(1-p)^3=p^d(1-p)^{5-d}$ where $d=d_H(Y,X)=2$

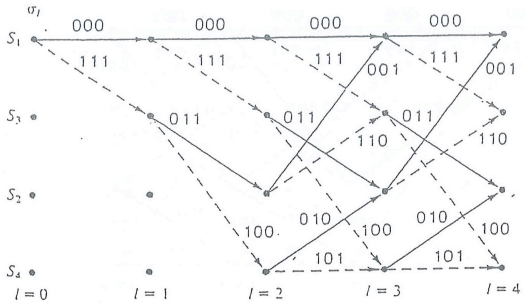
Hard Decoding (4)

Show that the estimated code word is the code word which minimizes the hamming distance between the received word and any code word.

From the previous example we deduce:
$P(Y/X)=p^d(1-p)^{N-d}$ where d=dH(Y,X) and N is length of vectors X and Y

Decoding : find X xhich maximizes P(Y/X)
The only parameter depending on X is d => maximize P(Y/X) w.r.t d
P(Y/X) is a positive value => maximize log(P(Y/X)) w.r.t X
Log(P(Y/X))=-d log((1-p)/p) + N log(1-p)

Hard Decoding (5) : Viterbi Algorithm



The following sequence is received

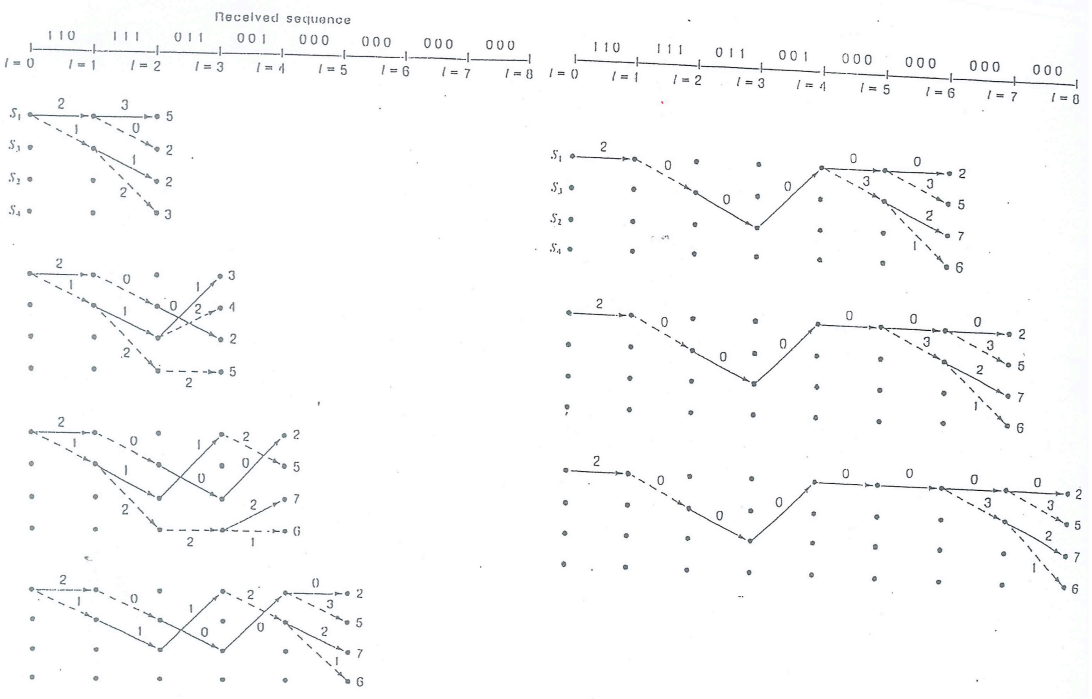
11011101100100000000000

-Determine the survivors

-Find the estimated transmitted sequence

-Find the estimated transitted bits

Hard Decoding (6) : Viterbi Algorithm



Soft Decoding (1)

Y: received sequence of length n (real samples), Y=(y_{n-1},...y_0)

N: noise sequence (n real samples), N=(n_{n-1},...,n_0)

X: candidate emitted codeword (n binary values), X=(x_{n-1},...x_0)

Hypothesis : noise samples are **independant**

$$P(Y/X) = \prod_{i=0}^{n-1} p(y_i/x_i) = \prod_{i=0}^{n-1} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(y_i - x_i)^2\right\}$$

$$P(Y/X) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left\{-\frac{1}{2\sigma^2}\sum_{i=0}^{n-1}(y_i - x_i)^2\right\}$$

$$\text{Maximizing P(Y/X)} \Leftrightarrow \text{Minimizing} \sum_{i=0}^{n-1}(y_i - x_i)^2 = d_E(Y,X)$$

Soft decoding (2)

In the Viterbi algorithm, the Hamming distance is replaced by the Euclidian distance. Coded bits (0/1) are mapped onto (-1/1)

Advantage of soft decoding : better performances in terms of BER

Drawback : complexity

Soft decoding (3)

Nota: complexity can be reduced noting that

$$\sum_{i=0}^{N-1}(y_i - x_i)^2 = \sum_{i=0}^{N-1} y_i^2 + \sum_{i=0}^{N-1} x_i^2 - 2\sum_{i=0}^{N-1} x_i y_i$$

So minimizing the Euclian distance is equivalent to maximazing the metric :

$$\sum_{i=0}^{N-1} x_i y_i$$

R=1/2 convolutional codes, d_{free} max

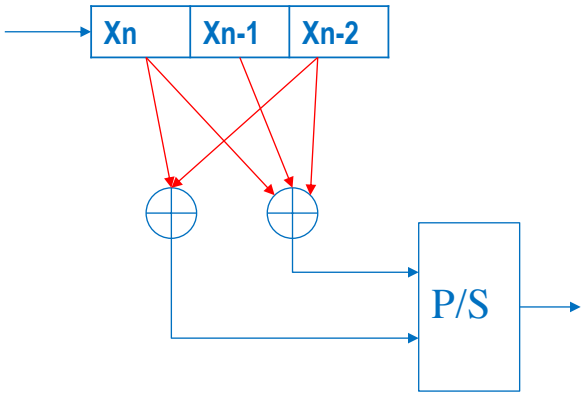
L	Generators in octal		d_{free}
3	5	7	5
4	15	17	6
5	23	35	7
6	53	75	8
7	133	171	10
8	247	371	10
9	561	753	12
10	1167	1545	12
11	2335	3661	14
12	4335	5723	15
13	10533	17661	16
14	21675	27123	16

Error correction capacity (linked to dfee) increases with the number of states => trade off between complexity and power efficiency

(Spectral efficiency unchanged, given by R)

R=1/2 convolutional codes, d_{free} max example 1

Example : R=1/2, L=3, g0=5, g1=7
g0 [octal]=101 g1 [octal]=111



Number of states = ?

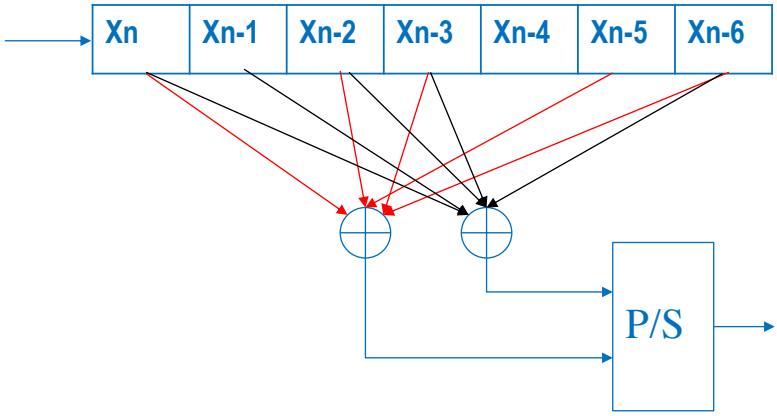
R=1/2 convolutional codes, d_{free} max example 2

Example : R=1/2, L=7, g0=1.3.3, g1=1.7.1
g0 [octal]=001.011.011 g1 [octal]=001.111.001

Number of states = ?

R=1/2 convolutional codes, d_{free} max example 2

Example : R=1/2, L=10, g0=1.3.3, g1=1.7.1
g0 [octal]=001.011.011 g1 [octal]=001.111.001



R=1/3 convolutional codes, d_{free} max

L	Generators in octal			d_{free}
3	5	7	7	8
4	13	15	17	10
5	25	33	37	12
6	47	53	75	13
7	133	145	175	15
8	225	331	367	16
9	557	663	711	18
10	1117	1365	1633	20
11	2353	2671	3175	22
12	4767	5723	6265	24
13	10533	10675	17661	24
14	21645	35661	37133	26

For the same complexity (L) w.r.t R=1/2 error correction capacity is increased => trade-off between spectral efficiancy and power efficiency

R=1/4 convolutional codes, d_{free} max

L	Generators in octal				d_{free}
3	5	7	7	7	10
4	13	15	15	17	13
5	25	27	33	37	16
6	53	67	71	75	18
7	135	135	147	163	20
8	235	275	313	357	22
9	463	535	733	745	24
10	1117	1365	1633	1653	27
11	2387	2353	2671	3175	29
12	4767	5723	6265	7455	32
13	11145	12477	15537	16727	33

Increased power efficiency at the expense of spectral efficiency

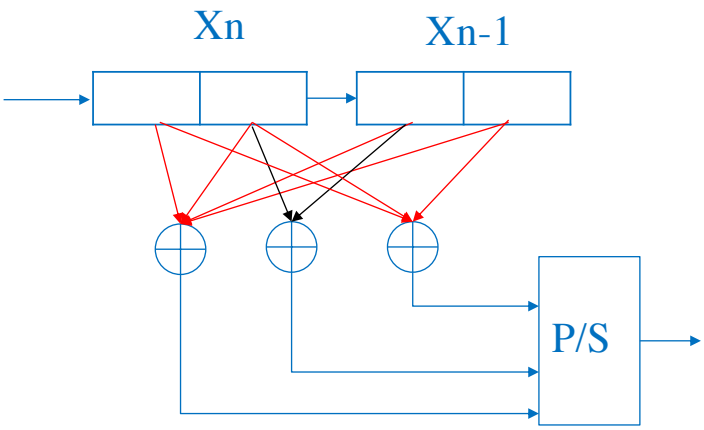
R=2/3 and R=k_0/5 convolutional codes, d_{free} max

