ADRIAN ATANASIU

TEORIA CODURILOR CORECTOARE DE ERORI

EDITURA UNIVERSITĂȚII DIN BUCUREȘTI

ADRIAN ATANASIU

TEORIA CODURILOR CORECTOARE DE ERORI

Prof. dr. ADRIAN ATANASIU

TEORIA CODURILOR CORECTOARE DE ERORI

Editura Universității din București
– 2001 –

208/02

Șos. Panduri 90-92, București - 76235; Tel./Fax: 410.23.84
E-mail: editura@unibuc.ro
Internet: www.editura.unibuc.ro

Descrierea CIP a Bibliotecii Naționale a României
ATANASIU, ADRIAN

Teoria codurilor corectoare de erori / Adrian Atanasiu. - București : Editura Universității din București, 2001
p. ; cm.
Bibliogr.
ISBN 973-575-589-0

004.052.44

Lui Silviu Guiaşu

Oamenii cunosc numai ce recunosc.
(proverb)

Prefaţă

Vă uitaţi la televizorul care transmite imagini prin satelit ? Vorbiţi la telefon (celular) ? Folosiţi Internet-ul ? Ascultaţi muzică de pe CD-uri ?

Aţi admirat fotografii extrem de reuşite ale planetelor Jupiter sau Saturn. Cum au fost obţinute ele ?

Aţi auzit de alfabetul Morse; v-aţi întrebat de ce pentru ajutor apelul este *S.O.S* ?

Sunt multe astfel de domenii intrate în cotidian, care folosesc coduri. De fapt oricare din exemplele de sus se referă la un transfer de informaţie. Ceea ce se solicită (în mod neexplicit, dar esenţial) este ca informaţia cerută să fie nemodificată la recepţie. Indiferent prin ce mediu (numit *canal*) se face transmisia (poate fi rază laser, cablu, unde etc) mesajul expeditorului trebuie să fie identic cu cel al destinatarului. Chiar dacă sunt "zgomote" care pot altera acest conţinut.

Tocmai din cauza acestor posibile perturbaţii de canal, ceea ce se transmite (numit *mesaj de informaţie*) se completează cu elemente redondante (numite *caractere de control*), care nu aduc informaţie suplimentară dar o confirmă pe cea existentă. În acest fel, o modificare posibilă a mesajului (o zgârietură pe CD, o interferenţă, un ecou de canal) poate fi eliminată la recepţie de către decodor.

Să considerăm de exemplu un mesaj *'abac'* format din 4 caractere. Îl vom codifica prin scrierea fiecărui caracter de trei ori. Deci vom transmite *'aaabbbaaaccc'*. Dacă se primeşte *'caabbwaaaccb'*, decodificatorul va separa textul în grupe de câte trei simboluri *'caa bbw aaa ccb'* şi va păstra din fiecare grup caracterul care apare de cele mai multe ori: *'abac'*. Este o modalitate de decodificare folosită cu precădere de aproape toate sistemele, numită *decodificarea cea mai probabilă.*

Evident, pot apare perturbări grupate, care să altereze mai multe caractere consecutive. Cum se operează atunci ? Pur şi simplu se foloseşte altă codificare (sau supra-codificare, care este o combinaţie de mai multe codificări). S-a ajuns în acest moment să existe clase de coduri (turbocodurile) care să corecteze aproape 75 % din simbolurile alterate ale unui

mesaj !

Pare fascinant. De fapt, aşa şi este !

Teoria codurilor este un domeniu teoretic dezvoltat relativ recent. Dacă se face abstracţie de unele confuzii de termeni cu criptografia (care foloseşte adesea termenul de *'cod'* pentru unele sisteme clasice de criptare) sau de ideea că de fapt şi scrierea sau vorbirea constituie sisteme de codificare ale ideilor, primul cod care merită acest nume în lumina obiectivelor menţionate anterior, este codul Morse, apărut în 1840 – de fapt un alfabet care să permită transmiterea mesajelor prin telegraf.

Explozia informaţională din a doua jumătate a secolului XX a condus la apariţia acestei discipline, ale cărei baze teoretice sunt puse în anii '50, are o dezvoltare fulgerătoare până la sfârşitul anilor '60, stagnează aproape două decenii, după care renaşte din 1985 şi cunoaşte astăzi din nou o creştere explozivă, explicabilă prin necesitatea transmiterii unui volum de informaţie nemai-întâlnit la o viteză aproape nebănuită până acum $10 - 15$ ani.

Cea mai mare parte a celor prezentate aici constituie suportul unui curs la Facultatea de Matematică a Universităţii Bucureşti (studii aprofundate, secţia informatică). Lucru explicabil, deoarece bazele teoriei codurilor sunt pur matematice (algebră liniară, teoria numerelor, corpuri finite, statistică). Aceasta nu face însă domeniul mai puţin atrăgător, dimpotrivă. Sugerează că pot fi gândite şi alte abordări care să genereze sisteme de codificare noi, cu performanţe superioare. Ideea în sine constituie o provocare, un apel al societăţii, al vieţii practice, adresat ştiinţelor fundamentale. Putem să îi răspundem ? Aceasta este întrebarea.

Autorul este conştient că nu a putut cuprinde şi explica tot ce conţine domeniul în sine. De asemenea, lucrarea poate cuprinde unele erori (nu numai de redactare). Este riscul pe care şi-l asumă orice autor. De aceea, rugămintea către cititor(ul interesat) este de a nu ezita stabilirea unei colaborări. Suntem deschişi şi receptivi la orice mesaj. În ciuda riscurilor unor posibile perturbaţii de canal.

Adrian Atanasiu

aadrian@pcnet.ro

mai 2001

Cuvânt înainte

Sfârşitul de secol este dominat clar de *revoluţia informatică*, considerată că reprezintă cea de-a treia (şi probabil, ultima) revoluţie industrială. *Internet* este o lume fascinantă şi în continuă schimbare, fiind în fruntea fenomenelor care influenţează viitorul economic mondial. Cea mai flexibilă structură pe care a cunoscut-o omenirea şi cel mai important proiect de conectivitate umană creat pe pământ, Internet înseamnă un volum imens de informaţie care este transferată de-a lungul conexiunilor. Se apreciază că sunt peste 15 milioane de utilizatori care navighează în fiecare moment. Câţi dintre ei se gândesc că la baza acestei uriaşe cantităţi de informaţie vehiculate stau codurile, cele care conferă acurateţe acestor date transmise pe canale imperfecte.

Oamenii din întreaga lume au fost fascinaţi de imaginile şi datele ştiinţifice pe care misiunile spaţiale le-au transmis de-a lungul a peste patru decade, din cele mai îndepărtate planete ale sistemului nostru solar. Şi probabil că şi-au pus întrebarea, firească, cum se poate ca această informaţie să fie transmisă într-o manieră fiabilă, la aceste distanţe de sute sau mii de milioane de kilometri, fără a fi complet "sufocată" de zgomot. Aici conlucrează mai multe discipline pentru recuperarea acestor semnale: ingineria electronică, calculatoarele şi matematica.

Sunt numai două exemple care subliniază importanţa domeniului abordat de această carte. Într-un limbaj simplu, *teoria codurilor* este o ramură a matematicii ce are ca obiect de studiu tocmai transmiterea datelor de-a lungul unor canale cu zgomot şi recuperarea mesajelor. Deşi are numeroase interferenţe şi o filiaţie comună cu *criptografia*, teoria codurilor este ştiinţa care face mesajele *uşor* de citit, pe când criptografia îşi propune tocmai să le facă *greu* de citit şi interpretat.

"Istoria" teoriei codurilora atins deja 50 ani, ceea ce pentru evoluţia ştiinţei contemporane reprezintă o perioadă de suficientă maturitate. În anul 1948, Claude Shannon, lucând la Laboratoarele Bell din SUA, a pus bazele studiilor de teoria codurilor, arătând că este posibil să se codifice mesaje astfel încât numărul de extra-biţi să fie minim. Din păcate

demonstraţia sa nu furniza nici un indiciu asupra modalităţilor de realizare a acestor *coduri optimale*.

Doi ani mai târziu, Richard Hamming, lucrând tot la Bell Labs, a început că studieze şi a publicat în mod concret rezultate asupra *codurilor corectoare de erori*. El a produs iniţial un cod în care 4 biţi de date erau urmaţi de 3 biţi de verificare, care să asigure nu doar detecţia ci şi corectarea erorilor de transmisie.

Valoarea codurilor corectoare de erori pentru transmisia informaţiei, atât pe pamânt cât şi în spaţiu, a fost evidenţiată imediat. A apărut o mare varietate de coduri care urmăreau două obiective fundamentale: economia în volumul de date transmis şi corectarea erorilor. Între anii 1969 şi 1973, modulul spaţial Mariner, trimis pe Marte de NASA, a folosit un cod puternic de tip Reed-Muller, capabil să corecteze 7 erori la fiecare 32 de biţi transmişi şi care foloseşte la fiecare 6 biţi de date 26 biţi pentru verificare ! Peste 16.000 biţi erau retransmişi în fiecare secundă spre Pământ.

O aplicaţie mai puţin evidentă a codurilor corectoare de erori vine din domeniul dezvoltării compact - discurilor. Pe un CD semnalele sunt codificate digital. Pentru a se proteja de zgârieturi sau de alte avarii, se folosesc intercalate două coduri, care pot corecta până la 4.000 de erori consecutive (aproximativ $2,5$ mm de pistă). La discurile audio, prin interpolarea semnalului, se poate recupera chiar mai multă informaţie avariată.

În ultimii ani, s-au făcut nenumărate studii pentru a se găsi coduri care să atingă limitele prevăzute de Shannon. Construcţia acestor coduri a cerut şi cere tehnici şi instrumente din domenii foarte variate ale matematicii, cum ar fi algebra lineară, teoria corpurilor, geometria algebrică etc. Au apărut lucrări de referinţă în domeniul matematicii codurilor corectoare de erori, cum ar fi cea a lui Dominic Welsh, *Codes and Cryptography*, apărută în Oxford University Press sau cea a lui Ray Hill, *A First Course in Coding Theory*, în aceiaşi prestigioasă editură.

La noi în ţară, sunt cunoscute lucrările din domeniul teoriei codurilor ale unor cunoscute personalităţi ştiinţifice: Grigore Moisil, Alexandru Spătaru, Silviu Guiaşu, Ion Angheloiu, Eugen Gyorfi, Dumitru Ene. Mai nou, cercetători şi universitari cunoscuţi ca Militon Frenţiu, Ferucio Ţiplea şi autorul acestei lucrari, Adrian Atanasiu, au obţinut rezultate semnificative în acest domeniu.

Făcând această prezentare, am trecut în revistă o foarte mică parte a subiectelor discutate în fascinanta lucrare elaborată de dr. Adrian

Atanasiu. Lucrarea are o construcţie în mai multe dimensiuni. Una este cea a şirului de teme expuse şi este pusă în evidenţă de împărţirea cărţii în capitole şi paragrafe. A doua dimensiune poate fi sesizată doar prin parcurgerea lucrării şi se referă la nivelul de dificultate al tratării. Este vorba de modelele matematice folosite în codificare. Prezentarea este riguroasă, justificările metodelor prezentate sunt bine concepute, iar exemplele sunt sugestive. Problemele discutate la acest nivel vor interesa cu siguranţă specialiştii sau studenţii care se pregătesc să lucreze în domeniul transmiterii informaţiei, dar şi doctoranzii care lucrează la teze conexe acestui domeniu. În final, mai există un nivel practic al lucrării, evidenţiat prin numeroasele exemple şi probleme propuse şi care conduce la o mai bună înţelegere a unor metode de codificare, unele foarte complexe. Ele vor interesa, desigur, tehnicienii şi proiectanţii care doresc să transmită informaţia prin mijloace cât mai fiabile.

Lucrarea, o adevărată monografie, prezintă domeniul teoriei codurilor aşa cum se înfăţişează astăzi, cu toată varietatea modelelor, metodelor şi instrumentelor produse în scurta dar densa sa istorie. Cartea dr. Adrian Atanasiu, cu valenţele sale teoretice dar şi aplicative, vine să completeze, la nivelul perioadei actuale, un gol în bibliografia românească a domeniului.

Bucureşti, 22 mai 2001

Prof. Dr. Ing. Victor-Valeriu Patriciu
Academia Tehnică Militară Bucureşti

Cuprins

Capitolul 1

Codificare şi decodificare

1.1 Notaţii preliminare

Peste tot în această lucrare, vom folosi câteva notaţii standard.

Astfel, cu litere latine mari ($A, B, \ldots$) vom nota mulţimi (numite adesea şi *alfabete* sau *vocabulare*). Dacă mulţimile sunt finite, $card(A)$ va desemna numărul de elemente din mulţimea A. Mulţimea vidă $\emptyset$ va fi mulţimea fără nici un element.

Cu elementele unei mulţimi se pot forma secvenţe (sau *cuvinte*) de forma $\alpha = a_1 a_2 \ldots$. Mulţimea (infinită a) tuturor cuvintelor obţinute cu elemente din mulţimea A, se notează A^*. Numărul elementelor care apar în secvenţa α formează *lungimea lui* α şi se notează cu $|\alpha|$.

Evident, în A^* există şi secvenţa fără nici un element, notată ϵ şi numită *cuvântul vid* ($|\epsilon| = 0$). Se notează $A^+ = A \setminus \{\epsilon\}$.

Vom considera permanent o corespondenţă biunivocă între secvenţa finită nevidă $a_1 a_2 \ldots a_n$ şi vectorul $(a_1, a_2, \ldots, a_n)$, cele două moduri de reprezentare fiind folosite în paralel.

Noţiunile elementare de algebră liniară (spaţii liniare, baze, matrici etc.) se consideră cunoscute şi nu mai sunt reamintite. Acelaşi lucru este valabil şi pentru algebra polinoamelor de o variabilă.

Se folosesc de asemenea şi elemente de algebră a corpurilor finite, dar în general acestea nu depăşesc materia parcursă în învăţământul preuniversitar.

1.2 Codificare

Definiţia 1.1 *Fiind date două mulţimi finite şi nevide A (alfabet sursă) şi B (alfabet cod), o* **codificare** *este o aplicaţie injectivă $\phi : A \to B^*$.*

Mulţimea $C = \phi(A)$ se numeşte *cod*, iar elementele sale sunt *cuvinte - cod*.

Dacă B are numai două simboluri, codificarea ϕ se numeşte *binară*, iar $\phi(A)$ este un *cod binar*.

Exemplul 1.1 *Fie $A = \{0, 1, \ldots, 9\}$ şi $B = \{0, 1\}$. Printre secvenţele binare de lungime 5, numărul celor care au doi de 1 este $C_5^2 = 10$. Ele pot fi folosite pentru a codifica cifrele din scrierea zecimală (Tabelul 1.1).*

Simbol zecimal	Cuvânt cod
1	11000
2	10100
3	01100
4	10010
5	01010
6	00110
7	10001
8	01001
9	00101
0	00011

Tabelul 1.1. Codul "doi din cinci".

Mesajul "173" are codul 110001000101100. De remarcat că între cuvintele - cod nu se lasă nici un spaţiu, deoarece "spaţiu" poate fi el însuşi un simbol din alfabetul - cod. Astfel de exemplu, codul Morse are alfabetul $B = \{., -, spaţiu\}$.

Decodificarea se face foarte simplu: se împarte mesajul codificat în grupe de câte cinci caractere şi se caută cifra din Tabelul 1.1 care corespunde grupei respective.

Aranjarea cuvintelor - cod a fost făcută pentru a realiza şi o decodificare pe baza unei formule. Astfel, dacă $a_0a_1a_2a_3a_4$ este un cuvânt - cod, el corespunde cifrei k rezultată din algoritmul:

{
    $x \leftarrow a_1 + 2a_2 + 4a_3 + 7a_4$;
    if $x = 11$ **then** $k \leftarrow 0$ **else** $k \leftarrow x$;
}

Definiţia 1.2 *Pentru o codificare $\phi : A \to B^*$, se numeşte "codificare a mesajelor (textului) sursă" aplicaţia $\phi^* : A^* \to B^*$ definită recursiv prin:*

$$\phi^*(\epsilon) = \epsilon;$$
$$\phi^*(a\alpha) = \phi(a)\phi^*(\alpha),\ \forall a \in A, \alpha \in A^*.$$

Definiţia 1.3 *Codificarea ϕ este "unic decodabilă" dacă ϕ^* este injectivă.*

Codificarea dată în Exemplul 1.1 este - după cum s-a observat - unic decodabilă. Acest lucru nu este totdeauna posibil. Astfel, dacă luăm codificarea

$$\phi(a) = 10, \quad \phi(b) = 101, \quad \phi(c) = 110,$$

ea nu este unic decodabilă, pentru că putem avea $\phi^*(ac) = \phi^*(ba) = 10110$.

Definiţia 1.4 *1. O codificare $\phi : A \to B^*$ în care toate cuvintele-cod au lungimea n se numeşte "codificare - bloc de lungime n", iar $\phi(A)$ este un "cod-bloc de lungime n".*

2. O codificare $\phi : A \to B^*$ se numeşte "instantanee" dacă $\phi(A)$ are proprietatea prefixului (dacă $\alpha, \alpha\beta \in \phi(A)$ atunci $\beta = \epsilon$);

$\phi(A)$ se numeşte "cod instantaneu".

Codul definit în Exemplul 1.1 este un cod - bloc de lungime 5.

Codurile bloc sunt eficiente în situaţia când simbolurile sursă au frecvenţe egale de apariţie; în caz contrar, ele devin greoaie şi sunt preferabile codurile instantanee cu lungimi variabile ale cuvintelor-cod.

Exemplul 1.2 *Codul Morse, dat în Tabelul 1.2 este un cod instantaneu cu alfabetul cod $B = \{., -, \}$. Deoarece spaţiul este folosit numai la sfârşitul fiecărui cuvânt - cod, procedura de decodificare este simplă: orice cuvânt - cod se află între două spaţii, de la începutul mesajului până la primul spaţiu, sau de la ultimul spaţiu până la sfârşit. Motivul pentru care nu se foloseşte un cod - bloc este simplu: frecvenţele literelor într-o limbă diferă foarte mult.*

A . -	F . . - .	K - . -	P . - - .	U . . -
B - . . .	G - - .	L . - . .	Q - - . -	V . . . -
C - . - .	H	M - -	R . - .	W . - -
D - . .	I	N - .	S	X - . . -
E	J . - - -	O - - -	T -	Y - . - -
				Z - - . .

Tabelul 1.2. Alfabetul Morse.