L	Générateurs en Octal			d_{free}
2	17	06	15	3
3	27	75	72	5
4	236	155	337	7

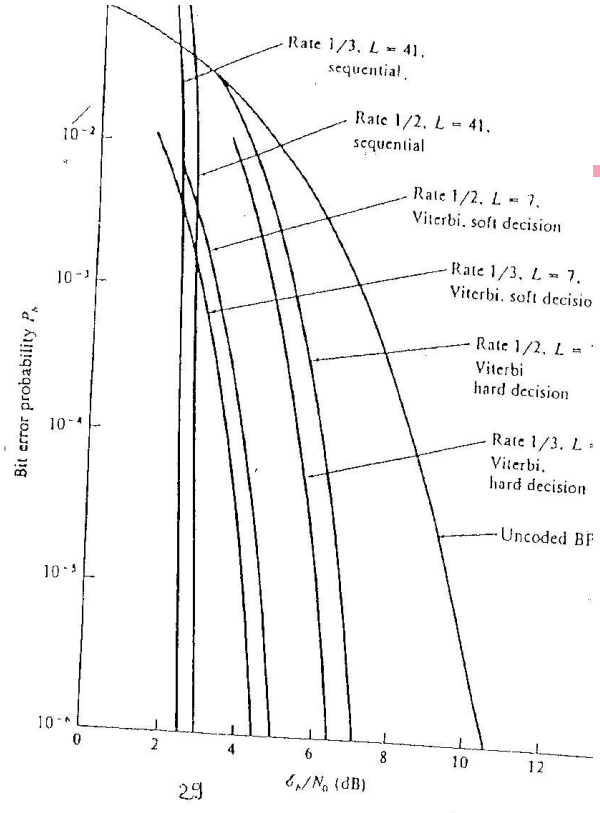
R	L	Générateurs en Octal					d_{free}
2/5	2	17	07	11	12	04	6
2/5	3	27	71	52	65	57	10
2/5	4	247	366	171	266	373	12
3/5	2	35	23	75	61	47	5
4/5	2	237	274	156	255	337	3

R=2/3 convolutional codes, d_{free} max example

Example : R=2/3, L=2, g0=17, g1=06, g2=15
g0 [octal]=001.111 g1 [octal]=000.110 g2 [octal]=001.101



Performances



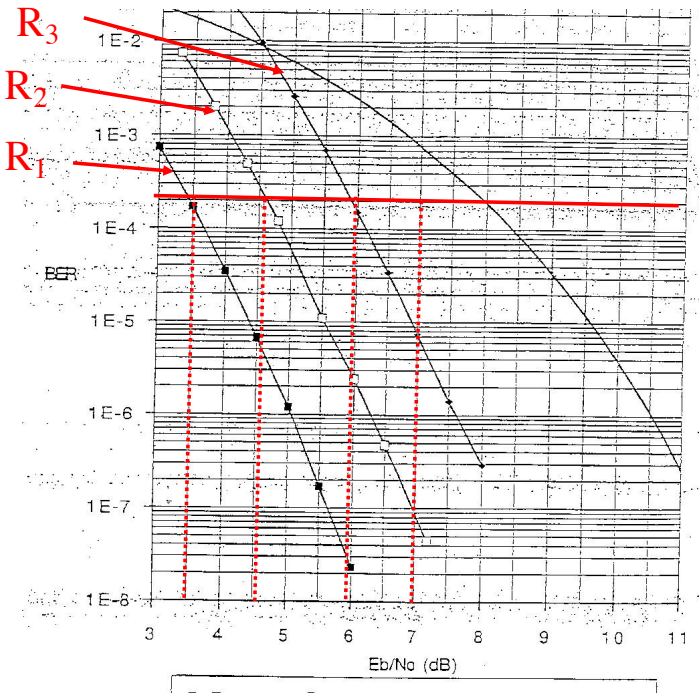
The need for ACM (Adaptive Coding Modulation) (1)

- The received power is in a range [P_{min} P_{max}] (propagation issue, distance variation,…) so the received E_b/N_0 in the range $\left[\frac{E_b}{N_0}_{min}, \frac{E_b}{N_0}_{max}\right]$.
- The specifed Bit Error Rate is BER_0 .

Two strategies:

- Use of a single code (defined from the worst case)
- Use of codes with different code rates (Adaptive Coding)

The need for ACM (Adaptive Coding Modulation) (2)



Example:
$E_b/N_{0\ min}$ = 3.5 dB
$E_b/N_{0\ max}$ = 7 dB
BER_0=2 10^{-4}

$R_1<R_2<R_3$

- 3.5 dB < E_b/N_0 < 4.5 dB code R_1
- 45 dB < E_b/N_0 < 6 dB code R_2
- 6 dB < E_b/N_0 < 7 dB code R_3

ADAPTIVE CODING

The need for ACM (Adaptive Coding Modulation) (1)

ACM : Both adaptive coding and adaptive modulation

Two strategies for adaptive coding :

- Constant useful bit rate (=information rate) :

Increasing R => decreasing occupied BW

- Constant occupied BW

Increasing R => increasing usefil bit rate (adopted solution for telecommunication standarts)

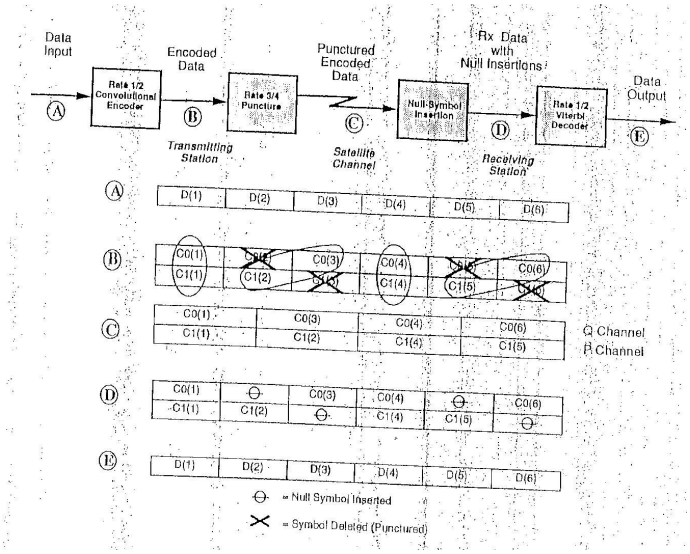
Exercise : show the above implications

Adaptive coding : case of convolutional codes

How to generate convolutional codes with different code rates ?

- Use different optimal encoders listed in tables
- Use punctured codes from a mother code

Puncturing (1)

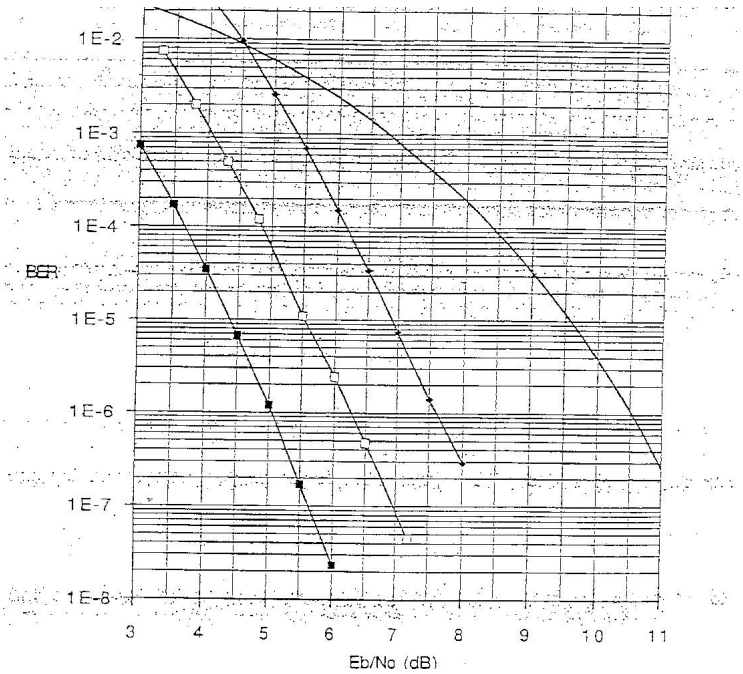


Generating a R=3/4 punctured code from a mother R=1/2 code

Puncturing (2)

	R1: 1
2/3	R0: 10 R1: 11
3/4	R0: 101 R1: 110
4/5	R0: 1000 R1: 1111
5/6	R0: 10101 R1: 11010
6/7	R0: 1000101 R1: 1110101
7/8	R0: 1000101 R1: 1111010
11/12	R0: 10001000001 R1: 11110111110
12/13	R0: 100000001010 R1: 111111110101
15/16	R0: 100110100101101 R1: 111001011010010
16/17	R0: 1010101101111010 R1: 1101010010000101

Puncturing (3)



Code rates :
½, ¾, 7/8

Block Codes

(n,k) linear block codes

- Main characteristic : memoriless encoder (≠ convolutional codes)
- k information symbols are coded into a n symbols long word (= code word)
- Code rate : R=k/n
- spectral efficiency: $\eta_{coded} = R\eta_{uncoded}$

Examples (1)