Exemplul 1.3 *Un alt exemplu de cod - bloc este codul hexazecimal - binar:*

0	0000	4	0100	8	1000	C	1100
1	0001	5	0101	9	1001	D	1101
2	0010	6	0110	A	1010	E	1110
3	0011	7	0111	B	1011	F	1111

Exemplul 1.4 *Să presupunem că vrem să construim un cod binar pentru alfabetul* $A = \{0, 1, 2, 3\}$ *şi observăm că* 0 *apare în mesajele sursă mai des decât orice alt simbol. Atunci următoarea schemă de codificare pare rezonabilă:*

$$\phi(0) = 0, \qquad \phi(1) = 01, \qquad \phi(2) = 011, \quad \phi(3) = 111.$$

Aici $\phi(0)$ *are lungime minimă, iar algoritmul de decodificare este foarte simplu: se aplică recursiv regula:*

"Fie sufixul 01^k*: el este codificarea numărului* $p\underbrace{33\ldots3}_{k \text{ div } 3}$ *unde* $p \equiv k \pmod 3$*."*

Totuşi această codificare nu este instantanee. Într-adevăr, dacă se primeşte un mesaj lung de forma

$$011111111111111\ldots$$

nu vom şti dacă primul simbol sursă este $0, 1$ *sau* 2 *până nu se termină citirea mesajului.*

1.3 Exemple de coduri - bloc importante

Deşi simple, codurile binare sunt de obicei lungi şi deci greu de manipulat. Adesea este mai convenabil să grupăm simbolurile binare formând alfabete mai complexe.

Astfel, formând grupuri de câte patru simboluri binare, se obţine codul hexazecimal (Exemplul 1.3). Reprezentarea în această bază (care foloseşte simbolurile auxiliare A, B, C, D, E, F pentru numerele $10, 11, 12, 13, 14$ şi respectiv 15) se indică adesea prin indicele 16 aşezat la sfârşit. De exemplu, $(61)_{16} = 0110\ 0001$.

Un cod foarte important folosit în reprezentarea standard a simbolurilor alfabetice şi numerice este codul *ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange)*. El are $2^8 = 256$ simboluri sursă, codificate în secvenţe binare de lungime 8; prezentăm în Tabelul 1.3

simbolurile printabile având codurile ASCII în intervalul $[32, 127]$ (în domeniul $[0, 31]$ nu sunt simboluri printabile, iar în $[128, 255]$ se definesc simboluri auxiliare, care nu sunt pe tastatură).

Char	Cod	Char	Cod	Char	Cod
	$(20)_{16}$	@	$(40)_{16}$	'	$(60)_{16}$
!	$(21)_{16}$	A	$(41)_{16}$	a	$(61)_{16}$
"	$(22)_{16}$	B	$(42)_{16}$	b	$(62)_{16}$
#	$(23)_{16}$	C	$(43)_{16}$	c	$(63)_{16}$
$	$(24)_{16}$	D	$(44)_{16}$	d	$(64)_{16}$
%	$(25)_{16}$	E	$(45)_{16}$	e	$(65)_{16}$
&	$(26)_{16}$	F	$(46)_{16}$	f	$(66)_{16}$
'	$(27)_{16}$	G	$(47)_{16}$	g	$(67)_{16}$
(	$(28)_{16}$	H	$(48)_{16}$	h	$(68)_{16}$
)	$(29)_{16}$	I	$(49)_{16}$	i	$(69)_{16}$
*	$(2A)_{16}$	J	$(4A)_{16}$	j	$(6A)_{16}$
+	$(2B)_{16}$	K	$(4B)_{16}$	k	$(6B)_{16}$
‘	$(2C)_{16}$	L	$(4C)_{16}$	l	$(6C)_{16}$
−	$(2D)_{16}$	M	$(4D)_{16}$	m	$(6D)_{16}$
.	$(2E)_{16}$	N	$(4E)_{16}$	n	$(6E)_{16}$
/	$(2F)_{16}$	O	$(4F)_{16}$	o	$(6F)_{16}$
0	$(30)_{16}$	P	$(50)_{16}$	p	$(70)_{16}$
1	$(31)_{16}$	Q	$(51)_{16}$	q	$(71)_{16}$
2	$(32)_{16}$	R	$(52)_{16}$	r	$(72)_{16}$
3	$(33)_{16}$	S	$(53)_{16}$	s	$(73)_{16}$
4	$(34)_{16}$	T	$(54)_{16}$	t	$(74)_{16}$
5	$(35)_{16}$	U	$(55)_{16}$	u	$(75)_{16}$
6	$(36)_{16}$	V	$(56)_{16}$	v	$(76)_{16}$
7	$(37)_{16}$	W	$(57)_{16}$	w	$(77)_{16}$
8	$(38)_{16}$	X	$(58)_{16}$	x	$(78)_{16}$
9	$(39)_{16}$	Y	$(59)_{16}$	y	$(79)_{16}$
:	$(3A)_{16}$	Z	$(5A)_{16}$	z	$(7A)_{16}$
;	$(3B)_{16}$	[	$(5B)_{16}$	{	$(7B)_{16}$
<	$(3C)_{16}$	\	$(5C)_{16}$	—	$(7C)_{16}$
=	$(3D)_{16}$	]	$(5D)_{16}$	}	$(7D)_{16}$
>	$(3E)_{16}$	^	$(5E)_{16}$	~	$(7E)_{16}$
?	$(3F)_{16}$	_	$(5F)_{16}$	DEL	$(7F)_{16}$

Tabelul 1.3

De exemplu, litera A, va avea codul hexazecimal - binar $(41)_{16}$ =

01000001, căruia în zecimal îi corespunde numărul 65.

Un ultim cod, folosit de toate editurile din lume este *International Standard Book Number* (ISBN). El este un cod - bloc de lungime 10 (lungimea cuvintelor - cod creşte prin folosirea simbolului '-' pe diverse poziţii, dar acest caracter este ignorat la prelucrarea automată). Alfabetul cod este $B = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, X\}$ (X pentru numărul 10).

De exemplu, cartea *S. B. Honary, G. Markarian - Trellis Decoding of block Codes* are codul *ISBN 0 – 7923 – 9860 – 2*

Primul număr (0) reprezintă ţara (SUA), 7923 reprezintă editura (*Kluwer Academic Publ.*), iar următoarele patru cifre sunt asignate de editură ca număr de identificare al cărţii. Ultimul simbol este de control (similar bitului de paritate) şi se defineşte astfel:

Pentru codul ISBN $a_{16}1a_2 \dots a_{10}$, $\sum_{i=1}^{10} i a_{11-i} = 0 \ (mod\ 11)$.

Astfel, pentru $ISBN$-ul dat ca exemplu,

$10 \cdot 0 + 9 \cdot 7 + 8 \cdot 9 + 7 \cdot 2 + 6 \cdot 3 + 5 \cdot 9 + 4 \cdot 8 + 3 \cdot 6 + 2 \cdot 0 + 1 \cdot 2 = 264 \equiv 0 \ (mod\ 11)$

Unele publicaţii au codul de identificare de două cifre (*Prentice Hall* are codul 13) sau trei (*Wiley-Interscience* are 471); în acest caz numărul pentru fiecare publicaţie are şase, respectiv cinci simboluri.

Pentru România, codul de ţară este 973; editura Teora are codul 601.

1.4 Construcţia codurilor instantanee

În acest paragraf, ne punem problema construirii unui cod binar instantaneu $\phi : A \longrightarrow B$, unde $A = \{a_1, \dots, a_n\}$ ($n \geq 1$) şi $card(B) = 2$.

Iniţial se specifică lungimile $d_1, d_2, \dots, d_n$ ale cuvintelor - cod: $\phi(a_i) = d_i$ ($1 \leq i \leq n$). Fără a micşora generalitatea, putem presupune de asemenea $1 \leq d_1 \leq d_2 \leq \dots \leq d_n$.

La primul pas se alege un cuvânt - cod binar arbitrar $\phi(a_1)$ de lungime d_1.

În continuare se alege un cuvânt - cod arbitrar $\phi(a_2)$ din mulţimea secvenţelor binare de lungime d_2, care nu au pe $\phi(a_1)$ ca prefix. Aceasta este totdeauna posibil pentru că:

Numărul tuturor secvenţelor binare de lungime d_2 este 2^{d_2}; dintre acestea, numărul celor care au prefixul $\phi(a_1)$ este $2^{d_2-d_1}$. Cum $2^{d_2} \geq 2^{d_2-d_1} + 1$, există cel puţin o alegere posibilă pentru $\phi(a_2)$ de lungime d_2.

Va trebui să selectăm în continuare un cuvânt de lungime d_3 care nu are ca prefix $\phi(a_1)$ sau $\phi(a_2)$. Deci, din cele 2^{d_3} secvenţe binare posibile

trebuiesc eliminate cele $2^{d_3-d_1}$ secvenţe cu prefixul $\phi(a_1)$ şi $2^{d_3-d_2}$ secvenţe cu prefixul $\phi(a_2)$. Aceasta este posibil dacă şi numai dacă

$$2^{d_3} \geq 2^{d_3-d_2} + 2^{d_3-d_1} + 1.$$

Împărţind această inegalitate cu 2^{d_3}, se obţine

$$1 \geq 2^{-d_1} + 2^{-d_2} + 2^{-d_3}.$$

În mod analog se poate arăta în general inegalitatea

$$1 \geq 2^{-d_1} + 2^{-d_2} + \ldots + 2^{-d_n}$$

Faptul că această codificare conduce la un cod instantaneu rezultă imediat din construcţie.

Teorema 1.1 *(Kraft). Fiind dat un alfabet sursă A de n simboluri şi un alfabet cod B de k simboluri, se poate construi un cod instantaneu cu lungimile cuvintelor - cod $d_1, d_2, \ldots, d_n$, dacă şi numai dacă este verificată inegalitatea:*

$$k^{-d_1} + k^{-d_2} + \ldots + k^{-d_n} \leq 1.$$

Demonstraţie: Fie $A = \{a_1, a_2, \ldots, a_n\}$ şi putem presupune relaţia $d_1 \leq d_2 \leq \ldots \leq d_n$. Construim codificarea instantanee ϕ prin inducţie astfel:

- Se alege $\phi(a_1) \in A^*$ arbitrar ($|\phi(a_1)| = d_1$).
- Presupunem că au fost alese $\phi(a_1), \phi(a_2), \ldots \phi(a_{s-1})$. Atunci se va alege un cuvânt arbitrar $\phi(a_s) \in a^*$, ($|\phi(a_s)| = d_s$) care nu are ca prefix nici unul din cuvintele selectate anterior. Aceasta este posibil deoarece numărul cuvintelor cu prefixul $\phi(a_i)$ este $k^{d_s-d_i}$ ($1 \leq i \leq s-1$); deci alegerea poate fi făcută din

$$k^{d_s} - \sum_{i=1}^{s-1} k^{d_s-d_i} \text{ elemente.}$$

Din inegalitatea lui Kraft avem

$$1 - \sum_{i=1}^{s-1} k^{-d_i} \geq k^{-d_s}$$

care, prin multiplicare cu k^{d_s} conduce la

$$k^{d_s} - \sum_{i=1}^{s-1} k^{d_s-d_i} \geq 1,$$

deci există cel puţin un cuvânt care poate fi ales drept $\phi(a_s)$.

Afirmaţia reciprocă se demonstrează similar (pentru $k = 2$ ea a fost dată anterior). □

Teorema 1.2 *(McMillan) Orice codificare unic decodabilă satisface inegalitatea lui Kraft.*

Demonstraţie: Fie ϕ o codificare unic decodabilă. Notăm cu d_i lungimea cuvântului - cod $\phi(a_i)$ $(1 \leq i \leq n)$. Se observă că $\forall j$ $(j \geq 1)$ se pot forma k^j cuvinte de lungime j peste un alfabet - cod B având k simboluri. Din proprietatea de unic decodabilitate, numărul mesajelor sursă $\alpha = a_{i_1}a_{i_2}\ldots a_{i_r}$ cu $|\phi(\alpha)| = j$, nu depăşeşte k^j. Lungimea codului pentru α este $d_{i_1} + d_{i_2} + \ldots + d_{i_r}$; deci numărul tuturor sumelor de forma

$$d_{i_1} + d_{i_2} + \ldots + d_{i_r} = j$$

este cel mult k^j.

Rămâne de demonstrat că numărul $c = \sum_{i=1}^{n} k^{-d_i}$ este cel mult 1.

Pentru aceasta, vom arăta că $\forall r \geq 1$, $\frac{c^r}{r}$ este mărginit.

Să calculăm puterile lui c:

$$c^2 = \left(\sum_{i=1}^{n} k^{-d_i}\right)\left(\sum_{j=1}^{n} k^{-d_j}\right) = \sum_{i,j=1}^{n} k^{-(d_i+d_j)}$$

şi - în general,

$$c^r = \sum_{i_1,i_2,\ldots,i_r=1}^{n} k^{-(d_{i_1}+d_{i_2}+\ldots+d_{i_r})}.$$

Această sumă se poate reordona grupând toţi termenii de forma k^{-j} unde j satisface egalitatea anterioară. Cel mai mare j posibil este $j = d + d + \ldots + d = rd$, unde $d = max\{d_1, d_2, \ldots, d_n\}$.

Numărul tuturor termenilor de forma k^{-j} din sumă este cel mult k^j. Deci,

$$c^r \leq \sum_{j=1}^{rd} k^j k^{-j} = \sum_{j=1}^{rd} 1 = rd.$$

Am obţinut relaţia $\frac{c^r}{r} \leq d$, de unde va rezulta $c \leq 1$ (pentru $c > 1$ şirul $a_r = \frac{c^r}{r} \to \infty$, deci nu este mărginit). □

Corolarul 1.1 *Pentru orice cod unic decodabil există un cod instantaneu care are toate cuvintele - cod de lungimi egale.*

Demonstraţie: Rezultă din demonstraţia teoremei precedente.

Exemplul 1.5 *Să considerăm alfabetul sursă $A = \{a, b, c\}$ şi alfabetul cod $B = \{0, 1\}$; deci $n = card(A) = 3$, $k = card(B) = 2$. Vrem să*

construim o codificare instantanee $\phi : A \to B$ care are toate cuvintele - cod de lungime d. Inegalitatea Kraft va da $2^{-d} + 2^{-d} + 2^{-d} \leq 1$, deci $2^{-d} \leq \frac{1}{3}$. Cel mai mic d care o verifică este $d = 2$. Deci orice mulţime de trei secvenţe binare de lungime 2 va putea fi folosită drept cod. Sunt 4 astfel de mulţimi:

$$\{00, 01, 10\}, \quad \{00, 01, 11\}, \quad \{00, 10, 11\}, \quad \{01, 10, 11\}$$

Dacă alfabetul sursă are patru elemente, $A = \{a, b, c, d\}$, atunci este posibil un singur astfel de cod instantaneu: $\{00, 01, 10, 11\}$.

1.5 Coduri Huffman

Am menţionat anterior faptul că dacă frecvenţa simbolurilor sursă nu este constantă, atunci codurile instantanee sunt preferabile codurilor - bloc, deoarece simbolurile care apar mai frecvent vor fi codificate prin cuvinte - cod mai scurte. Ne punem problema aflării unor codificări cât mai eficiente, în ipoteza că frecvenţele simbolurilor sursă sunt cunoscute exact (de exemplu probabilitatea distribuţiei simbolurilor sursă în mesaje).

Definiţia 1.5 *O sursă de informaţie este o pereche $S = (A, P)$ unde*

- *$A = \{a_1, a_2, \ldots, a_n\}$ este alfabetul sursă (mulţime ordonată);*
- *$P = \{P(a_1), P(a_2), \ldots, P(a_n)\}$ este mulţimea ordonată a probabilităţilor elementelor lui A, deci*
  - *$0 \leq P(a_i) \leq 1 \quad (1 \leq i \leq n)$;*
  - *$\sum_{i=1}^{n} P(a_i) = 1$.*

Fie ϕ o codificare a unei surse de informaţie. Dacă se notează cu $d_i = |\phi(a_i)|$, se poate defini *lungimea medie* $L(\phi)$ a cuvintelor - cod prin

$$L(\phi) = \sum_{i=1}^{n} d_i P(a_i).$$

O codificare este eficientă dacă lungimea medie a cuvintelor - cod este cât mai mică posibil.

Definiţia 1.6 *Fiind dată o sursă de informaţie S şi un alfabet cod, un cod Huffman este un cod instantaneu cu lungimea medie minimă.*

Lungimea medie (minimă) a unui cod Huffmann se notează cu $L_{min}(S)$.

Exemplul 1.6 *Să se determine un cod Huffman binar pentru alfabetul sursă* $A = \{a, b, c, d, e, f\}$ *ştiind că 'a' apare de două ori mai des decât 'e' şi 'e' de două ori mai des decât orice consoană.*

Deci, vom avea sursa de informaţie

Simbol	a	b	c	d	e	f
Probabilitate	0,4	0,1	0,1	0,1	0,2	0,1

Putem asigna deci un cuvânt - cod de lungime 1 lui 'a' şi unul de lungime 2 lui 'e'. Atunci lungimile cuvintelor - cod rămase sunt egale cu 4, iar inegalitatea lui Kraft este saturată: $\frac{1}{2} + \frac{1}{2^2} + \frac{4}{2^4} = 1$. *O astfel de codificare poate fi:*

$$\phi(a) = 0 \qquad \phi(c) = 1101 \qquad \phi(e) = 10$$
$$\phi(b) = 1100 \qquad \phi(d) = 1110 \qquad \phi(f) = 1111$$

Lungimea sa medie este

$$L = 0,4 + 2 \times 0,2 + 4 \times 4 \times 0,1 = 2,4.$$

Deci, pentru acest exemplu, $L_{min}(S) \leq 2,4$.

Se poate arăta că nu există o codificare cu lungimea medie mai mică, deci ce am construit poate fi considerat un cod Huffman.

1.5.1 Construcţia codurilor Huffman binare

O sursă de informaţie cu două simboluri are evident un cod Huffman de cuvinte - cod $\{0, 1\}$ (şi deci $L_{min}(S) = 1$).

O sursă cu trei simboluri $\{a_1, a_2, a_3\}$ în care a_1 are probabilitate maximă poate fi redusă la cazul a două simboluri $\{a_1, a_{2,3}\}$ unde $P(a_{2,3}) = P(a_2) + P(a_3)$. Vom construi o codificare Huffman pentru sursa redusă

$$\phi(a_1) = 0, \qquad \phi(a_{2,3}) = 1,$$

după care "spargem" cuvântul - cod 1 în două cuvinte: 10 şi 11; în acest fel se obţine un cod Huffman ($\{0, 10, 11\}$) pentru sursa de informaţie originală, având codificarea

a_1	a_2	a_3
0	10	11

În general, fie S o sursă de informaţie, cu simbolurile $\{a_1, a_2, \ldots, a_n\}$ ordonate după probabilităţi, adică:

$$P(a_1) \geq P(a_2) \geq \ldots \geq P(a_n).$$

Construim o sursă redusă S^* cu simbolurile $\{a_1, \ldots, a_{n-2}, a_{n-1,n}\}$ unde $a_{n-1,n}$ este un simbol nou, cu probabilitatea $P(a_{n-1,n}) = P(a_{n-1}) + P(a_n)$.

Dacă nu se poate construi un cod Huffman pentru S^*, se reia procedeul pentru această sursă de informaţie (reordonând eventual simbolurile după probabilitate); în final se va ajunge la o sursă (pentru două simboluri problema a fost rezolvată) în care codul Huffman poate fi construit.