Example 1 : (3,1) repetition code

x1=u1, x2=u1, x3=u1
u=(u1) information word
x=(x1 x2 x3) coded word

0		000
	<=>	
1		111

2 received words out of 8 possible received words are code words

d_{min}=3 error detection capacity dmin-1=2
Error correction capacity = int ((dmin-1)/2)=1

Examples (2)

Example 2 : (3,2) parity code
x1=u1, x2=u2, x3=u1+u2
u=(u1 u2)
x=(x1 x2 x3)

00		000
01	<=>	011
10		101
11		110

4 code words out of 8 possible received words

Systematic code
x1,x2 : information bits
(=systematic bits)

Bit parity : x3

d_{min}=2 error detection capacity dmin-1=1
Error correction capacity = int ((dmin-1)/2)=0

+ : addition modulo 2 (XOR)

Number of code words : 16
Number of received words : 2^7=128

Examples (3)

Example 3 : (7,4) Hamming codes : systematic code
xi=ui i=1,2,3,4 : systematic bits (=information bits)
3 parity bits :
x5=u1+u2+u3
x6=u2+u3+u4
x7=u1+u2+u4

The Generator Matrix is defined by: G=(I_4 P) I_4 : identity matrix

$$x = uG \qquad G = \begin{bmatrix} 1000101 \\ 0100111 \\ 0010110 \\ 0001011 \end{bmatrix}$$

Detection and error correction capabilities of block codes

• Recall : Hamming distance
m1=00111 m2=11100 dH(m1,m2)=4

d_{min} : minimum Hamming distance between 2 codewords (minimum distance of the code)

⇒ A (N,K,d_{min}) linear code can detect up to t errors with $t \leq dmin-1$

⇒ A (N,K,d_{min}) linear code can correct up to t' errors with $t \leq int(d_{min}-1)/2$ where int () is the integer part (demo fellows).

Decoding (1)

- **Maximum likelihood decoding**

Y: received word

X: emitted code word

$\tilde{X}$: candidate code word

$\hat{X}$: estimated code word

$\hat{X}$ is the code word $\tilde{X}$ such that $d_H(\tilde{X}, Y)$ is minimum.

Direct decoding is too complex for large values of k !

Ex : binary (n,k) code => 2^k code words=> 2^k distances to calculate!

Decoding (2)

$\hat{X} = \tilde{X}$ which maximizes $P(Y/\tilde{X})$

ex: $Y = 01011$ $\tilde{X} = 11001$

p = Bit Error Rate

$P(0/1) = P(1/0) = p$ $P(0/0) = P(1/1) = 1-p$

$P(Y/\tilde{X}) = P(0/1)P(1/1)P(0/0)P(1/0)P(1/1)$

$= p^2(1-p)^3$

In the general case

$P(Y/\tilde{X}) = p^d \cdot (1-p)^{N-d}$

$d = d_H(Y, \tilde{X})$, N = length of Y, $\tilde{X}$

$\ln(P(Y/\tilde{X})) = -d \ln\left(\frac{1-p}{p}\right) + N \ln(1-p)$

$\frac{1-p}{p} > 1$ for $p < 0.5 \Rightarrow \ln\left(\frac{1-p}{p}\right) > 0$

So $P(Y/\tilde{X})$ is maximum when $d = d_H(Y, \tilde{X})$ is minimum.

Decoding (3)

Exercise: Decoding rule for a repetition code (N,1) with N a odd number (information bit is repeated N times)

a) What is the decoding rule in this case ?

b) Is the rule of a) valid for a code which is not a repetition code ?

Standart Array (1)

Decoding complexity can be reduced using standart array.

Let y=x+e with x: emitted code word ((n,k) linear block code), y : received word, e: error word

Instead of directly estimating x, e is estimated ($\hat{e}$) and the estimate of x is given by : $\hat{x}$=y-$\hat{e}$

The parity matrix H (size : n-k,k) is defined such that : $GH^t=0$ where G is the generator matrix (size : (k,n))

Remark : Parity matrix is easy to determine for a systematic code

$$G = [I_k \quad P] \Rightarrow H = [-P^T \quad I_{n-k}]$$

Standart Array (2)

Recall : x=uG where u is the information word (k bits)

For all code words : $xH^t=uGH^t=0$ => $S=yH^t=xH^t+eH^t=eH^t$

S is called a **syndrom**, S is a (1,n-k) vector taking 2^{n-k} values

Problem : estimate e from $S=yH^t=eH^t$

e can take 2^n values, S can take 2^{n-k} values …

Standart Array (3)

Example : S=0 => associated error vectors : all the code words (2^k)

The 2^n error vectors can be partitionned into 2^{n-k} sets of 2^k vectors. Each set is associated with a distinct values of S.

Problem : 2^k error vectors give the same syndrom. What is the estimated error vector among them ?

Standart Array (4)

Recall : the estimated x is the x which minimzes $d_H(y,x)$

y=x+e => $d_H(y,x)=w(e)$ => $\hat{e}$: vector e with **minimum weigth** such that $S=eH^t$

Construction of standart array:

- Generate error vectors of **increasing weigth** w and determine associated syndroms
- Stop when less than $\binom{n}{w+1}$ syndroms are left

k

ENSEEIHT

Cyclic codes

Cyclic codes : generalities

A cyclic code is a linear code such that if $c=(c_0,c_1,\ldots,c_{N-1})$ is a code word then $c=(c_{N-1},c_0,\ldots,c_{N-2})$ is a code word

Cyclic codes are very popular because:

- They can be designed for correcting a given number of symbol errors (with symbols belonging to GF(2^n)
- Decoding algorithms exist with reasonable complexity
- They have good performances (trade-off spectral efficiency / coding gain)

Popular cyclic codes : Binary BCH, (Bon-binary) Reed Solomon

Galois fields (GF)

Field with a **finite number** q of elements are called **Galois fields** and noted GF(q) (F_q).

The set of integers Iq=0,1,…,q-1 with the "classical" + and x (modulo q) is a field when q is a prime number.

Examples:

GF(2)={0,1}, + ="or", x="and"

In GF(3), 2+2=1 (4 mod 3), 2x2=1

Order of an element β:

The smallest integer e such that $\beta^e=1$ is called "the order of b"

Primitive element α :

All element of GF(q) (exept 0) can be expressed as a power of an element of the field α called primitive element.

Order of α = q-1

Ex: in GF(3), 2 is a primitive element ($2^2=1$) $1=2^0$, $2=2^1$

Exercise (1)

Build addition and multiplication tables for GF(2) and GF(3)

Exercise (2)

GF(2)={0,1}

+	0	1
0	0	1
1	1	0

x	0	1
0	0	0
1	0	1

$1+1=0$ $-1=1$ $1^{-1}=1$ because $1x1=1$

GF(3)={0,1,2}

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

X	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

$2+1=0$ $-1=2$ $-2=1$
$1^{-1}=1$ $2^{-1}=2$
Order of α (primitive element) : 2 => $\alpha^2=1 \Rightarrow \alpha=2$ ($2x2=1$)

Galois fields (GF): extension fied (1)

Let GF(q) be a finite field. Let $N=q^m$

Ex: q=2 GF(2) N=16

$GF(q^m)$={m-dimensional vectors with components GF(q)}
Notation: $\beta=(\beta_{m-1},\beta_{m-2},\ldots,\beta_1,\beta_0)$ or $\beta(x)=\beta_0+\beta_1x+\beta_2x^2+\ldots+\beta_{m-1}x^{m-1}$

Addition rule : addition on GF(q) on a component-by-component basis

Multiplication rule :
Let p(x) be an irreducible polynomial of degree m over GF(q) such that p(x) divides $x^{N-1}-1$ => $x^{N-1}-1=p(x)h(x)$ => $p(x)h(x)=0$ modulo p(x)
$\beta(x)\gamma(x)=[\beta(x)\gamma(x)]$ modulo p(x)

Nota: $x^{N-1}-1=0$ modulo p(x) => $x^{N-1}=1$ modulo p(x)
$\Rightarrow$ Order of x N-1 => x is the primitive element of $GF(q^m)$ ($\alpha=x$)
Note that x=(0…010) in vector form

Galois fields (GF): extension fied (2)

Example : GF(2^2), $\alpha^3=1$ Nota:$x^3+1=(x+1)(x^2+x+1)$ (recall: -1=+1 in GF(2)!)
Let $p(x)=x^2+x+1$

Element (1)	Element (2)	Vector	Polynomial
0	0	00	0
1	1	01	1
2	α	10	X
3	α^2	11	1+X

$GF(4)=\{0,1,\alpha,\alpha^2\}$
$\alpha^3=1$

Ex : $2.3=\alpha.\alpha^2=1$, 3-1=2, 2+3=(10)+(11)=(01)=1, -2=2

Nota : p(x)=0 modulo p(x) => $x^2+x+1=0$ => $x^2=-x-1=x+1$

Galois fields (GF): extension fied (3)

Exercice : Building GF(8) with $p(x)=x^3+x+1$

a) Check that p(x) divides x^7-1
b) Deduce α=x (polynomial form) α=(010) (vector form)
c) Recall : $GF(8)=\{0,1,\alpha,\alpha^2,\alpha^3,\alpha^4,\alpha^5,\alpha^6\}$ $\alpha^7=1$

Element number	Power of a and 0	Vector	Polynomial
0	0	000	0
1	1	001	1
2	α	010	X
…	…	…	…
	α^6		

Fill the table (solution p28)
Check that $\alpha^7=1$
2+6=? -3=? 4-5=?
4^{-1}=? 6x7=? 4/5=?

Galois fields (GF): extension fied (3.1)

$p(x)=x^3+x+1$ $p(x)=0$ $[p(x)] \Rightarrow x^3+x+1=0$ $[p(x)] \Rightarrow x^3=x+1$

Element number	Power of α and 0	Vector	Polynomial
0	0	000	0
1	1	001	1
2	α	010	X
4	α^2	100	X^2
3	α^3	011	X+1
6	α^4	110	X^2+X
7	α^5	111	X^2+X+1
5	α^6	101	X^2+1

$\alpha^3=X^2*X$ (p(x))
$=X+1$
$\alpha^4=(X+1)*X$ (p(x))
$=X^2+X$
$\alpha^5=(X^2+X)*X$ (p(x))
$=X^3+X^2$ (p(x)
$= X^2+X+1$
$\alpha^6=(X^2+X+1)*X$
(p(x))
$=X^3+X^2+X$ (p(x)
$= X+1+ X^2+X$
$= 1+ X^2$
$\alpha^7=(X^2+1)*X$ (p(x))
$= X^3+X$ (p(x))
$=X+1+X=1$

$\alpha^7=\alpha^6\alpha=(1+x^2)x$ (modulo p(x)) $=x+x^3$ (modulo p(x)) $=1$

X^7+1 | X^3+X+1

$X^7+X^5+X^4$ | $X^4 +X^2 +X +1$

X^5+X^4+1
$X^5+X^3+X^2$

$X^4+X^3+X^2+1$
X^4+X^2+X

X^3+X+1
X^3+X+1

0

X^7+1
$=(X^3+X+1)(X^4+X^2+X+1)$

$X^7+1=0$ modulo p(x)
$X^7=-1=1 \Rightarrow$ ordre of
X=(010) is 7 => $X=\alpha$

Galois fields (GF): extension fied (4)

TABLE 2.12 Vectors for $GF(2^3)$: $p(x) = x^3 + x + 1$.

Zero and Powers of α	Vector over $GF(2)$
0	000
1	001
α	010
α^2	100
α^3	011
α^4	110
α^5	111
α^6	101

TABLE 2.13 Vectors $GF(2^4)$: $p(x) = x^4 + x + 1$.

Zero and Powers of α	Vector over $GF(2)$
0	0000
1	0001
α	0010
α^2	0100
α^3	1000
α^4	0011
α^5	0110
α^6	1100
α^7	1011
α^8	0101
α^9	1010
α^{10}	0111
α^{11}	1110
α^{12}	1111
α^{13}	1101
α^{14}	1001

$X^4=X+1$ in GF(16)

$\alpha^4=X+1=(0011)$

Galois fields (GF): extension fied (5)

Vectors for $GF(2^6)$: $p(x) = x^6 + x + 1$.

Zero and Powers of α	Vectors over $GF(2)$	Zero and Powers of α	Vectors over $GF(2)$
0	000000	α^{31}	101001
1	100000	α^{32}	100100
α	010000	α^{33}	010010
α^2	001000	α^{34}	001001
α^3	000100	α^{35}	110100
α^4	000010	α^{36}	011010
α^5	000001	α^{37}	001101
α^6	110000	α^{38}	110110
α^7	011000	α^{39}	011011
α^8	001100	α^{40}	111101
α^9	000110	α^{41}	101110
α^{10}	000011	α^{42}	010111
α^{11}	110001	α^{43}	111011
α^{12}	101000	α^{44}	101101
α^{13}	010100	α^{45}	100110
α^{14}	001010	α^{46}	010011
α^{15}	000101	α^{47}	111001
α^{16}	110010	α^{48}	101100
α^{17}	011001	α^{49}	010110
α^{18}	111100	α^{50}	001011
α^{19}	011110	α^{51}	110101
α^{20}	001111	α^{52}	101010
α^{21}	110111	α^{53}	100101
α^{22}	101011	α^{54}	111010
α^{23}	100101	α^{55}	011101
α^{24}	100010	α^{56}	111110
α^{25}	010001	α^{57}	011111
α^{26}	111000	α^{58}	111111
α^{27}	011100	α^{59}	101111
α^{28}	001110	α^{60}	100111
α^{29}	000111	α^{61}	100011
α^{30}	110011	α^{62}	100001

Galois fields (GF): extension fied (6)

Exercise:

Solve the following set of equations in GF(8) :

$\alpha X+Y=1$ (1)

$X+Y=\alpha^3$ (2)

Galois fields (GF): extension fied (7)

(1)-(2)=> $(\alpha+1)X=\alpha^3+1 \Rightarrow \alpha^3X=\alpha \Rightarrow X=\alpha^{-2} \Rightarrow X=\alpha^5$

(2) => $Y=\alpha^3-X=\alpha^3+\alpha^5 \Rightarrow Y=\alpha^2$

$\alpha+1=(010+(001)=(011)=\alpha^3$ $\alpha^3+1=(011)+(001)=(010)=\alpha$

$\alpha^{-2}=\alpha^5$ ($\alpha 7=1$)

$\alpha^3+\alpha^5=(011)+(111)=(100)=\alpha^2$

Definition of cyclic codes

Definition :

A linear code C defined on GF(q) (c_i belongs to GF(q)) is a cyclic code if and only if $\forall\ c \in C, \beta \in GF(q)$ $c=(c_{N-1}, c_{N-2} \ldots c_0)$ then $c'=(\beta c_{N-2}, \beta c_{N-3}, \ldots, \beta c_0, \beta c_{N-1}) \in C$.

To each code word $c=(c_{N-1}, c_{N-2} \ldots c_0)$ the polynomial c(x) is associated :

$c(x)=c_0+c_1x+\ldots+c_{N-1}x^{N-1}$

$c'(x)=\beta\{c_{n-1}+c_0x+\ldots+c_{N-2}x^{N-1}\}$

Degree of c(x) : N-1 (N symbols)

Properties of cyclic codes

Definition : generator polynomial g(x)

Unitary polynomial with minimum degree belonging to C. This degree is noted N-k

$g(x)=g_0+g_1x+g_2x^2+\ldots+g_{N-k-1}x^{N-k-1}+x^{N-k}$

Property 1: for a cyclic code, g(x) divides X^N-1 (on GF(q))

Property 2: Each g(x) dividing X^N-1 is associated to a cyclic code.

Property 3 : let $c(x) \epsilon C$, g(x) divides c(x) or in another way:

C={c(x)=g(x)i(x)} where degree of i(x) less or equal to k-1 (c(x) :degree less or equal to N-1, g(x) : degree : N-k)

i(x) : k coefficients on GF(q) => q^k polynomials i(x) (and so for c(x))

In general, C is not a systematic code.

Number of code words : q^k

Generator polynomial (1)

Ex: n=9 binary cyclic codes (=on GF(2))
$X^9+1=(1+X)(1+X+X^2)(1+X^3+X^6)$ on GF(2)

Exercise :

1) Find all the cyclic codes of length 9. Determine the number of code words and the code rate for each code.
2) Show that the (9,1) code is a repetition code over GF(2)

Generator polynomial (2)

Solution:

$X^9+1=(1+X)(1+X+X^2)(1+X^3+X^6)$
g(x)=1+x => (9,8) cyclic code R=8/9 degree of g(x) =n-k 2^8 code words
$g(x)= 1+X+X^2$ => (9,7) cyclic code R=7/9 2^7 code words
$g(x)= 1+X^3+X^6$ => (9,3) cycle code R=3/9 2^3 code words
g(x)= (1+X)(1+X+X²) = > (9,6) cyclic code R=6/9 2^6 code words
$g(x) =(1+X)(1+X^3+X^6)$ => (9,2) cyclic code R=2/9 4 code words
$g(x) =(1+X+X^2)(1+X^3+X^6)$ => (9,1) cyclic code R=1/9 2 code words

$g(x) =(1+X+X^2)(1+X^3+X^6)=1+X+X^2+X^3+X^4+X^5+X^6+X^7+X^8$
Degree of g(x): 8 degree of c(x): 8 => degree of i(x) : 0 => i(x) is a constant i_0
$i(x)=i_0$ => $c(x)=i(x)g(x)=i_0g(x)$ => $c=(i_0\ i_0\ i_0\ i_0\ i_0\ i_0\ i_0\ i_0\ i_0)$
$\Rightarrow$ Repetition code 0 $\Leftrightarrow$ 000000000 1$\Leftrightarrow$111111111
$\Rightarrow$ 2 code words (000000000) and (111111111)

Systematic/ Non systematic cyclic codes (1)

Non systematic cyclic code:

i(x) : information polynomial (maximum degree : k-1= k symbols)

c(x)=g(x)i(x) (coefficients of c(x) are convolutions of coefficients of g(x) and i(x))

Systematic cyclic code: (N=n)

$c(x)=x^{N-k}i(x)+t(x)$ with $t(x)=-R_{g(x)}[x^{N-k}i(x)]$

t(x): remainder of the Euclidian division of $x^{N-k}i(x)$ by g(x)

Systematic/ Non systematic cyclic codes (2)

Systematic cyclic code:

Consider : $x^{N-k}i(x)$ max. degree i(x) : k-1 degree g(x) = N-k

$x^{N-k}i(x)$ degree min : N-k degree max : N-k+k-1= N-1

Let $c(x)= x^{N-k}i(x)+t(x)$ with max degree of t(x)=N-k-1

Find t(x) such that g(x) divides c(x) (cyclic code)

$R_{g(x)}[x^{N-k}i(x)+t(x)]=0= R_{g(x)}[x^{N-k}i(x)]+t(x)$

$R_{g(x)}[t(x)]=t(x)$ because degree t(x) < degree g(x)

$\Rightarrow$ t(x): remainder of the Euclidian division of $x^{N-k}i(x)$ by g(x)

Systematic/ Non systematic cyclic codes (3)

Example of systematic encoding:

(9,7) binary cyclic code with $g(x)=x^2+x+1$

c(x) : degree 8, i(x):degree 6, n=9, k=7

i:(0000010) => i(x)=x => $x^{n-k}i(x)= x^2\, i(x)=x^3$

x^3	x^2+x+1
x^3+x^2+x	$x+1$
x^2+x	
x^2+x+1	
1	

Systematic bits

$t(x)=-R_{g(x)}[x^{N-k}i(x)]=-1=1$ t(x)

Parity bits

$c(x)=x^3+1$ c:(000001001)

Remark: $c_{NS}(x)=x^3+x^2+x$ (NS)

$c_{NS}(x)= i(x)g(x)=(000001110)$

Exercise

Exercice 1:

Let C be a binary code of length 6 and generator polynomial $g(x) = x^2 + x + 1$.