Dacă se poate găsi o codificare Huffman ϕ^* pentru sursa de informaţie redusă S^*, atunci codificarea din Tabelul 1.4

a_1	a_2	...	a_{n-2}	a_{n-1}	a_n
$\phi^*(a_1)$	$\phi^*(a_2)$	...	$\phi^*(a_{n-2})$	$\phi^*(a_{n-1,n})0$	$\phi^*(a_{n-1,n})1$

Tabelul 1.4

defineşte un cod Huffman pentru S (vom justifica această afirmaţie).

Lema 1.1 $L(\phi) = L(\phi^*) + P(a_{n-1}) + P(a_n)$.

Demonstraţie: Fie $d_1, d_2, \ldots, d_{n-2}, d^*$ lungimile cuvintelor cod corespunzătoare lui ϕ^*. Atunci lungimile cuvintelor - cod pentru ϕ sunt $d_1, d_2, \ldots, d_{n-2}, d^* + 1, d^* + 1$. Efectuând calculele, se obţine:

$$\begin{aligned} L(\phi) &= \sum_{i=1}^{n-2} d_i P(a_i) + (d^*+1)P(a_{n-1}) + (d^*+1)P(a_n) = \\ &= \sum_{i=1}^{n-2} d_i P(a_i) + d^*[P(a_{n-1}) + P(a_n)] + P(a_{n-1}) + P(a_n) = \\ &= L(\phi^*) + P(a_{n-1}) + P(a_n). \end{aligned}$$

□

Teorema 1.3 *Fie ϕ^* o codificare Huffman pentru o sursă de informaţie redusă S^*. Atunci codificarea ϕ definită de Tabelul* 1.4 *determină un cod Huffman pentru sursa de informaţie S.*

Demonstraţie: Fie $a_1, a_2, \ldots, a_n$ simbolurile sursă, ordonate descrescător după probabilitate. Deoarece teorema este evidentă pentru $P(a_n) = 0$, vom considera doar cazul $P(a_n) > 0$. Demonstraţia constă din trei paşi:

- S admite o codificare Huffman ϕ_0 cu lungimile cuvintelor - cod ordonate:

  $$d_1 \leq d_2 \leq \ldots \leq d_n \quad (d_i = |\phi_0(a_i)|, 1 \leq i \leq n).$$

  Pentru a demonstra aceasta, plecăm de la o codificare Huffman arbitrară ϕ pentru S. Dacă există un simbol a_i astfel ca $d_i > d_{i+1}$,

notăm cu ϕ' codificarea obţinută din ϕ prin permutarea cuvintelor - cod corespunzătoare lui a_i şi a_{i+1}. ϕ' este evident o codificare instantanee, iar diferenţa dintre lungimile medii $L = L_{min}$ (al lui ϕ) şi L' (al lui ϕ') este:

$$L_{min} - L' = [d_i P(a_i) + d_{i+1} P(a_{i+1})] - [d_{i+1} P(a_i) + d_i P(a_{i+1})] =$$
$$= (d_i - d_{i+1})[P(a_i) - P(a_{i+1})].$$

Această expresie este produsul dintre două numere pozitive, deci $L_{min} \geq L'$; din proprietatea de minimalitate rezultă şi $L_{min} = L'$. Cu alte cuvinte, ϕ' este tot o codificare Huffman. Procedeul continuă până se obţine codificarea ϕ_0 cerută.

- S admite o codificare Huffman ϕ_1 în care ultimele cuvinte - cod, $\phi_1(a_{n-1})$ şi $\phi_1(a_n)$ diferă doar prin ultimul simbol.

  Fie ϕ_0 codificarea Huffman anterioară şi ψ codificarea rezultată din ϕ_0 eliminând ultimul simbol din $\phi_0(a_n)$. Lungimea medie a lui ψ va fi evident mai mică decât cea a lui ϕ_0 (pentru că $P(a_n) > 0$), deci ψ nu poate fi codificare instantanee. Cuvântul - cod $\psi(a_i) = \phi_0(a_i)$ ($1 \leq i \leq n-1$ arbitrar) nu este prefixul nici unui alt cuvânt - cod; deci există un i ($i \leq n-1$) astfel încât $\psi(a_n)$ este prefixul lui $\phi_0(a_i)$. Aceasta este posibil numai dacă $d_i = d_n$ şi deci $\phi_0(a_i)$ diferă de $\phi_0(a_n)$ numai prin ultimul simbol. Dacă $i = n-1$, se ia $\phi_1 = \phi_0$. Altfel, se observă că $d_i = d_n$ implică $d_i = d_{i+1} = \ldots = d_n$; deci se pot permuta cuvintele - cod definite în ϕ_0 pentru a_i şi a_{n-1}. Codificarea ϕ_1 astfel obţinută are aceeaşi lungime medie ca şi ϕ_0, deci defineşte o codificare Huffman.

- Să presupunem că se dă o codificare Huffman ϕ^* pentru sursa redusă S^* şi definim o codificare ϕ pentru S conform Tabelului 1.4. Lungimile lor medii $L(\phi)$, $L(\phi^*)$ verifică relaţia din Lema 1.1.

  Să folosim acum codificarea Huffman ϕ_1 construită mai sus. Deoarece ultimele două cuvinte - cod diferă numai prin ultimul simbol, ϕ_1 poate fi obţinută dintr-o codificare φ_1^* a lui S^* prin "spargerea" ultimului cuvânt - cod. În plus, ϕ_1^* este evident instantanee. Prin calcule se ajunge la relaţia

  $$L(p_1) - L(\phi_1^*) = P(a_{n-1}) + P(a_n).$$

  Cum avem şi $L(\phi) - L(\phi^*) = P(a_{n-1}) + P(a_n)$, rezultă

  $$L(\phi) = L(\phi_1) - L(\phi_1^*) + L(\phi^*).$$

Acum, $L(\phi^*) = L_{min}(S^*)$, deci $-L(\phi_1^*) + L(\phi^*) \leq 0$. Rezultă $L(\phi) \leq L(\phi_1) = L_{min}(S)$. Deci, ϕ defineşte un cod Huffman.

□

1.6 Exerciţii

1.1 *Care este cea mai mică lungime a unui cod - bloc cu alfabetul sursă* $A = \{a, b, \ldots, z\}$ *şi alfabetul cod* $B = \{., -, spaţiu\}$ *(ca la codul Morse). Dar dacă* B *are patru caractere ?*

1.2 *Se defineşte codificarea*

1	→	01	4	→	1000
2	→	011	5	→	1100
3	→	10	6	→	0111

Este unic decodabilă ? Este instantanee ? Se poate găsi un cod instantaneu cu aceleaşi lungimi ale cuvintelor cod ?

1.3 *Se defineşte codificarea*

a	→	1010	d	→	0001
b	→	001	e	→	1101
c	→	101	f	→	1011

Este unic decodabilă ? Dacă nu, găsiţi două mesaje sursă cu acelaşi cod.

1.4 *Este unic decodabilă codificarea:*

0	→	xx	4	→	$xyyxx$	7	→	$xxxxyy$
1	→	$xxyxy$	5	→	$yxyyx$	8	→	$xxxxyx$
2	→	$xyyyyy$	6	→	$yyyxy$	9	→	$xxxxxy$
3	→	$xyxyx$						

1.5 *Se poate decide unic decodabilitatea codificărilor*

(i)	*(ii)*
$\phi(a) = 001$	$\phi(a) = 00$
$\phi(b) = 1001$	$\phi(b) = 10$
$\phi(c) = 0010$	$\phi(c) = 011$
$\phi(d) = 1110$	$\phi(d) = 101$
$\phi(e) = 1010$	$\phi(e) = 111$
$\phi(f) = 01110$	$\phi(f) = 110$
$\phi(g) = 0101$	$\phi(g) = 010$

folosind inegalitatea lui Kraft ?

1.6 *Să se construiască un cod binar instantaneu pentru următorul alfabet sursă, cu lungimile corespunzătoare ale cuvintelor - cod:*

Simbol	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*	*K*	*L*
Lungime	2	4	7	7	3	4	7	7	3	4	7	7

1.7 *Să se construiască un cod instantaneu pentru următorul alfabet sursă, cu lungimile corespunzătoare ale cuvintelor - cod:*

Simbol	1	2	3	4	5	6	7	8	9	0
Lungime	1	3	3	3	3	3	2	2	2	2

Care este numărul minim de simboluri cod necesare ?

1.8 *Câte simboluri cod sunt necesare pentru ca următorul alfabet sursă să poată fi codificat într-un cod instantaneu cu lungimile cuvintelor - cod cunoscute:*

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*m*	*n*	*o*	*p*	*q*	*r*	*s*	*t*
1	2	2	2	1	2	2	2	1	2	2	2	2	2	1	2

1.9 *Frecvenţa literelor în limba română este dată de următorul tabel (în procente), unde pe prima coloană sunt listate literele de frecvenţă rificată, pe coloana a doua cele de frecvenţă medie, iar pe ultima coloană, cele de frecvenţă mică.*

A	13,04%	*L*	4,58%	*Z*	0,87%
I	12,89%	*O*	3,85%	*G*	0,82%
E	11,75%	*D*	3,68%	*B*	0,77%
R	7,39%	*M*	3,33%	*H*	0,44%
T	6,62%	*P*	2,91%	*J*	0,19%
N	6,44%	*F*	1,50%	*X*	0,14%
U	6,44%	*V*	1,26%	*Y*	0,07%
S	5,50%			*Q*	0,04%
C	5,47%			*K*	0,01%
				W	0,00%

1. *Să se construiască un cod instantaneu, cu lungimile codurilor egale pe fiecare din cele trei coloane. Câte simboluri - cod sunt necesare?*

2. *Să se construiască un cod Huffman care să codifice alfabetul limbii române.*

Capitolul 2

Coduri liniare

Cea mai generală clasă de coduri detectoare şi corectoare de erori este clasa codurilor liniare. Toate celelalte tipuri de coduri care se definesc ulterior s-au dezvoltat doar în ideea obţinerii de performanţe superioare în privinţa decodificării sau a numărului de erori detectabile/corectabile. Indiferent însă dacă vor fi coduri ciclice sau convoluţionale, acestea pot fi rescrise în final sub formă de coduri liniare.

2.1 Matrice generatoare

Definiţia 2.1 *Fie q un număr prim şi $n \in N^*$ un număr natural nenul. Se numeşte "cod liniar" orice subspaţiu liniar al lui Z_q^n.*
Un subspaţiu k-dimensional al lui Z_q^n $(k \leq n)$ se numeşte (n,k)- cod liniar peste alfabetul Z_q.

Reamintim, $Z_q = \{0, 1, \ldots, q-1\}$ este corpul claselor de resturi modulo q (deoarece q este prim), cu operaţiile de adunare şi înmulţire modulo q.

Să notăm în general cu $A_{n,k}$ ($A_{n,k} \subseteq Z_q^n$) un (n,k) - cod liniar. Elementele sale se numesc *cuvinte-cod*. Dacă nu este necesar să specificăm dimensiunile (n,k), vom mai folosi şi notaţia C pentru un cod liniar oarecare.

Fie $n, k \in N^*$, $k \leq n$. După cum s-a definit în capitolul precedent, o *codificare* este o aplicaţie injectivă $\phi : Z_q^k \longrightarrow Z_q^n$, iar $A_{n,k} = \phi(Z_q^k)$. Elementele lui Z_q^k se numesc *mesaje de informaţie*.

Deci, $\mathbf{x} \in Z_q^k$ este un mesaj de informaţie scris cu caractere din alfabetul Z_q. Dacă transmitem succesiunea $\mathbf{x}$ de semnale printr-un canal de comunicaţie, mesajul este supus diverselor perturbări "de canal" care-l

modifică. Ideea teoriei codurilor este următoarea: în loc de a transmite numai elementele lui Z_q^k, să "lungim" mesajul informaţional scufundând (prin intermediul aplicaţiei ϕ) Z_q^k într-un spaţiu liniar Z_q^n $(n \geq k)$ astfel încât cele $n - k$ poziţii noi - numite *poziţii de control* - să asigure redondanţa necesară refacerii mesajului de informaţie iniţial (în cazul în care a fost modificat).

Astfel, cu preţul lungirii mesajului, se câştigă protecţie faţă de (anumite tipuri de) erori.

Observaţia 2.1

- *Din Definiţia 2.1 rezultă că un cod liniar de lungime n este un set C de cuvinte (secvenţe, şiruri, vectori) de lungime n cu proprietăţile:*

  (i) $\mathbf{a} = (a_1, \ldots, a_n) \in C$, $\mathbf{b} = (b_1, \ldots, b_n) \in C \Longrightarrow \mathbf{a} + \mathbf{b} = (a_1 + b_1, \ldots, a_n + b_n) \in C$;

  (ii) $\mathbf{a} = (a_1, \ldots, a_n) \in C, t \in Z_q \Longrightarrow t\mathbf{a} = (ta_1, \ldots, ta_n) \in C$.

- *Orice cod - liniar conţine cuvântul - cod nul* $\mathbf{0} = (0, 0, \ldots, 0)$.

- *Deoarece Z_q are q elemente, numărul de cuvinte - cod dintr-un cod liniar $A_{n,k}$ este q^k $(card(A_{n,k}) = q^k)$.*

$A_{n,k}$ fiind un spaţiu liniar k-dimensional, admite o bază formată din k secvenţe cu n elemente fiecare. Fie

$$\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_k}$$

o astfel de bază. Atunci orice cuvânt - cod $\mathbf{v} \in A_{n,k}$ are forma

$$\mathbf{v} = \sum_{i=1}^{k} u_i \mathbf{e_i}$$

unde $(u_1, u_2, \ldots, u_k) \in Z_q^k$ este unic determinat.

Cu alte cuvinte, k simboluri de informaţie $u_1, \ldots, u_k \in Z_q$ determină în mod unic un cuvânt - cod $\mathbf{v} \in A_{n,k}$ prin relaţia de sus, şi reciproc. Operaţia de codificare ϕ face această asociere biunivocă:

$$\mathbf{u} = (u_1, \ldots, u_k) \in Z_q^k \quad \Longleftrightarrow \quad \mathbf{v} = \sum_{i=1}^{k} u_i \mathbf{e_i} \in A_{n,k}.$$

Fiecare cuvânt - cod într-un cod liniar va avea deci k simboluri de informaţie; celelalte $n - k$ simboluri se numesc *simboluri de control.*

Definiţia 2.2 *Fie $A_{n,k}$ un cod liniar şi $\mathbf{e_1}, \ldots, \mathbf{e_k}$ o bază a sa. Matricea*

$$G_{k,n} = \begin{pmatrix} \mathbf{e_1} \\ \mathbf{e_2} \\ \ldots \\ \mathbf{e_k} \end{pmatrix}$$

se numeşte "matrice generatoare" a codului.

Deci operaţia de codificare este definită prin matricea generatoare:

$$\forall \mathbf{u} \in Z_q^k, \quad \phi(\mathbf{u}) = \mathbf{u}G.$$

Exemplul 2.1 *Matricea*

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

generează un $(5,2)$-cod liniar peste Z_2 ($n = 5, k = 2$). Rangul ei este 2 (primele două coloane formează matricea unitate), deci cele două linii ale lui G sunt cuvinte - cod liniar independente.

Mulţimea mesajelor de informaţie este $Z_2^2 = \{00, 01, 10, 11\}$.
Înmulţind fiecare mesaj cu matricea G se obţine spaţiul liniar al cuvintelor - cod

$$A_{5,2} = \{00000, 01101, 10011, 11110\}.$$

Funcţia de codificare este:

$$00 \to 00000, \quad 01 \to 01101, \quad 10 \to 10011, \quad 11 \to 11110.$$

Un cod liniar poate fi generat de mai multe baze posibile. Deci se pot construi mai multe matrici generatoare pentru un acelaşi (n,k)-cod liniar. Ele se pot transforma una în alta prin operaţiile liniare obişnuite, definite pentru liniile unei matrici. Prin schimbarea matricii generatoare nu se schimbă spaţiul liniar al cuvintelor - cod, ci numai modalitatea de codificare (funcţia ϕ). Cum prin termenul *cod* se înţelege de obicei spaţiul liniar $A_{n,k}$, rezultă că două matrici diferite care se deduc una din alta prin operaţii pe linii, reprezintă acelaşi cod.

Exemplul 2.2 *Reluând Exemplul 2.1, prin adunarea celei de-a doua linii la prima, se ajunge la matricea*

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

Ea generează același spațiu liniar $A_{5,2}$, dar codificarea diferă:

$$00 \to 00000, \quad 01 \to 01101, \quad 10 \to 11110, \quad 11 \to 10011.$$

Exemplul 2.3 *Matricea*

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

generează un $(6,3)$-cod liniar peste Z_3. Aducând matricea la forma canonică, se obține

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

Toate cele $3^3 = 27$ cuvinte ale codului au următoarea proprietate: fiecare simbol este scris de două ori. Din acest motiv, astfel de cod este numit "cod cu repetiție".

Cea mai convenabilă regulă de codificare constă în scrierea celor k simboluri de informație și apoi suplimentarea lor cu $n-k$ simboluri de control. Aceasta corespunde matricii generatoare (unice)

$$G = (I|B)$$

unde I este matricea unitate de ordin k, iar B este o matrice cu k linii și $n-k$ coloane.

Definiția 2.3 *Un cod liniar este numit "sistematic" dacă admite o matrice generatoare de forma $G = (I|B)$ unde I este matricea unitate.*

Definiția 2.4 *Două coduri liniare C și C' de aceeași lungime n se numesc echivalente dacă există o permutare $\pi \in S_n$ astfel încât*

$$v_1 v_2 \dots v_n \in C \iff v_{\pi(1)} v_{\pi(2)} \dots v_{\pi(n)} \in C'$$

(s-a notat cu S_n mulțimea permutărilor de n elemente).

Exemplul 2.4 *Codul din Exemplul 2.1 este un cod sistematic. Din modalitatea de codificare se observă că mesajul de informație se află în cuvântul - cod pe primele două poziții.*

Codul cu repetiție din Exemplul 2.3 nu este sistematic. Totuși, folosind permutarea $(1,4,6,2,5,3)$ (se permută simbolurile doi cu patru și trei cu șase) se ajunge la un cod sistematic generat de matricea

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

Teorema 2.1 *Orice cod liniar este echivalent cu un cod liniar sistematic.*

Demonstraţie: Matricea generatoare G a unui (n, k) - cod liniar $A_{n,k}$ este de rang k; deci ea are k coloane liniar independente.

1. Să presupunem că primele k coloane ale lui G sunt liniar independente. Deci $G = (X|Y)$ unde X este o matrice $k \times k$ inversabilă. Există atunci o succesiune de operaţii de linii care transformă X în matricea unitate. Aceeaşi succesiune de operaţii efectuate acum pentru toată matricea G va conduce la matricea $G' = (I|Y')$. Deoarece şi G' este matrice generatoare pentru codul $A_{n,k}$, rezultă că acesta este sistematic.

2. Fie $(p_1, p_2, \ldots, p_n)$ permutarea care aduce coloanele liniar independente ale matricii G pe primele k poziţii. Se obţine în acest fel o nouă matrice G'. Fie $A'_{n,k}$ codul liniar obţinut prin codificarea cu matricea generatoare G'. Atunci $A_{n,k}$ şi $A'_{n,k}$ sunt echivalente, iar A' este sistematic, conform cazului anterior. □

2.2 Matrice de control

Definiţia 2.5 *Fie $A_{n,k}$ un cod liniar generat de matricea $G = G_{k,n}$. Se numeşte "matrice de control" o matrice $H = H_{n-k,n}$ cu proprietatea $GH^T = \mathbf{0}$.*

Observaţia 2.2

- *Din definiţie rezultă că matricea de control H a unui cod liniar C are următoarea proprietate:*

$$\mathbf{v} \in C \iff \mathbf{v}H^T = \mathbf{0}. \quad (1)$$

Lăsăm ca exerciţiu demonstrarea acestei echivalenţe.

- *Prin transpunere, relaţia de sus se poate scrie şi $HG^T = \mathbf{0}$. Aceasta înseamnă că şi H este matricea generatoare a unui $(n, n-k)$ - cod liniar peste corpul Z_q, cod pentru care G este matrice de control.*

Cele două coduri astfel definite se numesc coduri duale. Cuvintele - cod din cele două coduri duale sunt ortogonale (produsul lor scalar este zero). Într-adevăr, dacă C şi C' sunt două coduri duale, generate de matricile G respectiv H, iar $\mathbf{x} \in C$, $\mathbf{y} \in C'$ sunt cuvinte - cod arbitrare, există $\mathbf{u}, \mathbf{v}$ cu $\mathbf{x} = \mathbf{u}G$, $\mathbf{y} = \mathbf{v}H$. În plus, $\mathbf{x}H^T = \mathbf{0}$, $\mathbf{y}G^T = \mathbf{0}$.

Atunci, $\mathbf{x}\mathbf{y}^T = \mathbf{u}G\mathbf{y}^T = \mathbf{u}(\mathbf{y}G^T)^T = \mathbf{u}\mathbf{0}^T = \mathbf{0}$.

Exemplul 2.5 $(7,4)$ - *codul liniar binar cu matricea generatoare*

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

are drept matrice de control

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

care la rândul ei este matricea generatoare a unui $(7,3)$ - *cod liniar binar.*

Se verifică imediat relaţia

$$GH^T = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Un cod care coincide cu codul său dual se numeşte *cod auto - dual*.

Teorema 2.2 *Un cod liniar sistematic cu matricea generatoare* $G = (I|B)$ *admite ca matrice de control* $H = (-B^T|I)$.

Demonstraţie: Cele două matrici unitate din scrierea lui G şi H sunt de ordin k respectiv $n-k$. Efectuând calculele, se obţine:

$$GH^T = (I|B)\begin{pmatrix} -B \\ I \end{pmatrix} = -IB + BI = -B + B = \mathbf{0}.$$

□

Corolarul 2.1 *Matricea de control a unui* (n,k)*-cod liniar are rangul* $n-k$.

Teorema 2.2 permite un algoritm de calcul extrem de simplu al matricii de control, atunci când se cunoaşte matricea generatoare. Să arătăm aceasta pe un exemplu:

Exemplul 2.6 *Plecând de la matricea generatoare construită în Exemplul 2.4, se poate construi matricea de control (calculele se fac în Z_3):*

$$H^* = \begin{pmatrix} -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Aplicând acum permutarea inversă coloanelor lui H^*, se ajunge la matricea de control a codului definit în Exemplul 2.3:

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

Relaţia $\mathbf{x}H^T = \mathbf{0}$ pe care o verifică orice cuvânt - cod $\mathbf{x} \in A_{n,k}$ permite să definim un cod şi sub forma unui sistem de ecuaţii. Astfel, $A_{n,k}$ este un cod peste Z_q dacă şi numai dacă elementele sale sunt soluţii ale sistemului liniar $\mathbf{x}H^T = \mathbf{0}$.

Exemplul 2.7 *Codul cu matricea de control* $H = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix}$ *este definit ca mulţimea soluţiilor binare $(x_1, x_2, x_3, x_4, x_5)$ ale sistemului*

$$x_1 + x_2 + x_4 = 0, \qquad x_1 + x_3 + x_4 + x_5 = 0.$$

Sistemul are rangul 2 şi 5 necunoscute, deci el admite $2^{5-2} = 8$ soluţii, care formează codul liniar

$$A_{5,2} = \{00000, 00101, 10010, 01011, 01110, 11001, 11100, 10111\}.$$

2.3 Sindrom

Fie codul liniar $A_{n,k}$ peste Z_q, cu matricea de control H şi $\mathbf{a} \in Z_q^n$. Se numeşte *sindrom* al cuvântului $\mathbf{a}$, secvenţa $\mathbf{z}$ cu $n-k$ componente, obţinută prin relaţia

$$\mathbf{z}^T = H\mathbf{a}^T,$$

sau - echivalent - $\mathbf{z} = \mathbf{a}H^T$.

Observaţia 2.3 $\mathbf{z} = \mathbf{0} \Longleftrightarrow \mathbf{a} \in A_{n,k}$.

Dacă se transmite printr-un canal de comunicaţie un cuvânt $\mathbf{a}$ pentru care sindromul corespunzător verifică relaţia $\mathbf{z} = \mathbf{a}H^T \neq \mathbf{0}$, înseamnă că *s-a detectat* faptul că în timpul transmisiei au apărut erori.

Teorema 2.3 *În Z_2, sindromul recepţionat este egal cu suma coloanelor din matricea de control, corespunzătoare poziţiilor perturbate.*

Demonstraţie: Fie $\mathbf{a} = (a_1, a_2, \ldots, a_n) \in A_{n,k}$ cuvântul-cod transmis. Fără a restrânge generalitatea, să presupunem că au intervenit trei erori, pe poziţiile i, j, k, fiind recepţionat vectorul

$$\mathbf{a}' = (a_1, \ldots, a_{i-1}, a_i', a_{i+1}, \ldots, a_{j-1}, a_j', a_{j+1}, \ldots, a_{k-1}, a_k', a_{k+1}, \ldots, a_n)$$

unde $a_i' \neq a_i$, $a_j' \neq a_j$, $a_k' \neq a_k$. Avem

$\mathbf{0} = \mathbf{a}H^T = a_1(h_1) + \ldots + a_i(h_i) + \ldots + a_j(h_j) + \ldots + a_k(h_k) + \ldots + a_n(h_n)$

pentru că $\mathbf{a} \in A_{n,k}$, iar $(h_1), \ldots, (h_n)$ sunt coloanele matricii H.

Avem, de asemenea

$\mathbf{a}'H^T = a_1(h_1) + \ldots + a_i'(h_i) + \ldots + a_j'(h_j) + \ldots + a_k'(h_k) + \ldots + a_n(h_n)$.

Prin adunarea acestor două egalităţi se obţine:

$\mathbf{z}' = \mathbf{a}'H^T = \mathbf{a}H^T + \mathbf{a}'H^T = (0) + \ldots + (0) + (h_i) + \ldots + (h_j) + \ldots + (h_k) + (0) + \ldots + (0) = (h_i) + (h_j) + (h_k)$. □

2.4 Pondere, distanţă Hamming

Pentru orice cuvânt $\mathbf{x} \in Z_q^n$, se numeşte *pondere* numărul $w(\mathbf{x})$ de elemente diferite de 0 ale lui $\mathbf{x}$. Evident, $0 \leq w(\mathbf{x}) \leq n$.

Pentru două cuvinte $\mathbf{x}, \mathbf{y} \in Z_q^n$, se numeşte *distanţa Hamming* între ele, numărul

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

Propoziţia 2.1 *d este o distanţă definită pe Z_q.*

Demonstraţie: Se verifică imediat proprietăţile unei distanţe:

1. $d(\mathbf{x}, \mathbf{x}) = 0$;
2. $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$;
3. $d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}) \geq d(\mathbf{x}, \mathbf{z})$.

□

Observaţia 2.4 *Deoarece* $\mathbf{x} - \mathbf{y} \in Z_q^n$, *rezultă că distanţa Hamming dintre două cuvinte este ponderea unui cuvânt din* Z_q^n.

Cu ajutorul lui d, Z_q^n *se poate structura ca spaţiu metric.*

Definiţia 2.6 *Se numeşte distanţă (Hamming) minimă a codului liniar* $A_{n,k} \subseteq Z_q^n$, *cea mai mică distanţă (Hamming) dintre elementele codului* $A_{n,k}$, *adică*

$$d = \min_{\mathbf{x},\mathbf{y} \in A_{n,k}, \mathbf{x} \neq \mathbf{y}} d(\mathbf{x}, \mathbf{y}).$$

Folosind proprietatea anterioară şi faptul că $\mathbf{0} \in A_{n,k}$, rezultă că

$$d = \min_{\mathbf{x} \in A_{n,k}, \mathbf{x} \neq \mathbf{0}} w(\mathbf{x}).$$

Adesea, distanţa minimă d este introdusă printre parametrii generali ai codului, folosindu-se notaţia "(n, k, d) - cod liniar".

Teorema 2.4 *Fie* H *matricea de control a unui cod liniar* $A_{n,k}$. *Codul are distanţa minimă* d *dacă şi numai dacă orice combinaţie liniară de* $d - 1$ *coloane ale lui* H *este liniar independentă şi există cel puţin o combinaţie liniară de* d *coloane liniar dependente.*

Demonstraţie: Pentru orice cuvânt $\mathbf{a}$, cu $w(\mathbf{a}) = s$, $\mathbf{a}H^T$ este o combinaţie liniară de s coloane ale lui H. Cum există un cuvânt - cod de pondere minimă egală cu distanţa d a codului, rezultă că avem cel puţin o combinaţie de d coloane liniar dependente ale lui H. O combinaţie de mai puţin de d coloane liniar dependente ar conduce la aflarea unui cuvânt - cod nenul de pondere mai mică decât d, deci contradicţie. □

Definiţia 2.7 *Pentru* $r > 0$ *şi* $\mathbf{x} \in Z_q^n$, *definim sfera de rază* r *şi centru* $\mathbf{x}$ *ca fiind*

$$S_r(\mathbf{x}) = \{\mathbf{y} \in Z_q^n \mid d(\mathbf{x}, \mathbf{y}) \leq r\}.$$

Teorema 2.5 *Fie* $A_{n,k} \subseteq Z_q^n$ *un cod liniar cu distanţa Hamming* d. *Dacă* $r = \left\lfloor \frac{d-1}{2} \right\rfloor$, *atunci*

$$\forall \mathbf{x}, \mathbf{y} \in A_{n,k} \left[\mathbf{x} \neq \mathbf{y} \implies S_r(\mathbf{x}) \cap S_r(\mathbf{y}) = \emptyset \right].$$

Demonstrație: Presupunem prin absurd că există $\mathbf{z} \in Z_q^n$, $\mathbf{z} \in S_r(\mathbf{x}) \cap S_r(\mathbf{y})$. Atunci $d(\mathbf{x},\mathbf{z}) \leq r$, $d(\mathbf{y},\mathbf{z}) \leq r$ deci, conform inegalității triunghiului, $d(\mathbf{x},\mathbf{y}) \leq d(\mathbf{x},\mathbf{z}) + d(\mathbf{y},\mathbf{z}) \leq 2r = 2\left[\frac{d-1}{2}\right] < d$, ceea ce contrazice afirmația că d este distanța minimă a codului $A_{n,k}$. □

Evident, în fiecare astfel de sferă există un singur cuvânt - cod: cel aflat în centrul sferei.

Definiția 2.8 *Un cod liniar $A_{n,k} \subseteq Z_q^n$, de distanță minimă d, este "perfect" dacă*

$$Z_q^n = \bigcup_{\mathbf{x} \in A_{n,k}} S_t(\mathbf{x}).$$

unde $t = \left[\frac{d-1}{2}\right]$.

2.5 Detectare și corectare de erori

Fie $A_{n,k} \subseteq Z_q^n$ un cod liniar, de distanță minimă d și $t = \left[\frac{d-1}{2}\right]$. Să presupunem că s-a recepționat cuvântul $\mathbf{z} \in Z_q^n$; va exista cel mult un cuvânt $\mathbf{x} \in A_{n,k}$ astfel încât $\mathbf{z} \in S_t(\mathbf{x})$ (în cazul codurilor perfecte, acest cuvânt există totdeauna).

În cazul (ideal) când $\mathbf{z} = \mathbf{x}$, cuvântul a fost transmis fără erori (sau cu erori nedetectabile).

Dacă $\mathbf{z} \neq \mathbf{x}$, atunci mesajul a fost perturbat (și avem o detectare de erori); în ipoteza că numărul de erori apărute este minim și există un cuvânt $\mathbf{x} \in A_{n,k}$ astfel ca $d(\mathbf{x},\mathbf{z}) \leq t$, atunci $\mathbf{z}$ provine din cuvântul - cod $\mathbf{x}$ - și se va transforma în acesta prin corectarea corespunzătoare a erorilor.

În celelalte cazuri, $\mathbf{z}$ nu se poate corecta sau se corectează greșit, în alt cuvânt cod.

Această ultimă situație nu apare la codurile perfecte.

Metoda de detectare și corectare a erorilor descrisă mai sus se numește *decodificarea cea mai probabilă.* Pentru marea majoritate a codurilor liniare aceasta este singura metodă utilizată.

Fie $A_{n,k} \subseteq Z_q^n$ un cod liniar. Pentru orice cuvânt $\mathbf{e} \in Z_q^n$ formăm mulțimea

$$V_{\mathbf{e}} = \mathbf{e} + A_{n,k} = \{\mathbf{e} + \mathbf{v} | \mathbf{v} \in A_{n,k}\}$$

$V_{\mathbf{e}}$ este mulţimea cuvintelor $\mathbf{w} = \mathbf{e} + \mathbf{v}$ care pot fi recepţionate la transmiterea cuvântului - cod $\mathbf{v}$, atunci când a acţionat un vector - eroare $\mathbf{e}$. $\mathbf{e}$ va fi numit *eroare - tip.*

Se verifică imediat că $V_{\mathbf{e}}$ este subspaţiu liniar al lui Z_q^n (vezi şi Observaţia 2.1).

În particular, $A_{n,k} = \mathbf{0} + A_{n,k} = V_{\mathbf{0}}$.

Propoziţia 2.2 *Pentru orice* $\mathbf{a} \in V_{\mathbf{e}}$, $V_{\mathbf{a}} = V_{\mathbf{e}}$.

Demonstraţie: Din $\mathbf{a} \in V_{\mathbf{e}}$, rezultă că există $\mathbf{x} \in A_{n,k}$ cu $\mathbf{a} = \mathbf{e} + \mathbf{x}$. Deoarece $\mathbf{x} + A_{n,k} = A_{n,k}$ (evident, $A_{n,k}$ fiind subspaţiu liniar), avem $V_{\mathbf{a}} = \mathbf{a} + A_{n,k} = \mathbf{e} + \mathbf{x} + A_{n,k} = \mathbf{e} + A_{n,k} = V_{\mathbf{e}}$. □

Vom utiliza această propoziţie pentru definirea unei tehnici generale de decodificare.

Dacă vrem să detectăm o anumită eroare - tip $\mathbf{e}$ care modifică cuvintele din $A_{n,k}$ în cuvintele subspaţiului $V_{\mathbf{e}}$, atunci vor fi mai uşor de depistat cuvintele din $V_{\mathbf{e}}$ cu ponderea minimă.

Pentru fiecare $V_{\mathbf{e}}$ se alege un cuvânt numit *reprezentantul* lui $V_{\mathbf{e}}$; acesta este un element cu cea mai mică pondere din $V_{\mathbf{e}}$.

Se construieşte următorul tablou (numit *tablou standard*):

1. Pe prima linie se scriu cuvintele - cod, începând cu $\mathbf{0}$ (reprezentantul lui $A_{n,k} = V_{\mathbf{0}}$);
2. Pe prima coloană se scriu reprezentanţii $\mathbf{0}, \mathbf{e_1}, \mathbf{e_2}, \ldots$;
3. Pe linia cu reprezentantul $\mathbf{e_i}$, sub cuvântul cod $\mathbf{x_j}$ se scrie cuvântul $\mathbf{e_i} + \mathbf{x_j}(\bmod q)$.

Exemplul 2.8 *Fie matricea generatoare* $G = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}$ *peste* Z_2 *Ea va codifica* $Z_2^2 = \{00, 01, 10, 11\}$ *în* $A_{4,2} = \{0000, 1001, 0111, 1110\}$.

Calculul mulţimilor $V_{\mathbf{e}}$ *conduce la*

$$\begin{array}{cccccccc} V_{0000} & = & V_{1001} & = & V_{0111} & = & V_{1110} \\ V_{1000} & = & V_{0001} & = & V_{0110} & = & V_{1111} \\ V_{0100} & = & V_{1101} & = & V_{0011} & = & V_{1010} \\ V_{0010} & = & V_{1011} & = & V_{0101} & = & V_{1100} \end{array}$$

Alegem ca reprezentanţi pe 0000, 1000 *(se poate şi* 0001), 0100, 0010. *Tabloul standard va fi:*

0000	1001	0111	1110
1000	0001	1111	0110
0100	1101	0011	1010
0010	1011	0101	1100

Pentru destinatar, acest tablou este ca un dicţionar care se utilizează astfel: cuvântul primit **a** *se decodifică în cuvântul - cod din capul coloanei pe care se află* **a**.

De exemplu, dacă se recepţionează 1101, *el se va decodifica în* 1001.

Evident, cuvintele de pe prima coloană se decodifică în ele însele (sunt cuvinte - cod şi nu au fost perturbate de nici o eroare).

Pentru orice cod liniar, un dicţionar complet de tipul celui de mai sus constituie cea mai simplă metodă de decodificare.

Problema apare atunci când într-o mulţime $V_{\mathbf{e}}$ sunt mai multe cuvinte de pondere minimă. Atunci tabela va decodifica corect numai eroarea - tip aleasă ca reprezentant.

Astfel, revenind la Exemplul 2.8, cuvântul recepţionat 1111 se decodifică în 0111 (considerând că a fost alterat primul caracter).

Dacă se ia însă drept reprezentant pe linia a doua 0001 în loc de 1000, a doua linie din tabloul standard este

0001 1000 0110 1111

Atunci, 1111 se decodifică în 1110 (considerînd ultimul caracter ca fiind cel alterat de canalul de transmisie). Care este cuvântul - cod corect transmis ? Acest lucru nu poate fi decis. Singurul lucru care poate fi făcut este să se aleagă drept reprezentanţi erorile - tip cele mai probabile.

Pentru un (n, k) - cod peste Z_q, un dicţionar complet constă din toate cele q^n cuvinte posibile, lucru destul de dificil deoarece în practică codurile sunt destul de lungi (de obicei $n \geq 100$). De aceea este utilizată o altă manieră de lucru, care reduce mult mărimea tabloului standard.