1. Show that C is a cyclic code

2. What is the number of code words?

3. Let i(x) be the polynomial associated to the information word i. We consider i(x)=x+1.

Give the binary (vector) form of i

Determine the coded word in the two following cases: systematic and non systematic codes. The coded words will be given in the polynomial form and in the binary (vector) form.

Choice of a generator polynomial (1)

$x^N-1=f_1(x)f_2(x)\ldots f_s(x)$ on GF(q) (code on GF(q))

=> There are 2^s-2 possible polynomials (g(x)=1 and $g(x)=x^N-1$ discarded).

=> How to select g(x) in order to have a "good" code with a given error correction capacity?
(a good code = a code with good trade-off minimum distance/code rate)

... The search for good codes

Definition : Primitive cyclic code
A code defined on GF(q) is primitive if the code length $N=q^m-1$
Ex: binary codes (q=2) => $N = 2^m-1$

Definition : minimal polynomial
Let $\beta_j \in GF(q^m)$
The minimal polynomial $f_j(x)$ of β_j is the polynomial on GF(q) with minimum degree having β_j as a root.

Choice of a generator polynomial (2)

Analogy : real and complex fields (of course not finite) $C=R^2$

Minimal polynomial of c=a (a: real number) : f(x)=x-a

Minimal polynomial of j : $f(x)=(x-j)(x+j)=x^2+1$ ($j^2=-1$)

Minimal polynomial of c=a+jb (b≠0) : $f(x)=(x-c)(x-c^*)=x^2-2ax+(a^2+b^2)$

Definition : set of conjugates of β in $GF(q^m)$: $\left\{\beta^q, \beta^{q^2}, \beta^{q^3}, \ldots, \beta^{q^r}, \ldots\right\}$

Nota : this set has a finite number of distinct elements !

Property : β + all its conjugates are the roots of the minimal polynomial of β (cf analogy with real/complex fields)

Exercise : find the minimal polynomial of α , the primitive element of GF(8)

Choice of a generator polynomial (3)

Exercise : minimal polynomial f(x) of α , the primitive element of GF(8)

Recall : $\alpha^7=1$

Conjugates of α : α^2,α^4 (stop at α^4 because $\alpha^8=\alpha$)

Roots of f(x) : α,α^2,α^4

$f(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^4)=(x^2+(\alpha+\alpha^2)x+\alpha^3)(x+\alpha^4)$

$f(x)=x^3+(\alpha^4+\alpha+\alpha^2)x^2+(\alpha^5+\alpha^{6+}\alpha^3)x+1=x^3+x+1$

Coefficients of f(x) belong to GF(2) : OK

TABLE 2.12 Vectors for $GF(2^3)$: $p(x) = x^3 + x + 1$.

Zero and Powers of α	Vector over $GF(2)$
0	000
1	001
α	010
α^2	100
α^3	011
α^4	110
α^5	111
α^6	101

Exercises

Exercise : minimal polynomial $f_2(x)$ of $\beta=\alpha^2$, the primitive element of GF(8)

Exercise : minimal polynomial $f_3(x)$ of $\beta=\alpha^3$, the primitive element of GF(8)

Exercises : correction (1)

Exercise : minimal polynomial of $\beta=\alpha^2$, α : primitive element of GF(8)

Recall : $\alpha^7=1$

Conjugates of β : $\beta^2=\alpha^4$, $\beta^4=\alpha$ (stop at β^4 because $\beta^8=\alpha^2=\beta$)

Roots of $f_2(x)$: β,β^2,β^4

$f_2(x)=(x-\beta)(x-\beta^2)(x-\beta^4)=)=(x-\alpha^2)(x-\alpha^4)(x-\alpha)$

=> $f_2(x)=f(x)= x^3+x+1$ (minimal polynomial of α)

Important result :

All conjugate elements have the same minimal polynomial

Exercises : correction (2)

Exercise : minimal polynomial of $\beta=\alpha^3$, α : primitive element of GF(8)

α^3 is not a conjugate of α !

Conjugates of β : $\beta^2=\alpha^6$, $\beta^4=\alpha^{12}=\alpha^5$ (stop at β^4 because $\beta^8= \alpha^{10}= \alpha^3 =\beta$)

Roots of $f_3(x)$: β,β^2,β^4

$f_3(x)=(x-\beta)(x-\beta^2)(x-\beta^4)=)=(x-\alpha^3)(x-\alpha^6)(x-\alpha^5)$ (nota: constant term=$\alpha^{14}=1$)

$f_3(x)=(x^2+(\alpha^3+\alpha^6)x+\alpha^2)(x+\alpha^5)=(x^2+ \alpha^4x+\alpha^2)(x+\alpha^5)$

$=x^3+(\alpha^4+ \alpha^4)x2+(\alpha^2+ \alpha^2)X+ \alpha^7$

$=x^3+1$

Choice of a generator polynomial (4)

Let $n=q^m$, $\alpha^{n-1}=1$ => α is a root of $x^{n-1}-1$ (note that **$N=q^m-1$**=n-1)

α is the primitive element of GF(q^m)=GF(n)

Let β be any element of GF(n) exept 0. β can take n-1 values.

β can be expressed as a power of α => $\beta^{n-1}=(\alpha^j)^{n-1}=1$ => β is a root of $x^{n-1}-1$ (n-1 roots)

=> Roots of $x^{n-1}-1$ are the non-null elements of GF(n)

$$x^{n-1}-1=\prod_{k=0}^{n-2}(x-\alpha^k)=\prod_{\beta\in GF(n)/\{0\}}(x-\beta)=\prod_i f_i(x)$$

$x^{n-1}-1$ can be expressed either as a product or monomials over GF(q^m) or a product of minimal polynomials $f_i(x)$ of elements of GF(q^m) belonging to distincts sets of conjugates (these minimal polynomials have coefficients on GF(q))

Choice of a generator polynomial (5)

Example : GF(8)={0,1,α, α^2,…, α^6}

1) Find the set of distinct conjugates (definition of conjugates : p 42)

2) Find the associated minimal polynomials

3) Factorize x^7-1 on GF(8) and GF(2)

Remark :Conjugates of $\beta=\alpha^6$: $\beta^2=\alpha^{12}=\alpha^5$ $\beta^4=\alpha^{10}=\alpha^3$ $\beta^8=\beta^1$

Choice of a generator polynomial (5)

Example : GF(8) correction

1) Find the set of distinct conjugates

$\{\alpha,\alpha^2,\alpha^4\}$ $\{\alpha^3,\alpha^6,\alpha^5\}$ $\{1\}$

2) Find the associated minimal polynomials

$\{1\} \Rightarrow f_0(x)=x-1$ (no conjugates for 1)

$\{\alpha,\alpha^2,\alpha^4\} \Rightarrow f_1(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^4)=x^3+x+1$ (p 44)

$\{\alpha^3,\alpha^6,\alpha^5\} \Rightarrow f_3(x)=(x-\alpha^3)(x-\alpha^6)(x-\alpha^5)= x^3+x^2+1$ (p46)

3) Factorize x^7-1 on GF(8) and GF(2)

$$x^7-1=\prod_{k=0}^{6}(x-\alpha^k)=(x+1)(x^3+x+1)(x^3+x^2+1)$$

Choice of a generator polynomial (6)

g(x) : minimal polynomial of C

C : primitive cyclic code of length $N=q^m-1$ with coefficients on GF(q)

g(x) divides x^N-1 and coefficients of g(x) are on GF(q) (g(x) is a code word).

$\Rightarrow$ g(x) is the product of minimal polynomials of some elements $\{\beta_j\}$ of $GF(q^m)/\{0\}$

$\Rightarrow$ How to select the set $\{\beta_j\}$?

BCH codes (1)

BCH : Bose-Chaudhuri-Hocquenghem (1959)

Definition of primitive BCH codes on GF(q):
$N=q^m-1$ α: primitive element of $GF(q^m)$ fj(x) : minimal polynomial of α^j

$$g(x) = SCM(f_1(x), f_2(x), ..., f_{2t}(x))$$

SCM : smallest common multiple
g(x) : product of distinct minimal polynomials
d=2t+1 : distance *by construction* of the BCH code, $d_{min} \geq d$, t: error correction capacity

Practical evaluation of g(x), code on GF(q), $N=q^m-1$:
1- Construct $GF(q^m)$
2- Find $f_j(x)$, minimal polynomial of α^j for j=1...2t (2t successive powers of α)
3- Calculate g(x) as the product of the DISCTINCT mimimal polynomials.

BCH codes (2)

Example: binary BCH code ($q=2, N=2^m-1$)
BCH(N=15,k=?,t=2) g(x)=? k=?
Note that k is not known before the evaluation of g(x)

m=4 extension field $GF(q^m)=GF(16)$
α : primitive element of GF(16) $\alpha^{15}=1$

BCH codes (3)

BCH(N=15,k=?,t=2)

t=2 => evaluation of minimal polynomials of α, α^2, α^3, α^4 (GF(16))

α, α^2 and α^4 are conjugates on GF(16)=> same minimal polynomials

So : $g(x)=f_\alpha(x)f_{\alpha 3}(x)$

Conjugates of α : α^2, α^4, α^8 ($\alpha^{16}=\alpha$)=> $f_\alpha(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8)$

$f_\alpha(x)=x^4+x+1$

Conjugates of α^3 : α^6, α^{12}, α^{24} = α^9 ($\alpha^{18}=\alpha^3$)

$f_{\alpha 3}(x)=(x-\alpha^3)(x-\alpha^6)(x-\alpha^{12})(x-\alpha^9)= x^4+x^3+x^2+x+1$

Degree of g(x) = 8 =N-k and N=15

N-k=8 => k=7 BCH(N=15,k=7,t=2) , R=7/15 ,number of code words =2^7=128

Nota: the value ok k is unknown before calculation

BCH codes (4)

Peterson table (page 55)

Find the minimal plynomial of α^3 in GF(16)

$\Rightarrow$ Line « Degree 4 » (16=2^4)

$\Rightarrow$ Column : « 3 37D » '3' is for α^3 '37' s the minimal polynomial in octal form 'D' is irrelevant for us

$\Rightarrow$ 3.7=011.111 in binary

$\Rightarrow$ Minimal polynomial = $X^4+X^3+X^2+X+1$ (rightmost term : constant term)

Minimal polynomials (1)

Minimum Polynomials for $GF(2^m)$.