Fie H matricea de control a unui (n, k) - cod liniar $A_{n,k}$; putem considera (Corolarul 2.1) că liniile lui H sunt vectori liniar independenţi.

Teorema 2.6 *Pentru orice* $\mathbf{e} \in Z_q^n$, *toate cuvintele din* $V_{\mathbf{e}}$ *au acelaşi sindrom.*

Demonstraţie: Fie $\mathbf{v} \in A_{n,k}$ arbitrar şi $\mathbf{w} = \mathbf{e} + \mathbf{v}$. Avem

$$\mathbf{s} = \mathbf{w}H^T = (\mathbf{e} + \mathbf{v})H^T = \mathbf{e}H^T + \mathbf{v}H^T = \mathbf{e}H^T + \mathbf{0} = \mathbf{e}H^T.$$

Deci toate cuvintele din $V_\mathbf{e}$ au sindromul egal cu sindromul lui $\mathbf{e}$. □

Invers, pentru fiecare sindrom – deci pentru fiecare cuvânt $\mathbf{s}$ de lungime $n - k$, se poate determina un vector - eroare $\mathbf{e}$ având sindromul $\mathbf{s}$.

Mai mult, $\mathbf{s}$ va fi ales astfel încât să aibă pondere minimă (conform decodificării cele mai probabile). Pentru aceasta, se rezolvă sistemul de ecuaţii liniare $H\mathbf{e}^T = \mathbf{s}^T$, care are soluţie (deoarece liniile lui H sunt liniar independente). Din mulţimea soluţiilor alegem una de pondere minimă, cu ajutorul căreia construim mulţimea $V_\mathbf{e}$ a tuturor cuvintelor de sindrom $\mathbf{s}$.

Pe baza celor de mai sus, se poate folosi următoarea procedură de decodificare:

1. La recepţionarea unui cuvânt $\mathbf{w}$, se calculează sindromul $\mathbf{s}$:
$$\mathbf{s}^T = H\mathbf{w}^T.$$
2. Se află eroarea - tip $\mathbf{e}$ cu sindromul $\mathbf{s}$.
3. Se consideră cuvântul - cod corect ca fiind $\mathbf{v} = \mathbf{w} - \mathbf{e}$.

În acest fel, nu mai este necesar să se reţină tot tabloul standard; este suficient să se ştie doar reprezentanţii subspaţiilor $V_\mathbf{e}$ şi sindromurile corespunzătoare.

Exemplul 2.9 *Reluând codul definit în Exemplul 2.8, el are ca matrice de control*

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

Calculând sindromurile reprezentanţilor, se ajunge la tabloul:

Sindrom	*Reprezentant*
00	0000
01	1000
10	0010
11	0100

care reprezintă o reducere cu 50% *a datelor stocate (comparativ cu tabelul standard).*

De exemplu, la recepţionarea cuvântului 1101 *se calculează sindromul*

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

Reprezentantul sindromului 11 *este* 0100. *Efectuând operaţia*

$$1101 - 0100 = 1101 + 0100 = 1001$$

(în Z_2 *scăderea se poate înlocui cu adunarea) se ajunge la cuvântul transmis, anume* 1001.

Exemplul 2.10 *Să considerăm codul liniar* $A_{5,3}$ *peste* Z_3, *definit de matricea de control*

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

Sindromurile sunt cuvinte de lungime 2 *peste* Z_3, *deci în total nouă cuvinte. Lista sindromurilor şi a reprezentanţilor este:*

Sindrom	*Reprezentant*
00	00000
10	10000
01	01000
21	00100
12	00010
02	00001
20	20000
11	11000
22	22000

De remarcat că un tablou standard pentru acest cod are dimensiunea 9×27 *şi conţine* 243 *cuvinte; volumul de date s-a redus deci cu* 92%.

Să presupunem că s-a trimis cuvântul cod 01221 *şi s-a recepţionat* 01201. *Sindromul este*

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

deci $\mathbf{e} = 00010$*; decodificarea este*

$$\mathbf{v} = \mathbf{w} - \mathbf{e} = 01201 - 00010 = 01221.$$

Decodificarea a fost corectă deoarece eroarea - tip apărută a fost una din cele selectate prin reprezentanţi.

Astfel, dacă pentru acelaşi cuvânt cod trimis s-ar fi recepţionat 11201*, sindromul calculat ar fi* 22*; folosind reprezentantul din tabelă se ajunge la un cuvânt - cod*

$$\mathbf{v} = \mathbf{w} - \mathbf{e} = 11201 - 22000 = 22201,$$

care nu coincide cu ce s-a transmis.

Pentru sindomul 22 *se poate alege însă un alt reprezentant, tot de pondere doi:* 10010*. Dacă îl folosim pe acesta în tabela de sindromuri, se obţine o decodificare corectă:*

$$\mathbf{v} = \mathbf{w} - \mathbf{e} = 11201 - 10010 = 01221.$$

De fapt, codul definit nu are capacitatea de a corecta mai mult de o eroare; faptul că am solicitat să corecteze două erori a condus la incertitudine.

2.6 Capacităţi de detectare şi corectare de erori

După cum s-a văzut până acum, un cod liniar este definit ca un spaţiu liniar, codul dual este complementul său ortogonal etc. Ceea ce interesează aici este identificarea unor coduri cu proprietăţi deosebite în detectarea şi corectarea erorilor. Aceste proprietăţi sunt legate în special de distanţa minimă a codului. Vom prezenta pentru început câteva rezultate legate de diverse margini relativ la distanţă, numărul de simboluri de control, informaţie etc.

Teorema 2.7 *Un cod liniar $A_{n,k} \subseteq Z_q^n$ are distanţa minimă d dacă şi numai dacă poate detecta orice combinaţie de maxim $d-1$ erori.*

Demonstraţie: Dacă se transmite un cuvânt $\mathbf{a} \in A_{n,k}$ şi apar t $(0 \leq t < d)$ erori, se va recepţiona cuvântul $\mathbf{a} + \mathbf{e} \in Z_q^n$ cu $w(\mathbf{e}) = t$.

Deoarece $d(\mathbf{a}, \mathbf{a}+\mathbf{e}) = t$, rezultă $\mathbf{a}+\mathbf{e} \notin A_{n,k}$, deci se detectează eroare.

Afirnaţia reciprocă este evidentă. □

Teorema 2.8 *Un cod liniar $A_{n,k} \subseteq Z_q^n$ are distanţa minimă d dacă şi numai dacă poate corecta orice combinaţie de $t \leq \left[\frac{d-1}{2}\right]$ erori.*

Demonstraţie: Fie t un număr întreg pozitiv şi să presupunem că $d \leq 2t$; vom arăta că în acest caz există erori - tip de pondere t (numite şi *pachete de t erori independente*) care nu pot fi corectate de codul $A_{n,k}$. Aceasta va duce la concluzia că $d \geq 2t+1$, adică $A_{n,k}$ poate corecta orice pachet de $t \leq \left[\frac{d-1}{2}\right]$ erori.

Fie $\mathbf{a}, \mathbf{b} \in A_{n,k}$ cu $d(\mathbf{a}, \mathbf{b}) = d$ şi $i_1, i_2, \ldots, i_d$ toţi indicii în care $\mathbf{a}$ diferă de $\mathbf{b}$. Alegând $t = \left[\frac{d+1}{2}\right]$, avem $\frac{d}{2} \leq t \leq \frac{d+1}{2}$ deci $d \leq 2t$ (t astfel ales este minim).

Să presupunem că se trimite cuvântul - cod $\mathbf{a}$ şi se recepţionează cuvântul $\mathbf{a}' = (a'_1, a'_2, \ldots, a'_n)$ unde

$$a'_i = \begin{cases} a_i\ (= b_i) & \text{dacă} \quad i \neq i_1, i_2, \ldots, i_d, \\ b_i & \text{dacă} \quad i = i_1, i_3, \ldots, \\ a_i & \text{dacă} \quad i = i_2, i_4, \ldots \end{cases}$$

Atunci, evident $d(\mathbf{a}', \mathbf{a}) = \left[\frac{d+1}{2}\right] = t$ şi $d(\mathbf{a}', \mathbf{b}) = \left[\frac{d}{2}\right] \leq t = d(\mathbf{a}', \mathbf{a})$, ceea ce duce la decodificarea lui $\mathbf{a}$ în $\mathbf{b}$ – incorect.

Să presupunem acum $d \geq 2t+1$. Atunci codul $A_{n,k}$ poate corecta orice pachet de t erori. Pentru a arăta aceasta, să presupunem că se trimite un cuvânt - cod $\mathbf{a}$ şi se primeşte un cuvânt $\mathbf{a}'$ cu $d(\mathbf{a}, \mathbf{a}') \leq t$. Pentru orice cuvânt - cod $\mathbf{b}$ (deci cu $d(\mathbf{a}, \mathbf{b}) \geq d \geq 2t+1$) avem, conform inegalităţii triunghiului:

$$d(\mathbf{a}, \mathbf{a}') + d(\mathbf{a}', \mathbf{b}) \geq d(\mathbf{a}, \mathbf{b}) \geq 2t+1,$$

sau, $d(\mathbf{a}', \mathbf{b}) \geq 2t+1-d(\mathbf{a}, \mathbf{a}') \geq 2t+1-t = t+1 > d(\mathbf{a}', \mathbf{a})$.

Deci, în ipoteza decodificării cele mai probabile, $\mathbf{a}'$ se va decodifica în $\mathbf{a}$. □

Teorema 2.9 *Distanţa minimă a unui (n,k) - cod liniar verifică relaţia*

$$d \leq n - k + 1.$$

Demonstraţie: Vom separa demonstraţia în două etape:

A: Fie $A_{n,k}$ un cod sistematic. Atunci primele k simboluri din orice cuvânt - cod pot fi alese arbitrar (ele formează mesajul de informaţie, un element din Z_q^k). Fie $\mathbf{v} \in A_{n,k}$ cuvântul de forma

$$\mathbf{v} = 100\ldots0v_{k+1}v_{k+2}\ldots v_n.$$

Evident, $0 < w(\mathbf{v}) \leq n - k + 1$. Cum d este cea mai mică pondere a unui cuvânt - cod nenul, rezultă inegalitatea cerută.

B: Fie $A_{n,k}$ un cod liniar arbitrar şi $A'_{n,k}$ codul liniar sistematic echivalent. Se observă că cele două coduri au aceiaşi parametri n, k, d. Folosind acum **A**, inegalitatea se obţine din nou. □

Teorema 2.10 *În orice cod liniar $A_{n,k} \subseteq Z_q^n$ avem*

$$d \leq \frac{nq^{k-1}(q-1)}{q^k - 1} \qquad \textit{(marginea Plotkin)}$$

Demonstraţie: Să considerăm elementele din $A_{n,k}$ aşezate ca linii ale unui tablou. Se obţine un tablou cu q^k linii şi n coloane. Fiecare componentă nenulă din Z_q apare pe fiecare coloană în q^{k-1} linii. Atunci, suma ponderilor tuturor cuvintelor - cod este egală cu $nq^{k-1}(q-1)$ deoarece fiecare componentă nenulă apare de q^{k-1} ori în fiecare coloană şi avem $q-1$ componente nenule distribuite pe n coloane.

Distanţa minimă a codului nu poate să depăşească ponderea medie a cuvintelor codului, adică

$$d \leq \frac{nq^{k-1}(q-1)}{q^k - 1}$$

deoarece în $A_{n,k}$ sunt $q^k - 1$ cuvinte nenule. □

Teorema 2.11 *Fie $A_{n,k}$ un cod liniar peste Z_q care corectează orice combinaţie de maxim t erori. Într-un asemenea cod sunt necesare cel puţin*

$$n - k \geq log_q[1 + C_n^1(q-1) + C_n^2(q-1)^2 + \ldots + C_n^t(q-1)^t]$$

poziţii de control (marginea Hamming).

Demonstraţie: Pentru ca $A_{n,k}$ să corecteze orice combinaţie de cel mult t erori independente, este necesar ca fiecare astfel de eroare - tip să fie un reprezentant din tabloul standard, deci să fie caracterizată printr-un sindrom distinct. Sunt q^{n-k} sindromuri distincte, deci acesta este numărul maxim de erori care pot fi corectate de cod. Din cele q^{n-k} sindromuri, unul trebuie să fie pentru 0 erori, $C_n^1(q-1)$ - pentru erori - tip simple (cuvinte $\mathbf{e}$ cu o singură componentă nenulă), $C_n^2(q-1)^2$ pentru erori duble etc. Deci, este necesar ca

$$q^{n-k} \geq 1 + C_n^1(q-1) + C_n^2(q-1)^2 + \ldots + C_n^t(q-1)^t.$$

Apoi se logaritmează. □

Teorema 2.12 *Dacă*

$$n - k \geq log_q[1 + C_{n-1}^1(q-1) + C_{n-1}^2(q-1)^2 + \ldots + C_{n-1}^{d-2}(q-1)^{d-2}]$$

atunci există un cod liniar $A_{n,k} \subseteq Z_q^n$ cu distanţa minimă d (marginea Varşamov - Gilbert).

Demonstraţie: Pentru ca să existe un cod liniar $A_{n,k}$ peste Z_q cu distanţa minimă d este suficient (Teorema 2.4) ca orice coloană din matricea de control $H_{n-k,n}$ să nu fie combinaţie liniară a altor $d-2$ coloane, în acest fel ne-existând nici o combinaţie liniară între $d-1$ coloane nenule ale lui H. Această condiţie este echivalentă cu

$$q^{n-k} - 1 \geq C_{n-1}^1(q-1) + C_{n-1}^2(q-1)^2 + \ldots + C_{n-1}^{d-2}(q-1)^{d-2}.$$

Aici, $q^{n-k} - 1$ reprezintă numărul total de coloane distincte nenule care pot apare în matricea H. Semnificaţia termenilor din membrul drept este evidentă: astfel, de exemplu $C_{n-1}^2(q-1)^2$ reprezintă numărul combinaţiilor liniare cu coeficienţi nenuli a două din cele $n-1$ coloane etc.

Apoi se logaritmează. □

Fie $A_{n,k} \subseteq Z_q^n$ un cod liniar pentru care $d \geq 2t+1$ (deci cu capacitatea de a corecta orice combinaţie de maxim t erori independente). Reamintim că pentru orice $\mathbf{x} \in A_{n,k}$ s-a definit sfera centrată în $\mathbf{x}$ prin

$$S_t(\mathbf{x}) = \{\mathbf{y} \in Z_q^n | d(\mathbf{x}, \mathbf{y}) \leq t\}.$$

Mai introducem şi suprafaţa (scoarţa) acestei sfere, definită:

$$A_t(\mathbf{x}) = \{\mathbf{y} \in Z_q^n | d(\mathbf{x}, \mathbf{y}) = t\}.$$

Vom nota numărul de elemente ale fiecăreia din cele două mulţimi prin

$$S_t = card(S_t(\mathbf{x})), \quad A_t = card(A_t(\mathbf{x}))$$

(datorită proprietăţii de omogenitate a spaţiilor liniare, valorile S_t şi A_t sunt aceleaşi pentru orice $\mathbf{x} \in Z_q^n$).

Au loc relaţiile evidente:

$$A_t \leq S_t, \qquad S_t = \sum_{i=0}^{t} A_i.$$

Deoarece sferele de rază t centrate în cuvintele codului $A_{n,k}$ sunt disjuncte, avem

$$q^k S_t \leq q^n.$$

Aici, q^k reprezintă numărul de cuvinte - cod, iar q^n - numărul total de cuvinte din Z_q^n.

Din această relaţie se obţine imediat

$$n - k \geq log_q S_t,$$

cunoscută sub numele de *inegalitatea volumului.* Ea mai poate fi găsită şi sub forma

$$\frac{k}{n} \leq 1 - \frac{1}{n} log_q S_t.$$

Raportul $\frac{k}{n}$ se numeşte *rată de informaţie* şi dă o măsură a cantităţii de informaţie pe care o poartă un cuvânt - cod. O rată de informaţie mică (mai multe simboluri de control) asigură o securitate mai mare a datelor transmise. În schimb, condiţii practice de eficienţă cer o rată de informaţie cât mai mare (mai multă informaţie pe unitatea de mesaj). Aceasta este una din solicitările contradictorii ale teoriei codurilor.

2.7 Modificări ale codurilor liniare

Adesea este imposibil să se utilizeze un cod bun, deoarece el nu satisface anumite restricţii tehnice, cum ar fi lungimea sau rata de informaţie. De aceea este util să se aplice anumite prelucrări asupra codurilor, care să nu afecteze proprietătile principale de detectare şi corectare a erorilor.

Definiţia 2.9 *Numim extensie a unui* (n, k) *- cod liniar* $A_{n,k} \subseteq Z_q^n$, $(n+1, k)$ *- codul liniar* $A^*_{n+1,k}$ *obţinut din* $A_{n,k}$ *prin adăugarea la fiecare*

cuvânt - cod $\mathbf{a} = a_1a_2\ldots a_n$ *a unui simbol nou* a_{n+1}, *cu proprietatea* $\sum_{i=1}^{n+1} a_i = 0 \ (mod\ q)$.

Observaţii:

1. În cazul binar, noul caracter a_{n+1} poartă numele *bit de paritate.*

   Dacă H este matricea de control a codului $A_{n,k}$, atunci codul extins $A^*_{n+1,k}$ are matricea de control

$$H^* = \left(\begin{array}{ccccc|c} & & H & & & 0 \\ & & & & & \vdots \\ & & & & & 0 \\ \hline 1 & 1 & \ldots & 1 & 1 & 1 \end{array}\right)$$

   De fapt, ultima linie reprezintă ecuaţia $x_{n+1} = (q-1)\sum_{i=1}^{n} x_i$.

2. Dacă un cod liniar binar $A_{n,k}$ are o distanţă minimă impară d, atunci codul extins are distanţa minimă $d+1$. Într-adevăr, fie $\mathbf{a} = a_1a_2\ldots a_n \in A_{n,k}$ cu $w(\mathbf{a}) = d$. Cum d este impar, rezultă $\sum_{i=1}^{n} a_i = 1$ (în Z_2), deci $a_{n+1} = 1$. Cuvântul $\mathbf{a}' = a_1a_2\ldots a_n1 \in A^*_{n+1,k}$ are ponderea $w(\mathbf{a}') = d+1$, şi nu se poate construi un alt cuvânt - cod în $A^*_{n+1,k}$ de pondere mai mică.

Definiţia 2.10 *Fie* $A_{n,k} \subseteq Z_q^n$ *un cod liniar.*

1. *"Relaxarea" lui* $A_{n,k}$ *este un cod liniar* $\overline{A_{n-1,k}}$, *obţinut prin ştergerea ultimului simbol din cuvintele lui* $A_{n,k}$;

2. *"Completarea" lui* $A_{n,k}$ *este un cod definit* $A^*_{n,k} = A_{n,k} \cup (A_{n,k} + \mathbf{1})$, *unde* $\mathbf{1}$ *este cuvântul cu toate elementele egale cu* 1, *iar suma se face modulo* q;

3. *"Expurgarea" codului* $A_{n,k}$ *este* $A'_{n,k} = \{\mathbf{a} \in A_{n,k} | w(\mathbf{a}) \equiv 0 \ (mod\ 2)\}$.

Observaţii:

- Relaxarea este operaţia inversă extensiei.

- Prin completarea şi expurgarea codurilor liniare se obţin coduri liniare numai în cazul binar. În celelalte cazuri, noile mulţimi rezultate nu sunt spaţii liniare.

Propoziţia 2.3 *Prin completarea unui cod liniar binar $A_{n,k}$ se obţine un cod liniar binar $A_{n,k+1}$ cu un număr dublu de cuvinte - cod.*

Demonstraţie: Completarea unui cod liniar binar C înseamnă adăugarea la cuvintele - cod ale lui C a tuturor cuvintelor obţinute prin complementare (schimbarea lui 0 în 1 şi a lui 1 în 0).

Fie $G_{k,n}$ matricea generatoare a codului $A_{n,k}$. Se verifică uşor că matricea

$$G'_{k+1,n} = \begin{pmatrix} & G_{k,n} & \\ & & \\ 1 \quad 1 & & 1 \quad 1 \end{pmatrix}$$

generează $A_{n,k} \cup (\mathbf{1} + A_{n,k})$. Cuvintele acestui cod au lungime n şi $k+1$ poziţii de informaţie. Fiecare din cele două submulţimi are un număr egal de elemente. □

Propoziţia 2.4 *Orice cod liniar binar are sau toate cuvintele - cod de pondere pară, sau numărul cuvintelor - cod de pondere pară este egal cu numărul celor de pondere impară.*

Demonstraţie: Fie C un cod liniar binar cu un cuvânt $\mathbf{v_1}$ de pondere impară. Să presupunem că $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ sunt toate cuvintele lui C; atunci $C = C + \mathbf{v_1}$. Pentru orice cuvânt - cod $\mathbf{v_i}$ de pondere pară (impară), $\mathbf{v_1} + \mathbf{v_i}$ are pondere impară (pară). Pentru a justifica această afirmaţie, să presupunem că $w(\mathbf{v_1}) = 2p+1$, $w(\mathbf{v_i}) = 2q$ iar $\mathbf{v_i}$ şi $\mathbf{v_1}$ au 1 pe r poziţii comune. Atunci $w(\mathbf{v_i} + \mathbf{v_1}) = w(\mathbf{v_i}) + w(\mathbf{v_1}) - 2r$ (pentru că $1+1=0$) $= 2p+1+2q-2r = 2s+1$. Dacă $\mathbf{v_i}$ are pondere impară, se procedează în mod similar.

Deci adunarea cu $\mathbf{v_1}$ defineşte o corespondenţă biunivocă între cuvintele - cod de pondere pară şi cele de pondere impară, ceea ce completează demonstraţia. □

Corolarul 2.2 *Expurgarea unui cod liniar binar C este tot C, sau un cod liniar binar având ca mulţime de elemente jumătate din mulţimea elementelor lui C.*

Matricea generatoare G_{exp} a lui C_{exp} se poate obţine din matricea G a lui C astfel: dacă toate liniile lui G sunt vectori de pondere pară cele două coduri coincid. Altfel, fie $G = [\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_r}, \mathbf{e_{r+1}}, \ldots, \mathbf{e_k}]^T$ în care - fără a micşora generalitatea - putem presupune că primele r linii au pondere impară, iar celelalte $k - r$ linii au pondere pară. Atunci $G_{exp} = [\mathbf{0}, \mathbf{e_2} + \mathbf{e_1}, \ldots, \mathbf{e_r} + \mathbf{e_1}, \mathbf{e_{r+1}} \ldots, \mathbf{e_k}]^T$.

Exemplul 2.11 *Să construim codul $A_{4,2}$ peste Z_3, folosind matricea generatoare*

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

Ea codifică cele 9 elemente din Z_3^2 în

$$A_{4,2} = \{0000, 0111, 0222, 1010, 1121, 1202, 2020, 2101, 2212\}.$$

Codul liniar relaxat $A_{3,2} = \{000, 011, 022, 101, 112, 120, 201, 210, 221\}$ este generat de matricea

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

Construcţia a fost posibilă deoarece prin eliminarea ultimei coloane, liniile rămase sunt tot liniar independente. Dacă acest lucru nu este realizabil, se caută k cuvinte - cod în $A_{n,k}$, cu proprietatea că după eliminarea ultimei componente, ele sunt liniar independente. Acestea formează liniile noii matrici generatoare.

Codul completat este

0000	0111	0222	1010	1121	1202	2020	2101	2212
1111	1222	1000	2121	2202	2010	0101	0212	0020

De remarcat că el nu este un spaţiu liniar (nu este închis la adunarea din Z_3).

Codul expurgat are cinci elemente: $\{0000, 1010, 1121, 2020, 2212\}$.

Nici acesta nu este cod liniar.

Exemplul 2.12 *Să reluăm matricea generatoare din Exemplul 2.11, dar pentru un cod liniar peste Z_2.*

Codul generat de G este $A_{4,2} = \{0000, 0111, 1010, 1101\}$.

Toate codurile modificate sunt în acest caz coduri liniare. Astfel

- *Codul relaxat* $A_{rel} = \{000, 011, 101, 110\}$ *este generat de aceeaşi matrice* G_{rel} *din Exemplul 2.11.*

- *Codul completat* $A_{com} = \{0000, 0111, 1010, 1101, 1111, 1000, 0101, 0010\}$, *este un cod liniar generat de matricea*

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

- *Codul expurgat* $A_{exp} = \{0000, 1010\}$ *este un cod liniar generat de matricea*

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

2.8 Detectare şi corectare simultană de erori

Să începem cu un exemplu.

Exemplul 2.13 *Fie* $(7, 4)$ *- codul liniar binar având matricea de control*

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

El are distanţa minimă $d = 3$, *deci poate detecta* 2 *erori şi poate corecta o eroare (Teoremele 2.7 şi 2.8). Totuşi, codul nu poate realiza simultan ambele deziderate.*

Mai precis, atunci când codul este utilizat pentru corectare de erori, erorile duble scapă nedetectate. Astfel, dacă se trimite 0000000 *şi se recepţionează* 1010000, *sindromul este* 010. *Tabloul standard conduce la corectarea celui de-al doilea bit, şi decodifică (incorect) în cuvântul - cod* 111000.

Uneori însă, se solicită în mod explicit un cod capabil să detecteze şi să corecteze erori în acelaşi timp.

Definiţia 2.11 *Un cod* C *de lungime* n *corectează* t *erori şi detectează* s *erori simultan dacă orice cuvânt - cod* $\mathbf{v}$ *are următoarea proprietate:*

$$\forall \mathbf{x} \in Z_q^n \left[d(\mathbf{x}, \mathbf{v}) \leq s \implies \forall \mathbf{a} \in A \setminus \{\mathbf{v}\}, d(\mathbf{x}, \mathbf{a}) > t \right].$$

În această situaţie, detectarea şi corectarea simultană a erorilor se realizează astfel: la recepţionarea unui cuvânt $\mathbf{x} \in Z_q^n$ se caută cel mai apropiat cuvânt - cod $\mathbf{v}$ (în sensul distanţei Hamming). Dacă $d(\mathbf{x}, \mathbf{v}) \leq t$ atunci cuvântul se corectează în $\mathbf{v}$; altfel, se anunţă că cel puţin s simboluri sunt modificate.

Justificarea acestui procedeu rezultă imediat din definiţie.

Teorema 2.13 *Un cod liniar corectează t erori şi detectează s erori simultan dacă şi numai dacă*

$$d \geq t + s + 1.$$

Demonstraţie: **A**: Să presupunem $d \geq t + s + 1$. Fie $\mathbf{v}$ un cuvânt - cod şi $\mathbf{x} \in Z_q^n$ cu $d(\mathbf{v}, \mathbf{x}) \leq s$. Pentru orice cuvânt - cod $\mathbf{v}'$ ($\mathbf{v}' \neq \mathbf{v}$) avem $d(\mathbf{v}, \mathbf{v}') \geq d \geq t + s + 1$. Folosind inegalitatea triunghiului,

$$d(\mathbf{v}, \mathbf{x}) + d(\mathbf{x}, \mathbf{v}') \geq d(\mathbf{v}, \mathbf{v}') \geq t + s + 1,$$

se deduce

$$d(\mathbf{w}, \mathbf{v}') \geq t + s + 1 - d(\mathbf{v}, \mathbf{w}) \geq t + s + 1 - s = t + 1.$$

Deci, condiţia din Definiţia 2.11 este îndeplinită.

B: Să presupunem prin absurd $d < t + s + 1$. Fie $\mathbf{v}, \mathbf{v}'$ două cuvinte - cod cu $d(\mathbf{v}, \mathbf{v}') = d \leq t + s$. Construim cuvântul $\mathbf{x}$ din $\mathbf{v}$ în felul următor: dacă $d \leq s$, se ia $\mathbf{x} = \mathbf{v}'$; altfel, se înlocuiesc primele s simboluri care diferă de cele din $\mathbf{v}'$, cu valorile lor din $\mathbf{v}'$. Atunci $d(\mathbf{v}, \mathbf{x}) = s$ şi $d(\mathbf{v}', \mathbf{x}) = d - s \leq t + s - s = t$, ceea ce contrazice condiţia din Definiţia 2.11. □

Exemplul 2.14 *Să considerăm* $(8, 4)$ *- codul liniar binar generat de matricea*

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

El are distanţa minimă $d = 4$, deci - conform Teoremei 2.8 - poate corecta maxim o eroare, iar conform Teoremei 2.13, poate corecta o eroare şi detecta simultan două erori.

Astfel recepţionarea cuvântului 11110010 *conduce la corectarea sa în* 10110010 *(deoarece* $d(11110010, 10110010) = 1$ *şi* 10110010 *este cuvânt - cod).*

În schimb recepţionarea cuvântului 00001111 *anunţă că au apărut cel puţin două erori. În această situaţie, nu mai puţin de patru cuvinte - cod (*00010111, 00101011, 01001101, 10001110*) sunt situate la distanţa* 2 *de cuvântul primit.*

Exemplul 2.15 *Codul binar cu repetiţie de lungime* 7 *(care are* 2 *elemente:* 0000000 *şi* 1111111*) poate realiza una din condiţiile:*

- *Corectează* 3 *erori;*
- *Detectează* 6 *erori;*
- *Corectează* 2 *erori şi detectează* 4 *erori simultan.*

2.9 Probabilitatea nedetectării erorilor

Să ne punem următoarea problemă: care este probabilitatea ca, după transmiterea unui cuvânt - cod **a**, să fie recepţionat alt cuvânt - cod **b** ($\mathbf{b} \neq \mathbf{a}$). Altfel spus, care este probabilitatea ca o eroare să scape nedetectată ?

Notând cu $\mathbf{e} = \mathbf{b} - \mathbf{a}$, o eroare este nedetectată dacă şi numai dacă **e** este un cuvânt - cod nenul.

Vom considera un canal de transmisie *binar simetric*, adică un canal în care singurele simboluri transmise sunt 0 şi 1, iar probabilitatea p ($0 \leq p \leq 1$) ca la transmiterea lui 0 să fie recepţionat 1 este egală cu probabilitatea ca la transmiterea lui 1 să se recepţioneze 0. Într-un astfel de canal, dacă $w(\mathbf{e}) = i$ (adică au fost perturbate la transmisie i caractere), probabilitatea de apariţie a erorii - tip **e** este $p^i q^{n-i}$, unde $q = 1 - p$. Notând cu A_i numărul cuvintelor - cod cu ponderea i, probabilitatea P_{ned} a unei erori nedetectabile este suma probabilităţilor $p^i q^{n-i}$, fiecare termen apărând de A_i ori pentru $i = 1, 2, \ldots, n$. Formal,

$$P_{ned} = \sum_{i=1}^{n} A_i p^i q^{n-i}$$

Cum $A_1 = A_2 = \ldots = A_{d-1} = 0$, suma se reduce la $P_{ned} = \sum_{i=d}^{n} A_i p^i q^{n-i}$.

Exemplul 2.16 *Să considerăm* $(7, 4)$ *- codul liniar binar din Exemplul 2.13. Cele* $2^4 = 16$ *cuvinte - cod ale sale sunt: un cuvânt de pondere* 0*, câte şapte cuvinte de pondere* 3 *şi* 4 *şi un cuvânt de pondere* 7*. Atunci*

$$P_{ned} = 7p^3q^4 + 7p^4q^3 + p^7.$$

Dacă folosim acest cod într-un canal binar simetric cu eroare de probabilitate $p = 0.01$, *avem*

$$P_{ned} = 7(0.01)^3(0.99)^4 + 7(0.01)^4(0.99)^3 + (0.01)^7 \approx 7 \times 10^{-6},$$

deci în medie - apar cam şapte erori nedetectabile la un milion de cuvinte transmise.

Definiţia 2.12 *Polinomul de variabilă* $x \in [0,1]$ *definit*

$$P(x) = \sum_{i=0}^{n} A_i x^i$$

unde A_i *este numărul de cuvinte de pondere* i *din codul liniar* $A_{n,k}$, *se numeşte "numărătorul de ponderi" al codului* $A_{n,k}$.

Exemplul 2.17 *Codul liniar binar definit în Exemplul 2.12 are numărătorul de ponderi*

$$P(x) = 1 + x^2 + 2x^3.$$

Propoziţia 2.5 *Fie* $A_{n,k}$ *un cod liniar binar cu numărător de ponderi* $P(x)$. *Probabilitatea apariţiei unei erori nedetectabile la folosirea codului* $A_{n,k}$ *într-un canal binar simetric este*

$$P_{ned} = q^n \left[P\left(\frac{p}{q}\right) - 1 \right].$$

Demonstraţie: Relaţia de definiţie a lui P_{ned} se poate rescrie

$$q^n \sum_{i=1}^{n} A_i p^i q^{-i} = q^n \sum_{i=1}^{n} A_i \left(\frac{p}{q}\right)^i.$$

Deoarece $A_0 = 1$ (singurul cuvânt - cod de pondere 0 este cuvântul - cod **0**), expresia devine în continuare

$$P_{ned} = q^n \left[\sum_{i=0}^{n} A_i \left(\frac{p}{q}\right)^i - 1 \right] = q^n \left[P\left(\frac{p}{q}\right) - 1 \right].$$

□

2.10 Identitatea MacWilliams

În acest paragraf vom arăta un rezultat care face posibilă determinarea numărătorului de ponderi $P_{C^\perp}(x)$ al dualului $C^\perp$ unui cod liniar C, direct din numărătorul de ponderi $P_C(x)$ al codului C.

Propoziţia 2.6 *Fie* $\mathbf{x} \in Z_2^n$ *şi* $A_{n,k}$ *un* (n,k) *- cod liniar binar. Atunci are loc egalitatea*

$$\frac{1}{2^k} \sum_{\mathbf{v} \in A_{n,k}} (-1)^{\mathbf{vx}} = \begin{cases} 1 & \text{dacă } \mathbf{x} \in A_{n,k}^{\perp} \\ 0 & \text{altfel} \end{cases}$$

Demonstraţie: Dacă $\mathbf{x} \in A_{n,k}^{\perp}$ atunci evident, $(-1)^{\mathbf{vx}} = (-1)^0 = 1$ şi suma este egală cu numărul de cuvinte - cod din $A_{n,k}$, care este 2^k.

Să presupunem acum că $\mathbf{x} \notin A_{n,k}^{\perp}$, deci există un cuvânt - cod $\mathbf{v_0} \in A$ cu $\mathbf{v_0 x} = 1$. Vom arăta că în acest caz, numărul cuvintelor - cod ortogonale pe $\mathbf{x}$ este egal cu cel al cuvintelor - cod ne-ortogonale pe $\mathbf{x}$ (şi deci suma din formulă este 0).

Fie $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ toate cuvintele - cod ortogonale pe $\mathbf{x}$. Atunci $\mathbf{v_1} + \mathbf{v_0}, \mathbf{v_2} + \mathbf{v_0}, \ldots, \mathbf{v_r} + \mathbf{v_0}$ sunt toate cuvintele - cod ne-ortogonale pe $\mathbf{x}$. Într-adevăr:

1. $\forall i \ (1 \leq i \leq r) \quad (\mathbf{v_i} + \mathbf{v_0})\mathbf{x} = \mathbf{v_i x} + \mathbf{v_0 x} = 0 + 1 = 1;$

2. Dacă $\mathbf{v} \in A_{n,k}$ verifică relaţia $\mathbf{vx} = 1$, atunci $\mathbf{v} - \mathbf{v_0}$ este un cuvânt - cod ortogonal pe $\mathbf{x}$, deci $\mathbf{v} - \mathbf{v_0} = \mathbf{v_i}$ pentru un anumit i. □

Teorema 2.14 *Pentru orice* (n,k) *- cod liniar binar* $A_{n,k}$, *are loc relaţia (identitatea MacWilliams):*

$$P_{A_{n,k}^{\perp}}(x) = \frac{(1+x)^n}{2^k} P_{A_{n,k}}\left(\frac{1-x}{1+x}\right).$$

Demonstraţie: Să rescriem numărătorul de ponderi sub o formă puţin diferită:

$$B(x,y) = \sum_{i=0}^{n} A_i x^i y^{n-i}.$$

Evident, deoarece $P(x) = B(x,1)$ şi $B(x,y) = y^n P\left(\frac{x}{y}\right)$, cele două expresii sunt echivalente.

În notaţia cu polinomul B, identitatea MacWilliams se scrie

$$B_{A^{\perp}}(x) = \frac{1}{2^k} B_A(y-x, y+x).$$

Cu ajutorul ponderii cuvintelor - cod, numărătorul de ponderi are forma

$$B_A(x,y) = \sum_{i=0}^{n} A_i x^i y^{n-i} = \sum_{\mathbf{a} \in A_{n,k}} x^{w(\mathbf{a})} y^{n-w(\mathbf{a})}.$$

Prelucrând membrul drept al identităţii MacWilliams, avem:

$$B_{A_{n,k}}(y-x,y+x)=\sum_{\mathbf{a}\in A_{n,k}}(y-x)^{w(\mathbf{a})}(y+x)^{n-w(\mathbf{a})}=\sum_{\mathbf{a}\in A_{n,k}}\prod_{i=1}^{n}[y+(-1)^{a_i}x]$$

unde s-a notat $\mathbf{a}=(a_1,a_2,\ldots,a_n)$.