Degree	2	1	7H									
Degree	3	1	13F									
Degree	4	1	23F	3	37D	5	07					
Degree	5	1	45E	3	75G	5	67H					
Degree	6	1	103F	3	127B	5	147H	7	111A	9	015	
11	155E	21	007									
Degree	7	1	211E	3	217E	5	235E	7	367H	9	277E	
11	325G	13	203F	19	313H	21	345G					
Degree	8	1	435E	3	567B	5	763D	7	551E	9	675C	
11	747H	13	455F	15	727D	17	023	19	545E	21	613D	
23	543F	25	433B	27	477B	37	537F	43	703H	45	471A	
51	037	85	007									
Degree	9	1	1021E	3	1131E	5	1461G	7	1231A	9	1423G	
11	1055E	13	1167F	15	1541E	17	1333F	19	1605G	21	1027A	
23	1751E	25	1743H	27	1617H	29	1553H	35	1401C	37	1157F	
39	1715E	41	1563H	43	1713H	45	1175E	51	1725G	53	1225E	
55	1275E	73	0013	75	1773G	77	1511C	83	1425G	85	1267E	
Degree	10	1	2011E	3	2017B	5	2415E	7	3771G	9	2257B	
11	2065A	13	2157F	15	2653B	17	3515G	19	2773F	21	3753D	
23	2033F	25	3443F	27	3573D	29	2461E	31	3043D	33	0075C	
35	3023H	37	3543F	39	2107B	41	2745E	43	2431E	45	3061C	
47	3177H	49	3525G	51	2547B	53	2617F	55	3453D	57	3121C	
59	3471G	69	2701A	71	3323H	73	3507H	75	2437B	77	2413B	
83	3623H	85	2707E	87	2311A	89	2327F	91	3265G	93	3777D	
99	0067	101	2055E	103	3575G	105	3607C	107	3171G	109	2047F	
147	2355A	149	3025G	155	2251A	165	0051	171	3315C	173	3337H	
179	3211G	341	0007									

Minimal polynomials (2)

Degree	11	1	4005E	3	4445E	5	4215E	7	4055E	9	6015G	
11	7413H	13	4143F	15	4563F	17	4053F	19	5023F	21	5623F	
23	4757B	25	4577F	27	6233H	29	6673H	31	7237H	33	7335G	
35	4505E	37	5337F	39	5263F	41	5361E	43	5171E	45	6637H	
47	7173H	49	5711E	51	5221E	53	6307H	55	6211G	57	5747F	
59	4533F	61	4341E	67	6711G	69	6777D	71	7715G	73	6343H	
75	6227H	77	6263H	79	5235E	81	7431G	83	6455G	85	5247F	
87	5265E	89	5343B	91	4767F	93	5607F	99	4603F	101	6561G	
103	7107H	105	7041G	107	4251E	109	5675E	111	4173F	113	4707F	
115	7311C	117	5463F	119	5755E	137	6675G	139	7655G	141	5531E	
147	7243H	149	7621G	151	7161G	153	4731E	155	4451E	157	6557H	
163	7745G	165	7317H	167	5205E	169	4565E	171	6765G	173	7535G	
179	4653F	181	5411E	183	5545E	185	7565G	199	6543H	201	5613F	
203	6013H	205	7647H	211	6507H	213	6037H	215	7363H	217	7201G	
219	7273H	293	7723H	299	4303B	301	5007F	307	7555G	309	4261E	
331	6447H	333	5141E	339	7461G	341	5253F					
Degree	12	1	10123F	3	12133B	5	10115A	7	12153B	9	11765A	
11	15647E	13	12513B	15	13077B	17	16533H	19	16047H	21	10065A	
23	11015E	25	13377B	27	14405A	29	14127H	31	17673H	33	13311A	
35	10377B	37	13565E	39	13321A	41	15541G	43	15053H	45	15173C	
47	15621E	49	17703C	51	10355A	53	15321G	55	10201A	57	12331A	
59	11417E	61	13505E	63	10761A	65	00141	67	13275E	69	16663C	
71	11471E	73	16237E	75	16267D	77	15115C	79	12515E	81	17545C	
83	12255E	85	11673B	87	17361A	89	11271E	91	10011A	93	14755C	
95	17705A	97	17121G	99	17323D	101	14227H	103	12117E	105	13617A	
107	14135G	109	14711G	111	15415C	113	13131E	115	13223A	117	16475C	
119	14315C	121	16521E	123	13475A	133	11433B	135	10571A	137	15437G	
139	12067F	141	13571A	143	12111A	145	16535C	147	17657D	149	12147F	
151	14717F	153	13517B	155	14241C	157	14675G	163	10663F	165	10621A	

Peterson-Weldon tables

Decoding of BCH codes (1)

Peterson-Gorenstein-Zierler Algorithm.

C is a BCH code on GF(q) of length $N=q^m-1$ (primitive code)

g(x) : generator polynomial of C

α: primitive element of the extension field $GF(q^m)$.

c(x) : polynomial associated to code word c (coefficients on GF(q))

v(x) : polynomial associated to received word c (coefficients on GF(q))

e(x) : polynomial associated to error word e (coefficients on GF(q))

e(x) is defined as e(x)=v(x)-c(x) (so v(x)=c(x)+e(x))

Let $e(x) = Y_1 x^{i_1} + Y_2 x^{i_2} + Y_v x^{i_v}$ with $Y_1,\dots Y_\nu$ belonging to GF(q)

ν : the actual number of errors ($\nu \leq t$) , t : error correction capacity of the code

Decoding of BCH codes (1.1)

Recall :

1) g(x) divides c(x) because C is a cyclic codes
2) $\alpha,\alpha^2,\dots,\alpha^{2t}$ are roots of g(x) (BCH code)

1) 2) => $\alpha,\alpha^2,\dots,\alpha^{2t}$ are roots of c(x) $c(x) \in C \Leftrightarrow c(\alpha) = c(\alpha^2) = \dots = c(\alpha^{2t}) = 0$

v(x)=c(x)+e(x) => $v(\alpha\ ) = e(\alpha\ ), v(\alpha^2) = e(\alpha^2),\dots, v(\alpha^{2t}) = e(\alpha^{2t})$

Let's define $S_j=v(\alpha^j)$, Sj only depends on the error vector. It is called a **syndrom**.

Unknowns: ν (actual number of errors), $\{i_1,i_2,\dots,i_\nu\}$: error positions , error amplitudes : $Y_1,\dots,Y_\nu$

=> There are 2ν unknowns $\{i_1,i_2,\dots,i_\nu\}$ $\{Y_1, Y_2, \dots, Y_v\}$

We have a set of 2t syndroms:

$S_j = v(\alpha^j) = \sum_{l=1}^{v} Y_l (\alpha^j)^{i_l} = \sum_{l=1}^{v} Y_l (\alpha^{i_l})^j = \sum_{l=1}^{v} Y_l X_l^j$ with $X_l = \alpha^{i_l}$

Decoding of BCH codes (2)

Error location polynomial:

$$\Lambda(x) = \Lambda_\nu x^\nu + \Lambda_{\nu-1} x^{\nu-1} + ... + \Lambda_1 x + 1$$

$$\Lambda(x) = (1 - xX_1)(1 - xX_2)...(1 - xX_\nu)$$

Coefficients of $\Lambda(x)$ can be found by different ways (PGZ algorithm, Berlekamp-Massey algorithm,…).

One possibility is to solve the following linear system:

$$\begin{bmatrix} S_1 & S_2 & S_3 & ... & S_{\nu-1} & S_\nu \\ S_2 & S_3 & S_4 & ... & S_\nu & S_1 \\ S_3 & S_4 & S_5 & ... & S_{\nu+1} & S_{\nu+2} \\ ... & ... & ... & ... & ... & ... \\ S_\nu & S_{\nu+1} & S_{\nu+2} & ... & S_{2\nu-2} & S_{2\nu-1} \end{bmatrix} \begin{bmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \Lambda_{\nu-2} \\ ... \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{\nu+1} \\ -S_{\nu+2} \\ -S_{\nu+3} \\ ... \\ -S_{2\nu} \end{bmatrix}$$

Remark : ν has first to be evaluated (see after)

{Xi} are obtained by searching the roots of $\Lambda(x)$ on GF(q^m) (Chien search)

Decoding of BCH codes (3)

$$M = \begin{bmatrix} S_1 & S_2 & S_3 & ... & S_{\mu-1} & S_\mu \\ S_2 & S_3 & S_4 & ... & S_\mu & S_1 \\ S_3 & S_4 & S_5 & ... & S_{\mu+1} & S_{\mu+2} \\ ... & ... & ... & ... & ... & ... \\ S_\mu & S_{\mu+1} & S_{\mu+2} & ... & S_{2\mu-2} & S_{2\mu-1} \end{bmatrix}$$

Peterson-Gorenstein-Zierler algorithm

1- Calculate the 2t syndroms and initialize μ=t

2- Search largest μ such that det(M)≠0

3-Calculate the coefficients of $\Lambda(x)$

4-Find the roots of $\Lambda(x)$ (=> $\{X_i^{-1}\}$)

5-Stop for binary codes. Otherwise determine $\{Y_i\}$ by solving

$$S_j = \sum_{l=1}^{\nu} Y_l X_l^j, j = 1,2,..,2t$$

Exercise

We consider the binary BCH code (N=15,k=7) able to correct t=2 erreurs. GF(16) is given at the end of the text.
The received word is: 000001000000000 (the constant term of v(x) is the rigthmost bit)
1) Calculate the syndroms associated to v(x)
2) Show that the estimated number of errors is equal to 1.
Recall : Peterson algorithm, S_i are the syndroms ($S_i=r(\alpha^i)$).
•number of errors ν : largest μ such that det(M) is different from 0.