În mod analog, membrul stâng se scrie:

$$B_{A_{n,k}^{\perp}}(x,y)=\sum_{\mathbf{a}\in A_{n,k}^{\perp}}x^{w(\mathbf{a})}y^{n-w(\mathbf{a})}.$$

Folosind Propoziţia 2.6, el se poate reformula:

$$B_{A_{n,k}^{\perp}}(x,y)=\sum_{\mathbf{a}\in Z_2^n}\left[\frac{1}{2^k}\sum_{\mathbf{v}\in A_{n,k}}(-1)^{\mathbf{av}}\right]x^{w(\mathbf{a})}y^{n-w(\mathbf{a})}.$$

Expresia din paranteze este 0 pentru toate cuvintele $\mathbf{a}$ care nu sunt în $A_{n,k}^{\perp}$. Deci

$$B_{A_{n,k}^{\perp}}(x,y)=\frac{1}{2^k}\sum_{\mathbf{v}\in A_{n,k}}\sum_{\mathbf{a}\in Z_2^k}(-1)^{\mathbf{va}}x^{w(\mathbf{a})}y^{n-w(\mathbf{a})}.$$

Suma interioară se face după toate secvenţele binare $\mathbf{a}$ de lungime n. Vom ordona această sumă după ponderile lui $\mathbf{a}$: pentru $\mathbf{a}=0$ sumandul este y^n; pentru cuvintele $\mathbf{a}$ de pondere 1 avem: $[(-1)^{a_1}+(-1)^{a_2}+\ldots+(-1)^{a_n}]xy^{n-1}$ etc; în final, pentru ponderea n avem $[(-1)^{a_1}+\ldots+(-1)^{a_n}]x^n$.

Se observă uşor că suma tuturor acestor sumanzi $\sum_{k=0}^{n}[(-1)^{a_{i_1}}+\ldots+(-1)^{a_{i_k}}]x^ky^{n-k}$ este egală cu

$[y+(-1)^{a_1}x][y+(-1)^{a_2}x]\ldots[y+(-1)^{a_k}x]=\prod_{i=1}^{n}[y+(-1)^{a_i}x]$. Deci

$$B_{A_{n,k}^{\perp}}(x,y)=\frac{1}{2^k}\sum_{\mathbf{a}\in A_{n,k}}\prod_{i=1}^{n}[y+(-1)^{a_i}x]=\frac{1}{2^k}B_{A_{n,k}}(y-x,y+x).\qquad\square$$

Exemplul 2.18 *Să considerăm codul $A_{4,2}$ din Exemplul 2.12, al cărui numărător de ponderi a fost dat în Exemplul 2.17.*

Pentru codul dual, numărătorul de ponderi este

$$P_{A_{4,2}^{\perp}}(x)=\frac{(1+x)^4}{2^2}P_{A_{4,2}}\left(\frac{1-x}{1+x}\right)=\frac{(1+x)^4}{4}\left[1+\left(\frac{1-x}{1+x}\right)^2+\right.$$

$$\left.+2\left(\frac{1-x}{1+x}\right)^3\right]=\frac{(1+x)^4+(1+x)^2(1-x)^2+2(1+x)(1-x)^3}{4}=$$

$$\frac{4+4x^2+8x^3}{4} = 1+x^2+2x^3.$$

Deci cele două coduri au acelaşi numărător de ponderi. Aceasta nu înseamnă însă că cele două coduri coincid (şi deci codul ar fi auto - dual); un calcul simplu conduce la determinarea celor două coduri;

$A_{4,2} = \{0000, 0111, 1010, 1101\}$, $A_{4,2}^{\perp} = \{0000, 0101, 1011, 1110\}$

De reţinut că a avea acelaşi numărător de ponderi este doar o condiţie necesară, nu şi suficientă pentru ca un cod să coincidă cu dualul său.

Exemplul 2.19 *Fie codul cu repetiţie de lungime pară n:*

$$C = \{00\ldots0, 11\ldots1\};$$

numărătorul lui de ponderi este $P_C(x) = 1+x^n$.

Codul dual are numărătorul de ponderi

$$P_{C^{\perp}}(x) = \frac{(1+x)^n}{2}\left[1+\left(\frac{1-x}{1+x}\right)^n\right] = 1+C_n^2x^2+C_n^4x^4+\ldots+x^n.$$

Rezultă din această formă că dualul codului binar cu repetiţie este codul liniar binar format din toate cuvintele de pondere pară (şi lungime n).

2.11 Exerciţii

2.1 *Să se construiască coduri liniare care să transforme cuvântul* $(a_1, a_2, \ldots, a_n) \in Z_q^n$ *în:*

1. $(a_1, a_1, a_1, a_2, a_2, a_2, \ldots, a_n, a_n, a_n)$;
2. $(a_1, b_1, a_2, b_2, \ldots, a_n, b_n)$ *unde* $b_1 = a_1, b_2 = a_1 + a_2, \ldots, b_n = a_1 + \ldots + a_n$;
3. $(a_1, 2a_n, 3a_2, 4a_{n-1}, \ldots)$.

2.2 *Un cod liniar peste* Z_5 *are următoarea matrice generatoare:*

$$G = \begin{pmatrix} 1 & 2 & 3 & 1 & 2 \\ 2 & 2 & 4 & 1 & 0 \\ 1 & 1 & 2 & 2 & 1 \end{pmatrix}$$

Să se afle matricea de control.

2.3 *Aceeaşi problemă pentru* Z_7.

2.4 *Determinaţi dacă următorul cod liniar binar este sistematic sau nu:*

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

Dacă nu, găsiţi un cod sistematic echivalent.

2.5 *Găsiţi matricea generatoare a unui cod liniar binar care are matricea de control:*

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

2.6 *Să se demonstreze echivalenţa* (1).

2.7 *Se dă matricea de control*

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

a unui cod liniar binar.

Să se decodifice mesajele 110110, 010100.

2.8 *Descrieţi dualul* $(6,3)$ *- codului binar definit de ecuaţiile:*

$$x_1 = x_4, \quad x_2 = x_5, \quad x_3 = x_6.$$

2.9 *Fie A un cod liniar binar obţinut din toate sumele posibile ale cuvintelor* 101011, 011101, 011010.

1. *Aflaţi o matrice de control a codului.*

2. *Aflaţi un tablou standard şi decodificaţi* 111011.

2.10 *Demonstraţi că un cod liniar binar descris de ecuaţiile*

$$x_3 = x_1 + x_2, \quad x_4 = x_1, \quad x_5 = x_1 + x_2$$

corectează o eroare. Construiţi tabloul standard.

2.11 *Construiţi o tabelă de sindromuri de decodificare pentru:*

1. *Codul cu repetiţie de lungime 7;*

2. *Codul din Exerciţiul 2.8;*

3. *Codul peste Z_3 cu matricea generatoare:*

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

2.12 *Fie un cod liniar $A_{n,k} \subseteq Z_q^n$ şi tabela de sindromuri de decodificare cores-punzătoare. Definim o operaţie de decodificare $\theta : Z_q^n \to A_{n,k}$ cu proprietatea: $\forall \mathbf{v} \in A_{n,k}, \forall \mathbf{e}$ reprezentant din tabelă, $\theta(\mathbf{v}+\mathbf{e}) = \mathbf{v}$. ($\theta$ corectează erorile - tip din tabela de decodificare). Să se arate că atunci θ nu corectează nici o altă eroare - tip: dacă $\mathbf{e}$ nu este un reprezentant din tabelă, atunci există $\mathbf{v} \in A_{n,k}$ cu $\theta(\mathbf{v}+\mathbf{e}) \neq \mathbf{v}$.*

(Altfel spus, decodificarea cu sindromuri este optimală).

2.13 *În Z_2^n notăm cu $\overline{\mathbf{x}}$ cuvântul obţinut din $\mathbf{x}$ prin permutarea caracterelor 0 şi 1 între ele (complementare). Să se arate că pentru orice $\mathbf{a}, \mathbf{b} \in Z_2^n$:*

1. $\overline{\mathbf{a}} + \overline{\mathbf{b}} = \mathbf{a} + \mathbf{b}$;

2. $\mathbf{a} + \overline{\mathbf{b}} = \overline{\mathbf{a}} + \mathbf{b} = \overline{\mathbf{a} + \mathbf{b}}$;

3. $d(\mathbf{a}, \overline{\mathbf{b}}) = d(\overline{\mathbf{a}}, \mathbf{b}) = w(\overline{\mathbf{a} + \mathbf{b}})$.

2.14 *Descrieţi codurile modificate obţinute din codul liniar binar cu matricea generatoare*

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

2.15 *Aceeaşi problemă pentru codul peste Z_3 definit prin*

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

2.16 *Fie $A_{n,k}$ un (n,k) - cod liniar binar şi $A'_{n,k-1}$ $(n, k-1)$ - codul obţinut din $A_{n,k}$ prin expurgare. Ce relaţie există între matricile de control ale celor două coduri ?*

2.17 *Arătaţi cum poate codul binar cu repetiţie de lungime 7 să corecteze două erori şi să detecteze 4 erori simultan. Câte erori poate detecta acest cod dacă se impune restricţia să corecteze o singură eroare ?*

2.18 *Fie un* $(15, 4)$ *- cod liniar binar în care fiecare coloană* i *din matricea generatoare este scrierea binară a lui* i *sub forma unui vector cu* 4 *componente. Să se determine distanţa minimă, numărătorul de ponderi şi numărătorul de ponderi al codului dual.*

2.19 *Fie* C *un* $(2k + 1, k)$ *- cod binar astfel ca* $C^{\perp} \subset C$. *Descrieţi mulţimea* $C^{\perp} \setminus C$. *Este ea tot un cod liniar ?*

Capitolul 3

Clase de coduri liniare

3.1 Coduri Hamming

Fie H matricea de control a unui cod liniar binar. Dacă se transmite un cuvânt - cod $\mathbf{a}$ şi se recepţionează $\mathbf{a}+\mathbf{e}$ (deci cu eroare - tip $\mathbf{e}$), atunci sindromul este $\mathbf{e}H^T$. Acest sindrom este egal cu suma coloanelor lui H care corespund poziţiilor afectate de erori (Teorema 2.3).

În particular, o eroare care apare pe o poziţie corespunzătoare unei coloane nule din H nu influenţează sindromul. Deci, o astfel de eroare nu este detectată.

Dacă H are două coloane identice şi se întâmplă ca pe poziţiile corespunzătoare lor să apară simultan erori, acestea se anulează reciproc în calculul sindromului - şi deci nu pot fi detectate.

Pe de-altă parte, dacă toate coloanele lui H sunt distincte şi nenule, o eroare singulară pe poziţia s va face ca sindromul să fie egal cu coloana numărul s din H. În acest caz, erorile singulare pot fi detectate şi corectate foarte uşor.

Pe baza acestor observaţii am demonstrat teorema:

Teorema 3.1 *Un cod liniar binar poate corecta o eroare dacă şi numai dacă matricea sa de control are toate coloanele nenule şi distincte.*

Pentru a se putea corecta toate erorile simple, trebuie să existe sindromuri distincte pentru fiecare eroare - tip; deci, conform marginii Hamming (Teorema 2.11),

$$2^{n-k} \geq n+1.$$

Pe baza acestor considerente se defineşte codul Hamming binar:

Definiţia 3.1 *Codul liniar binar în care coloanele matricii H sunt reprezentarea binară a numerelor $1, 2, \ldots, 2^r - 1$, este numit cod Hamming binar.*

Deci, pentru orice număr natural r ($r \geq 2$) se poate construi un (n, k) - cod liniar binar în care $n = 2^r - 1$, $k = 2^r - r - 1$.

De remarcat că Definiţia 3.1 nu determină pentru fiecare r, în mod unic matricea de control a codului Hamming. De obicei se consideră acea matrice H în care coloana i reprezintă scrierea în binar a numărului i. Deoarece codul este echivalent cu un cod sistematic (există coloane pentru $2^0, 2^1, \ldots, 2^{r-1}$), toate celelalte reprezentări au aceleaşi proprietăţi.

Exemplul 3.1 *Pentru $r = 3$ avem codul Hamming binar de lungime $n = 2^3 - 1 = 7$ cu $k = 2^3 - 3 - 1 = 4$ simboluri de informaţie şi 3 simboluri de control. Matricea de control este:*

$$H_{3,7} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

De aici rezultă că el este determinat de soluţiile sistemului liniar:

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

Să determinăm matricea generatoare a acestui cod. Pentru aceasta, construim întâi codul sistematic echivalent, permutând coloanele pentru a aduce matricea de control la forma eşalonată canonic:

$$H^*_{3,7} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

De aici se obţine matricea generatoare eşalonată canonic:

$$G^*_{4,7} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Aplicând permutarea inversă asupra coloanelor, se obţine matricea generatoare a $(7,4)$ *- codului Hamming binar:*

$$G_{4,7} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Coloanele corespunzătoare matricii unitate $((3,5,6,7))$ *reprezintă poziţiile simbolurilor de informaţie în fiecare cuvânt - cod. Deci simbolurile* x_1, x_2 *şi* x_4 *sunt simboluri de control. Sistemul de sus poate fi rearanjat pentru a permite calculul simbolurilor de control plecând de la simbolurile de informaţie:*

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

Toate cuvintele codului sunt:

Informaţie	*Cuvânt cod*	*Informaţie*	*Cuvânt cod*
0000	0000000	0110	0110011
1000	1000011	0101	0101010
0100	0100101	0011	0011001
0010	0010110	1110	1110000
0001	0001111	1101	1101001
1100	1100110	1011	1011010
1010	1010101	0111	0111100
1001	1001100	1111	1111111

Teorema 3.2 *Orice cod Hamming binar are distanţa minimă* 3.

Demonstraţie: Evident, orice două coloane din matricea de control sunt liniar independente. În plus, se pot găsi trei coloane (de exemplu primele trei) a căror sumă să fie **0**. Conform Teoremei 2.4, distanţa minimă a codului este 3. □

Deci orice cod Hamming binar poate corecta o eroare sau poate detecta două erori. El nu poate realiza acest lucru simultan (nu verifică condiţia $d \geq s + t + 1$ din Teorema 2.13).

Decodificarea se realizează foarte simplu, conform următorului algoritm:

Algoritm A:
Fie **a** vectorul recepţionat.

1. Se calculează sindromul $\mathbf{s} = \mathbf{a}H^T$.
2. Dacă $\mathbf{s} = \mathbf{0}$, nu a apărut nici o eroare (sau eroarea este nedetectabilă), deci $\mathbf{v} = \mathbf{a}$, STOP.
3. Altfel, eroarea este pe poziţia i, unde i este numărul a cărui reprezentare în binar este sindromul **s**.

   Decodificarea este $\mathbf{v} = \mathbf{a} + \mathbf{e_i}$, unde $\mathbf{e_i}$ este cuvântul care are 1 pe poziţia i şi 0 în rest.

Exemplul 3.2 *Să considerăm din nou* $(7,4)$ *- codul Hamming binar din Exemplul 3.1 şi să presupunem că s-a recepţionat cuvântul* $\mathbf{x} = 0011101$. *Calculul sindromului conduce la valoarea*

$$\mathbf{x}H^T = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

care este scrierea în binar a numărului 5. *Deci a intervenit o eroare simplă pe poziţia a cincea. Corectăm această poziţie - schimbând* 1 *cu* 0 *şi se obţine cuvântul - cod* 0011001, *care pe poziţiile* $3,5,6,7$ *conţine mesajul de informaţie:* 1001.

Codul Hamming binar poate fi îmbunătăţit prin extensie. Această operaţie conduce la un $(2^r, 2^r - r - 1)$ - cod liniar binar, cu toate cuvintele - cod de pondere pară.

Exemplul 3.3 *Prin extensia* $(7,4)$ *- codului Hamming binar se obţine codul cu matricea de control*

$$H^* = \left(\begin{array}{ccccccc|c} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ \hline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right).$$

De remarcat că H^* *are rangul* 4: *în plus, toate liniile acestei matrici sunt cuvinte - cod în codul Hamming binar extins. Deci* H^* *poate fi considerată matrice generatoare a acestui cod. Rezultă că* $(8,4)$ *- codul Hamming binar extins este auto - dual.*

Un cod Hamming binar extins este soluţia unui sistem liniar de $n - k + 1$ ecuaţii, ecuaţia suplimentară $\sum_{i=1}^{n+1} x_i = 0$ fiind *ecuaţia de control a parităţii.*

Propoziţia 3.1 *Un cod Hamming binar extins are* $d = 4$.

Demonstraţie: Fie $\mathbf{a} = a_1 a_2 \dots a_n$ un cuvânt - cod de pondere $d = 3$ din codul Hamming binar. Trecând la codul extins, cuvântul $\mathbf{a}' = a_1 a_2 \dots a_n a_{n+1}$ verifică relaţia suplimentară $\sum_{i=1}^{n+1} a_i = 0$ (reamintim, sumele se fac modulo 2). Cum $\sum_{i=1}^{n} a_i = 1$ ($w(\mathbf{a}) = 3$), rezultă $a_{n+1} = 1$. Noul cuvânt are evident pondere minimă, şi aceasta este $3 + 1 = 4$. □

Ca o remarcă, demonstraţia putea rezulta imediat şi din Observaţia 2 care urmează Definiţiei 2.9

Codurile Hamming binare extinse corectează o eroare simplă şi detectează 2 erori simultan. Algoritmul prezentat este bazat pe verificarea celor $n - k + 1$ ecuaţii de control:

Algoritm B:

1. Dacă nu sunt verificate ecuaţia de control a parităţii şi cel puţin una din primele $n - k$ ecuaţii, înseamnă că a apărut o eroare simplă, care se corectează cu Algoritmul A;
2. Dacă ecuaţia de control a parităţii este verificată dar cel puţin una din primele $n - k$ ecuaţii de control nu se verifică, s-a detectat o eroare dublă;
3. În celelalte situaţii nu au apărut erori (sau eroarea este nedetectabilă).

Faptul că se poate lua totdeauna o decizie se bazează pe următorul rezultat:

Teorema 3.3 *Codul Hamming binar este perfect.*

Demonstraţie: Reamintim (Definiţia 2.8) că un cod $C \subseteq Z_q^n$ de distanţă

minimă d este perfect dacă

$$Z_q^n = \bigcup_{\mathbf{x} \in C} S_t(\mathbf{x}), \quad t = \left[\frac{d-1}{2}\right].$$

Scriind această relaţie în funcţie de numărul de elemente din fiecare sferă şi ţinând cont că toate sferele conţin un număr egal de elemente, avem

$$q^k \left[1 + C_n^1(q-1) + \ldots + C_n^t(q-1)^t\right] = q^n.$$

Pentru cazul codurilor Hamming, $q = 2$, $n = 2^r - 1$, $k = 2^r - r - 1$, $t = 1$, deci totul revine la verificarea egalităţii $2^k(1+n) = 2^n$. □

De remarcat că rata de informaţie a codurilor Hamming

$$R = \frac{k}{n} = 1 - \frac{r}{2^r - 1}$$

creşte rapid spre 1. Evident însă că odată cu această creştere scade protecţia faţă de erori.