3) Find the error positions by evaluating $\Lambda(x)$. Roots of $\Lambda(x)$ are the set $\left\{\frac{1}{X_j}=\frac{1}{\alpha^{i_j}}\right\}$ where $\{i_j\}$ is the set of error positions

Recall :

$$\Lambda(x)=\Lambda_\nu x^\nu+\Lambda_{\nu-1}x^{\nu-1}+...+\Lambda_1 x+1$$
$$\Lambda(x)=(1-xX_1)(1-xX_2)...(1-xX_\nu)$$

$$M=\begin{bmatrix} S_1 & S_2 & S_3 & ... & S_{\mu-1} & S_\mu \\ S_2 & S_3 & S_4 & ... & S_\mu & S_{\mu+1} \\ S_3 & S_4 & S_5 & ... & S_{\mu+1} & S_{\mu+2} \\ ... & ... & ... & ... & ... & ... \\ S_\mu & S_{\mu+1} & S_{\mu+2} & ... & S_{2\mu-2} & S_{2\mu-1} \end{bmatrix}$$

$$M_\nu\begin{bmatrix}\Lambda_\nu\\ \Lambda_{\nu-1}\\ \Lambda_{\nu-2}\\ ...\\ \Lambda_1\end{bmatrix}=\begin{bmatrix}-S_{\nu+1}\\ -S_{\nu+2}\\ -S_{\nu+3}\\ ...\\ -S_{2\nu}\end{bmatrix}$$

4) Determine the estimated emitted word. What is the associated information word is a systematic code is used?

Reed-Solomon (RS) codes (1)

RS codes are **non binary BCH** codes with m=1

$n=q^m-1=q-1$

$GF(q^m)=GF(q)$ => the minimal polynomial of $\beta\epsilon\ GF(q^m)$ is $(X-\beta)$

The generator polynomial for an error correction capacity equal to t is :
$g(x)=(x-\alpha)(x-\alpha^2)\ldots(x-\alpha^{2t})$

Degree of g(x) is 2t => k=n-2t

RS(n,k=n-2t,t) : RS code of length n symbols, k information symbols, error correction capacity : t

Reed-Solomon (RS) codes (2)

Ex: RS(255,239,t=8)

q=256 => 1 symbol is composed of 8 bits (byte)

Error correction capacity : 8 **bytes** (up to 64 bits)

RS codes are suited to error bursts

Systematic RS codes can be shortened.

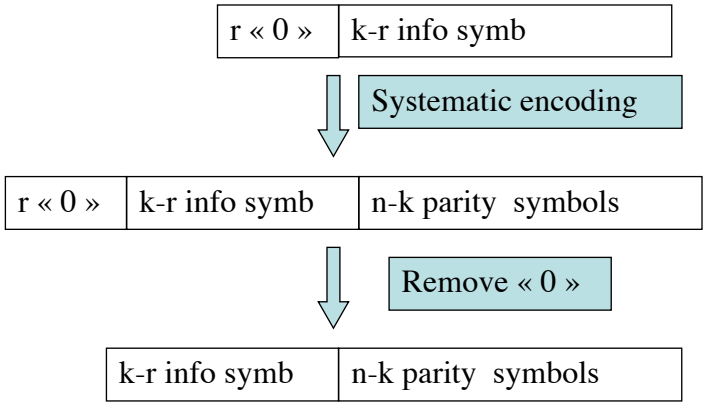
Ex: RS(204,188,t=8) : shortened version of RS(255,239,t=8)

Shortened codes (1)

Recall : primitive code $n=q^m-1$ BCH : (n,k) R=k/n

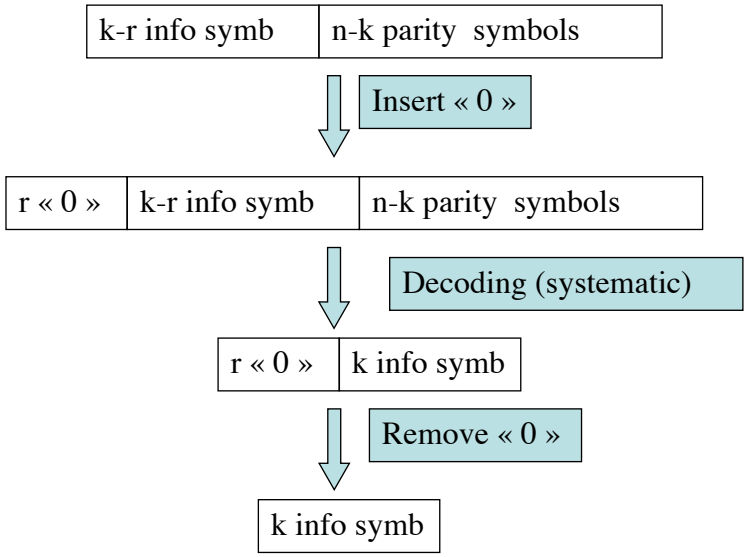
How can we obtain other code lengths ?

Possible solution : shortening => BCH(n-r,k-r) R=(k-r)/(n-r)



Shortened codes (2)

Decoding:



Exercise study of the (7,3) RS code

The RS code is defined over GF(8).

GF(8)={0=(000), 1=(001),α=(010), α^2=(100), α^3=(011)=1+α,

α^4=(110)= α^2+α, α^5=(111)= α^2+α+1, α^6=(101)= α^2+1}

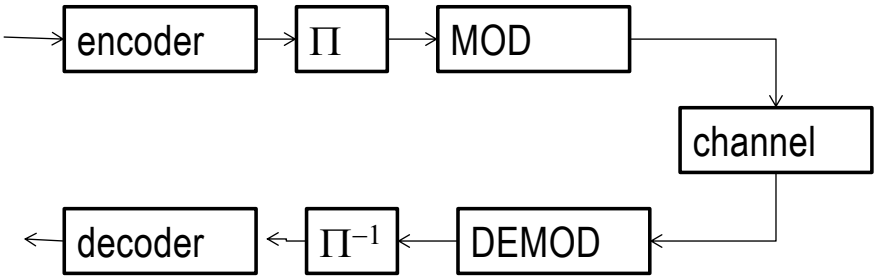
a)What is the error correction capability of the code and the number of code words?

b)Show that g(x), the generator polynomial of the code, is : $g(x) = x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3$

c)The encoder is fed with the following information bits : i=000 110 000
i) Express i as 3 symbols of GF(8) and define the associated information polynomial (the rightmost 3 bits of i corresponds to the degree 0)
ii)Show that the code word c(x) at the systematic encoder output is :
$c(x) = \alpha^4 x^5 + \alpha^6 x^3 + \alpha^4 x^2 + \alpha^3 x + \alpha^3$. Determine the corresponding coded bits.

d)The decoder receives: v=000 110 000 000 000 011 011Decoding is performed with the Peterson algorithm. The rightmost 3 bits of v corresponds to the degree 0 (same convention as encoder!)
i)Determine the received word v(x) with coefficients belonging to GF(8). Calculate the 4 syndromes S_j=v(α^j), j=1,2,3,4.
ii) Find that the (estimated) number of errors v is equal to 2. Let : $e(x) = Y_1 X^{i_1} + Y_2 X^{i_2}$.
iii) Find the error location polynomial $\Lambda(x)=\Lambda_2 x^2 + \Lambda_1 x + 1$
iv) Calculate $\Lambda(\alpha^4)$ and $\Lambda(\alpha^5)$. Give the location of errors i_1 and i_2.
v) Find the error amplitudes Y_1 and Y_2.
vi) Find the estimated emitted code word
vii) Find the estimated information word assuming systematic encoding
viii) Give the 9 estimated information bits

Block Interleaving (1)



Block interleaving can be seen as a permutation (Π)

Block Interleaving (2)

Example :

Let us consider a binary code with codewords of length 15 and correction capacity t=3

The channel is a bursty channel with maximum burst length of 15 bits.

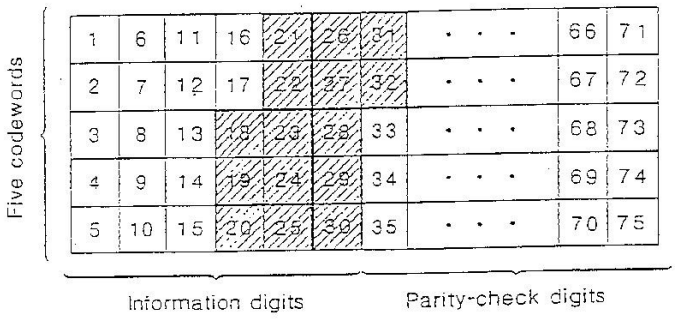
Without interleaving the codewords affected by error burts longer than 3 bits are not correctly decoded.

Solution: spreading long error bursts over several codewords in order to have a maximum of 3 erroneous bits per codeword.

This can be acheived by using a block interleaver of 5 codewords.

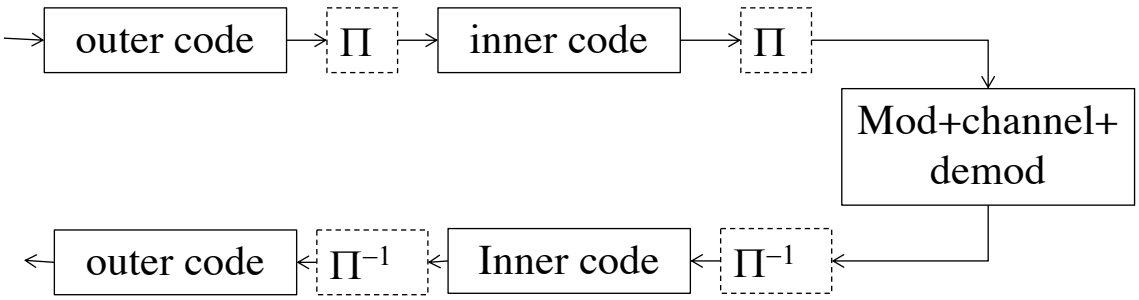
Block Interleaving (3)

This is a block interleaver. (5 lines, 15 columns).
At the emitter side, the codewords are written line by line and read colum by column.
At the receiver side, the bits are separated in blocks of 75 bits, written column by column and read line by line.
The shaded area represents an error burst of 15 bits. Note that the maximum of error per codeword is now 3 bits, which is within the error correction capacity of the code.



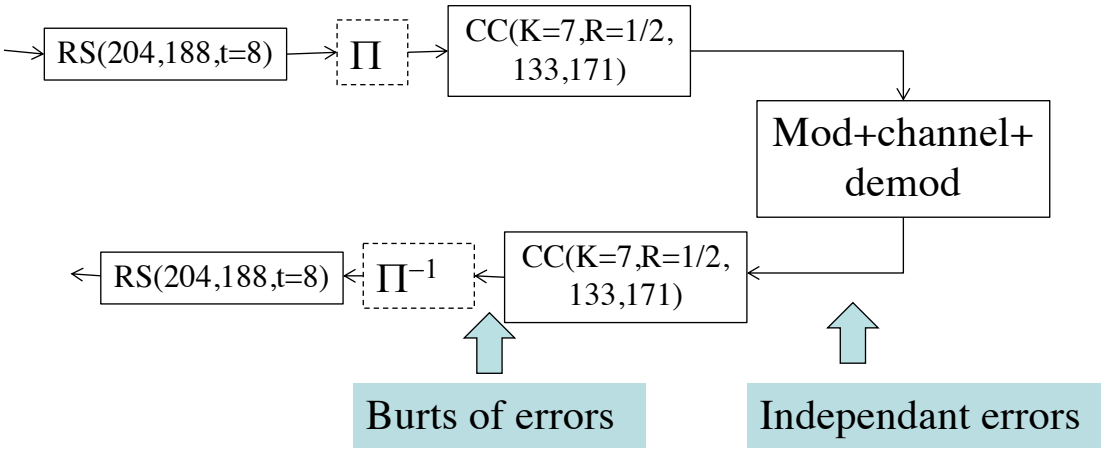
Concatenated codes (1)

A serially concatenated code allows to acheive the performance of a longer code with lower complexity.



:optionnal

Example : DVB-S2 (Fixed Satellite Service)



Burts of errors are due to the use of the Viterbi algorithm

ENSEEIHT

List of Binary BCH codes and their generator polynomials

N=7,15,31,63

n	k	t	$g(x)$
7	4	1	13
15	11	1	23
	7	2	721
	5	3	2467
31	26	1	45
	21	2	3551
	16	3	107657
	11	5	5423325
	6	7	313365047
63	57	1	103
	51	2	12471
	45	3	1701317
	39	4	166623567
	36	5	1033500423
	30	6	157464165547
	24	7	17323260404441
	18	10	1363026512351725
	16	11	6331141367235453
	10	13	472622305527250155
	7	15	5231045543503271737

N=127, 255

127	120	1	211
	113	2	41567
	106	3	11554743
	99	4	3447023271
	92	5	624730022327
	85	6	130704476322273
	78	7	26230002166130115
	71	9	6255010713253127753
	64	10	1206534025570773100045
	57	11	335265252505705053517721
	50	13	5444651252331401242150142l
	43	14	17721772213651227521220574343
	36	15	3146074666522075044764574721735
	29	21	403114461367670603667530141176155
	22	23	123376070404722522435445626637647043
	15	27	22057042445604554770523013762217604353
	8	31	704726405275103065147622427156773313021 7
255	247	1	435
	239	2	267543
	231	3	156720665
	223	4	75626641375
	215	5	23157564726421
	207	6	16176560567636227
	199	7	7633031270420722341
	191	8	2663470176115333714567
	187	9	52755313540001322236351
	179	10	22624710717340432416300455

255	171	11	154162142123423560770616306 37
	163	12	75004155100756025515747245146 01
	155	13	37575130054076650157225046467 7633
	147	14	16421301735371655253041653054410 11711
	139	15	46140173206017556157072273024745356 7445
	131	18	21571333147151015126125027744214202 41 65471
	123	19	12061405224206600371721032651614122 62 72506267
	115	21	60526665572100247263636404600276352 55 6313472737
	107	22	22205772322066256312417300235347420 17 6574750154441
	99	23	10656667253473174222741416201574332 25 2411076432303431
	91	25	67502650303274441727236317247325110 75 550762720724344561
	87	26	11013676341474323643523163430717204 62 06722545273311721317
	79	27	66700035637657500020270344207366174 62 10153267117665413423 55
	71	29	24024710520644321515554172112331163 20 54442503625576432217060 35
	63	30	10754475055163544325315217357707003 66 61117264552676136567025433 01
	55	31	73154252035011001330152753060320543 25 41432675501055704442603547 3617
	47	42	25335420170626465630330413774062331 75 12333414544604500506602455254317 3
	45	43	15202056055234161131101346376423701 56 36700244707623730332021570250515 41
	37	45	51363302550670074141774472454375304 20 73570617432343234764435473740304400 3
	29	47	30257155366730714655270640123613771 15 34224232420117411406025475741040356 5037
	21	55	12562152570603326560017731536076121 03 22734140565307454252115312161446651 3473725
	13	59	46417320050525645444265737142500660 04 33067744547656140317467721357026134 460500547
	9	63	15726025217472463201031043255355134 61 41623672120440745451127661155477055 61677516057

N=255 (con't)

ENSEEIHT

Complements

ENSEEIHT

CRC codes

Example : CCSDS CRC-16

What is CCSDS

CCSDS : The consultative committee for space data systems

Founded in 1982 by the major space agencies of the world, the CCSDS is a multi-national forum for the development of communications and data systems standards for spaceflight.

Frame integrity check

The CCSDS applications are packet-oriented, which means that data are collected and transmitted in frames. With all coding options, and also for uncoded data, it is important to have a reliable indication whether the decoded data is correct.

A frame integrity check can be used at the receiver side to validate the received frame or, when suitable, for requiring retransmission in case of check failure.

A solution to this data validation problem exists in the form of a **cyclic redundancy check (CRC) code**, as specified in the TM Space Data Link Protocol Blue Book

CRC code (1)

A **binary systematic linear code** is used to detect bit errors in the Frame Error Control Field (FECF) of the TM/TC Transfer Frame transmission.

Usually, the term CRC refers to the parity bits produced by the encoding circuit, which are appended to the message before transmission. Rather than a cyclic code, the CRC is usually a shortened cyclic code.

CRC (2)

Recall : systematic code

i(x) : information polynomial (degree : k-1)

c(x) : coded polynomial (degree N-1)

g(x) : generator polynomial (degree : N-k)

$c(x)=x^{N-k}i(x)+t(x)$ with $t(x)=x^{N-k}i(x)$ modulo g(x)

Error detection :

v(x) : received word (degree : N-1)

Check if v(x)=0 modulo g(x)

Property of binary (N,k) cyclic code (1)

A binary (n, k) CRC code, obtained by shortening a cyclic code, is capable of detecting the following error patterns:

a) all error bursts of length n–k or less

b) a fraction of error bursts of length equal to n–k+1; this fraction equals $1–2^{–(n–k–1)}$

c) a fraction of error bursts of length greater than n–k+1; this fraction equals $1–2^{–(n–k)}$

d) all error patterns containing $d_{min}–1$ (or fewer) errors

e) All errors patterns with an odd numbers of errors if g(x) has a even number of non zero coefficients

Property of binary (N,k) cyclic code (2)

Important remark:

If g(x) has a **even** number of non nul coefficients => g(1)=0 => x+1 divides g(x)

Let e(x) be a polynomial with a **odd** number of coeffiecients.

e(1)≠0 => x+1 does not divides e(x) => g(x) does not divides e(x)

=> e(x) is not a code word => v(x)=c(x)+e(x) is not a code word

=> **All patterns with odd number of errors are detected**

CCSDS CRC-16

Choice of the primitive cyclic code:

CRC-16 => 16 parity bits (N-k=16, degree of g(x))

$g(x)=x^{16}+x^{12}+x^5+1$

Smallest N such that g(x) divides x^N-1 : $N=2^{15}-1=32767$

d=4 (distance by construction)

CRC-16 : shortened (32767,32751) code

CCSDS CRC-16

Error detection capacity of CCSDS CRC-16 :

a) all error bursts of length 16 or less;

b) a fraction of error bursts of length equal to 17; this fraction equals $1–2^{–15}$;

c) a fraction of error bursts of length greater than 17; this fraction equals $1–2^{-16}$;

d) all error patterns containing 1, 2 or 3 errors;

e) all error patterns with an odd number of errors.

Performance of CCSDS CRC-16

Probability of undetected erroneous frame:

$P_u \approx A_4 \cdot Pe^4 \cdot (1–Pe)^{n–4}$

A_4 depends on k and can be found in:

D. Fiorini, M. Chiani, V. Tralli and C. Salati , “Can we trust in HDLC?” ACM SIGCOMM Computer Communication Review, vol. 24, no. 5, pp. 61–80, Oct. 1994