3.1.1 Coduri Hamming nebinare

Definiţia 3.2 *Fie q un număr prim, r ($r \geq 2$) un număr întreg şi $n = \frac{q^r - 1}{q - 1}$. Se numeşte cod Hamming nebinar un $(n, n-r)$ - cod liniar peste Z_q, în care matricea de control are orice pereche de două coloane liniar independente (nici o coloană nu este multiplu scalar al altei coloane).*

Mulţimea coloanelor unui astfel de cod formează o mulţime maximală de vectori liniar independenţi doi câte doi.

Exemplul 3.4 *Să considerăm $q = 3, r = 2$. Atunci $n = \frac{3^2 - 1}{3 - 1} = 4$. Un $(4,2)$ - cod Hamming ternar poate fi definit prin matricea de control*

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

Decodificarea se poate face folosind tabela de sindromuri, în care s-au luat ca reprezentanţi toate combinaţiile posibile de o eroare:

Sindrom	*Reprezentant*	*Sindrom*	*Reprezentant*
01	1000	12	0001
02	2000	20	0200
10	0100	21	0002
11	0010	22	0020

Dacă se recepţionează de exemplu 2222, *calculul sindromului dă* $\mathbf{s} = 02$. *Reprezentantul este* 2000. *Se calculează* $2222-2000 = 2222+1000 = 0222$ *deci cuvântul - cod transmis a fost* 0222.

Cum matricea generatoare a acestui cod este $G = \begin{pmatrix} 2 & 2 & 1 & 0 \\ 1 & 2 & 0 & 1 \end{pmatrix}$ *ultimele două caractere formează mesajul de informaţie; deci s-a codificat mesajul* 22.

Despre codurile Hamming nebinare se pot stabili următoarele rezultate:

- Deoarece pentru q şi r fixaţi codurile Hamming corespunzătoare sunt echivalente, se poate alege o anumită matrice de control. Uzual se foloseşte matricea în care se scriu pe coloane toate elementele din Z_q^r, cu condiţia ca primul element nenul (de sus în jos) să fie 1.

- Codurile Hamming nebinare au $d = 3$ (evident, din construcţia matricii de control de mai sus). Deci ele pot corecta o eroare.

- Proprietatea de a fi coduri perfecte se păstrează. Într-adevăr, deoarece un cod Hamming conţine $q^k = q^{n-r}$ cuvinte - cod iar $n = \dfrac{q^r - 1}{q - 1}$, egalitatea stabilită în demonstraţia Teoremei 3.3 se scrie $q^{n-r}[1 + n(q-1)] = q^n$, care se verifică imediat.

Exemplul 3.5 *Să considerăm* $(13, 10)$ *- codul Hamming ternar. Acest cod are o aplicaţie interesantă în problema Pronosportului. După cum se ştie, un buletin Pronosport conţine rezultatele (notate cu* $1, 2, X$*) a* 13 *meciuri. Pentru a avea sigur* 13 *rezultate exacte trebuie completate* 3^{13} *buletine. Câte buletine sunt însă necesare pentru a avea sigur* 12 *rezultate exacte ? La prima vedere s-ar părea că* 3^{12}. *Completând însă buletinele cu elementele codului Hamming ternar* $(13, 10)$ *(cu* 0 *în loc de* X*) - care sunt în număr de* 3^{10}, *se atinge scopul dorit. Într-adevăr, acesta fiind un cod perfect corector de o eroare, orice element din* Z_3^{13} *diferă prin cel mult o poziţie de un cuvânt - cod.*

Astfel, numărul buletinelor se reduce de nouă ori.

3.2 Codul Golay

Al doilea cod liniar prezentat aici are capacitatea de corecţie pentru maximum 3 erori.

Vom construi întâi varianta extinsă a codului, deoarece algoritmul de decodificare este mai simplu şi uşor de aplicat ulterior la codul Golay normal.

3.2.1 Codul Golay binar extins

Acest cod a fost folosit de programul spaţial *Voyager* la începutul anilor '80 pentru transmiterea fotografiilor planetelor Jupiter şi Saturn.

Să considerăm matricea 12×12:

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

Figura 3.1.

Fie matricea 12×24, $G = (I_{12}|B)$. Codul liniar binar generat de G se numeşte *cod Golay extins* şi va fi notat C_{24}.

Observaţii:

- Matricea B este mai uşor de construit decât pare la prima vedere. Astfel, eliminând ultima linie şi coloană, matricea rămasă - să spunem B_1 - este generată ciclic (spre stânga) de cuvântul binar 11011100010. Deci $B = \begin{pmatrix} B_1 & \mathbf{1}^T \\ \mathbf{1} & 0 \end{pmatrix}$, unde $\mathbf{1} = 11111111111$. Evident, B este simetrică ($B^T = B$).
- C_{24} are $n = 24$, $k = 12$ şi $2^{12} = 4096$ cuvinte - cod.

- Conform Teoremei 2.2, o matrice de control a codului este $H = (B|I_{12})$.

Teorema 3.4 *$H = (I_{12}|B)$ este de asemenea matrice de control pentru C_{24}.*

Demonstraţie: Liniile din B au pondere impară (7 sau 11); deci produsul (scalar) al unei linii cu ea însăşi este 1. O verificare simplă arată că produsul primei linii cu oricare altă linie din B este 0. Structura ciclică a lui B_1 asigură că atunci produsul scalar al oricăror două linii este 0.

În concluzie, $BB^T = I_{12}$. Dar $B^T = B$, aşa că putem scrie:

$$GH^T = (I|B)\begin{pmatrix} I \\ B \end{pmatrix} = I^2 + B^2 = I + BB^T = I + I = \mathbf{0}.$$

Vom folosi ambele matrici de control pentru decodificarea codului C_{24}.□

Corolarul 3.1

A. *C_{24} admite ca matrice generatoare şi pe $G = (B|I_{12})$.*

B. *Codul Golay extins este auto - dual ($C_{24} = C_{24}^{\perp}$).*

Demonstraţie: Se verifică imediat. □

Teorema 3.5 *C_{24} are distanţa minimă $d = 8$.*

Demonstraţie: Vom demonstra afirmaţia în trei paşi.

1. *Ponderea cuvintelor din C_{24} este multiplu de 4.*

   Să observăm că liniile lui G au pondere 8 sau 12. Fie $\mathbf{v} \in C_{24}$ scris ca sumă de două linii din G : $\mathbf{v} = \mathbf{r_i} + \mathbf{r_j}$. Cum B are liniile ortogonale, rezultă că şi liniile lui G sunt ortogonale. Deci $\mathbf{r_i}$ şi $\mathbf{r_j}$ au un număr par (să zicem $2x$) de elemente 1 în comun. Atunci $w(\mathbf{v}) = w(\mathbf{r_i}) + w(\mathbf{r_j}) - 2(2x)$, care este multiplu de 4.

   Fie acum $\mathbf{v} \in C_{24}$ reprezentat ca sumă de trei linii din G : $\mathbf{v} = \mathbf{r_i} + \mathbf{r_j} + \mathbf{r_s}$. Notăm $\mathbf{v_1} = \mathbf{r_i} + \mathbf{r_j}$. Deoarece $\mathbf{v_1}$, $\mathbf{r_s} \in C_{24}$, iar C_{24} este auto - dual (deci orice două cuvinte - cod sunt ortogonale), rezultă că $\mathbf{v_1}$ şi $\mathbf{r_s}$ au un număr par (să zicem $2y$) de elemente 1 în comun. Deci $w(\mathbf{v}) = w(\mathbf{v_1}) + w(\mathbf{r_s}) - 2(2y)$, care este multiplu de 4.

   Folosind acum un procedeu de inducţie, cum orice cuvânt - cod este combinaţie liniară de linii din G, ponderea sa va fi multiplu de 4.

2. Primele 11 linii din G sunt cuvinte - cod de pondere 8, deci distanţa minimă a codului C_{24} este 4 sau 8.

3. *C_{24} nu are cuvinte de pondere 4.*

   Să presupunem că există $\mathbf{v} \in C_{24}$ cu $w(\mathbf{v}) = 4$. Cum pentru C_{24} am considerat două matrici generatoare, există atunci două mesaje de informaţie $\mathbf{u_1}, \mathbf{u_2} \in Z_2^{12}$ cu $\mathbf{v} = \mathbf{u_1}(I_{12}|B)$, $\mathbf{v} = \mathbf{u_2}(B|I_{12})$. Una din cele două jumătăţi din $\mathbf{v}$ are cel puţin doi de 1; deci $w(\mathbf{u_1}) \geq 2$ sau $w(\mathbf{u_2}) \geq 2$. De aici rezultă că $\mathbf{v}$ se obţine ca suma a cel puţin două linii din B. Conform cu (1), această sumă nu poate avea o pondere mai mică de 4; deci $w(\mathbf{v}) = w(\mathbf{u_i}) + w(\mathbf{u_i}B) \geq 4 + 2 > 4$, contradicţie. □

3.2.2 Decodificarea codului Golay extins

Conform Teoremei 3.5, un cod Golay extins are distanţa minimă $d = 8$, el poate corecta orice combinaţie de maxim 3 erori independente.

În această secţiune vom nota cu $\mathbf{a} \in Z_2^{24}$ cuvântul recepţionat, cu $\mathbf{v}$ cuvântul - cod cel mai apropiat, şi cu $\mathbf{e}$ eroarea - tip ($\mathbf{v} = \mathbf{a} + \mathbf{e}$). Deoarece capacitatea de corecţie este de 3 erori, vom considera $w(\mathbf{e}) \leq 3$.

Pentru orice cuvânt $\mathbf{a} \in Z_2^{24}$, vom separa cu o virgulă prima jumătate a cuvântului de cea de-a doua: $\mathbf{a} = [\mathbf{a_1}, \mathbf{a_2}]$. Eroarea - tip va fi notată $\mathbf{e} = [\mathbf{e_1}, \mathbf{e_2}]$, unde $\mathbf{e_1}, \mathbf{e_2}$ au fiecare lungimea 12. Evident, condiţia $w(\mathbf{e}) \leq 3$ implică $w(\mathbf{e_1}) \leq 1$ sau $w(\mathbf{e_2}) \leq 1$.

Folosind cele două matrici de control, se pot defini două sindromuri pentru $\mathbf{a}$:

$$\mathbf{s_1} = \mathbf{e}\begin{pmatrix} I_{12} \\ B \end{pmatrix} = [\mathbf{e_1}, \mathbf{e_2}]\begin{pmatrix} I_{12} \\ B \end{pmatrix} = \mathbf{e_1} + \mathbf{e_2}B,$$

$$\mathbf{s_2} = \mathbf{e}\begin{pmatrix} B \\ I_{12} \end{pmatrix} = [\mathbf{e_1}, \mathbf{e_2}]\begin{pmatrix} B \\ I_{12} \end{pmatrix} = \mathbf{e_1}B + \mathbf{e_2}.$$

De aici rezultă următoarea observaţie: dacă $w(\mathbf{e_2}) \leq 1$, atunci $\mathbf{s_1}$ este sau un cuvânt de pondere maxim 3 (dacă $w(\mathbf{e_2}) = 0$), sau o linie a lui B cu cel mult doi biţi schimbaţi (dacă $w(\mathbf{e_2}) = 1$).

Similar, dacă $w(\mathbf{e_1}) \leq 1$, atunci $\mathbf{s_2}$ este sau un cuvânt de pondere maxim 3 sau o linie a lui B cu cel mult doi biţi schimbaţi.

Dacă se foloseşte şi faptul că $\mathbf{s_2} = \mathbf{e_1}B + \mathbf{e_2} = (\mathbf{e_1} + \mathbf{e_2}B)B = \mathbf{s_1}B$ (deci se poate folosi doar prima matrice de control), putem defini un algoritm de decodificare a codurilor Golay extinse, astfel:

Algoritm C:

1. Se calculează sindromul $\mathbf{s} = \mathbf{a}H^T$ unde $H = (I_{12}|B)$;
2. Dacă $w(\mathbf{s}) \leq 3$, atunci $\mathbf{e} = [\mathbf{s}, \mathbf{0}]$, STOP.
3. Dacă există o linie $\mathbf{b_i}$ a lui B cu $w(\mathbf{s} + \mathbf{b_i}) \leq 2$, atunci $\mathbf{e} = [\mathbf{s} + \mathbf{b_i}, \mathbf{f_i}]$, STOP.
4. Dacă $w(\mathbf{s}B) \leq 3$, atunci $\mathbf{e} = [\mathbf{0}, \mathbf{s}B]$, STOP.
5. Dacă există o linie $\mathbf{b_i}$ a lui B cu $w(\mathbf{s}B + \mathbf{b_i}) \leq 2$, atunci $\mathbf{e} = [\mathbf{f_i}, \mathbf{s}B + \mathbf{b_i}]$, STOP.
6. Dacă $\mathbf{e}$ nu a fost determinat încă, se cere retransmiterea.

S-a notat cu $\mathbf{f_i}$ un cuvânt de lungime 12 cu 1 pe poziţia i şi 0 în rest.

După determinarea erorii $\mathbf{e}$, cuvântul - cod transmis se determină prin $\mathbf{v} = \mathbf{a} + \mathbf{e}$.

Exemplul 3.6 *Să decodificăm cuvântul*

$$\mathbf{a} = [101111101111, 010010010010].$$

Sindromul este

$$\mathbf{s} = \mathbf{a}H^T = 101111101111 + 001111101110 = 100000000001.$$

Deoarece $w(\mathbf{s}) = 2 \leq 3$, *se găseşte*

$$\mathbf{e} = [\mathbf{s}, \mathbf{0}] = [100000000001, 000000000000]$$

deci s-a transmis cuvântul

$$\mathbf{v} = \mathbf{a} + \mathbf{u} = [001111101110, 010010010010].$$

Deoarece $G = (I_{12}|B)$ este în forma eşalonat canonică, mesajul de informaţie (orice cuvânt din Z_2^{12}) apare pe primele 12 poziţii ale cuvântului - cod. Astfel, în Exemplul 3.6, mesajul de informaţie a fost 001111101110.

Exemplul 3.7 *Se cere decodificarea cuvântului*

$$\mathbf{a} = [001001001101, 101000101000].$$

Sindromul este

$$\mathbf{s} = \mathbf{a}H^T = 001001001101 + 111000000100 = 110001001001.$$

Deoarece $w(\mathbf{s}) = 5$, *se trece la pasul* 3 *al Algoritmului C şi se calculează:*

$$\mathbf{s} + \mathbf{b_1} = 000110001100$$
$$\mathbf{s} + \mathbf{b_2} = 011111000010$$
$$\mathbf{s} + \mathbf{b_3} = 101101011110$$

$$\mathbf{s} + \mathbf{b_4} = 001001100100$$
$$\mathbf{s} + \mathbf{b_5} = 000000010010$$

Deoarece $w(\mathbf{s} + \mathbf{b_5}) \leq 2$, *se determină*

$$\mathbf{e} = [\mathbf{s} + \mathbf{b_5}, \mathbf{f_5}] = [000000010010, 000010000000]$$

şi se decide că s-a transmis cuvântul - cod

$$\mathbf{v} = \mathbf{a} + \mathbf{e} = [001001011111, 101010101000].$$

Exemplul 3.8 *Să decodificăm cuvântul*

$$\mathbf{a} = [000111000111, 011011010000].$$

Sindromul este

$\mathbf{s} = \mathbf{a}H^T = \mathbf{e_1} + \mathbf{e_2}B = 000111000111 + 101010101101 = 101101101010$ *care are ponderea* 7. *Trecând la pasul* 3 *se găseşte* $w(\mathbf{s} + \mathbf{b_i}) \geq 3$ *pentru toate liniile lui* B; *deci se continuă cu pasul* 4: *al doilea sindrom este* $\mathbf{s}B = 111001111101$ *cu ponderea* 8. *Pasul* 5 *va da:*

$$\begin{aligned} \mathbf{s}B + \mathbf{b_1} &= 001110111000 \\ \mathbf{s}B + \mathbf{b_2} &= 010111110110 \\ \mathbf{s}B + \mathbf{b_3} &= 100101101010 \\ \mathbf{s}B + \mathbf{b_4} &= 000001010000 \end{aligned}$$

S-a ajuns la $w(\mathbf{s}B + \mathbf{b_4}) \leq 2$, *deci se poate determina eroarea:*

$$\mathbf{e} = [\mathbf{f_4}, \mathbf{s}B + \mathbf{b_4}] = [000100000000, 000001010000].$$

Cuvântul - cod transmis a fost:

$$\mathbf{v} = \mathbf{a} + \mathbf{e} = [000011000111, 011010000000].$$

3.2.3 Codul Golay binar

Prin relaxarea codului Golay binar extins (eliminarea ultimului bit din fiecare cuvânt - cod) se ajunge la *Codul Golay binar.*

Fie $\hat{B}$ matricea 12×11 obţinută din B prin eliminarea ultimei coloane. Definim $\hat{G} = (I_{12}|\hat{B})$. Codul liniar binar generat de $\hat{G}$ se numeşte *cod Golay binar* şi este notat cu C_{23}. Caracteristicile sale sunt:

$$n = 23, \quad k = 12, \quad \text{număr de cuvinte - cod: } 2^{12} = 4096.$$

Evident, extensia lui C_{23} este C_{24}.

Teorema 3.6 *Distanţa minimă a codului Golay binar este* $d = 7$.

Demonstraţie: Demonstraţia se poate face fie direct (similar celei de la Teorema 3.5) fie folosind faptul că C_{23} este relaxarea codului C_{24}, care are distanţa minimă 8. □

În consecinţă, un cod Golay binar va corecta orice combinaţie de maxim 3 erori.

Teorema 3.7 *Codul Golay binar este perfect.*

Demonstraţie: Se verifică relaţia:

$2^{12}(C_{23}^0 + C_{23}^1 + C_{23}^2 + C_{23}^3) = 2^{12}(1 + 23 + 253 + 1771) = 2^{12}2^{11} = 2^{23}$.

□

Rezultă că orice cuvânt $\mathbf{a} \in Z_2^{23}$ se află la o distanţă cel mult 3 de un cuvânt - cod. Astfel, dacă se adaugă la sfârşit 0 sau 1, formând $\mathbf{a}0$ respectiv $\mathbf{a}1$ pentru a obţine un cuvânt de pondere impară, acest cuvânt este la distanţă maxim 3 de un cuvânt - cod $\mathbf{c} \in C_{24}$. Se foloseşte Algoritmul C pentru a obţine acest cuvânt - cod, apoi se elimină ultimul caracter din $\mathbf{c}$; se ajunge astfel la cel mai apropiat cuvânt - cod din C_{23} faţă de $\mathbf{a}$.

Algoritmul D:

1. Se formează cuvântul extins de pondere impară $\mathbf{a}0$ sau $\mathbf{a}1$;
2. Se decodifică $\mathbf{a}i$ folosind Algoritmul C şi se obţine $\mathbf{c} \in C_{24}$;
3. Se elimină ultimul caracter din $\mathbf{c}$.

Exemplul 3.9 *Să decodificăm* $\mathbf{a} = [001001001001, 11111110000]$.

Deoarece $\mathbf{a}$ *are pondere impară, se construieşte*

$\mathbf{a}0 = 001001001001, 111111100000$.

Sindromul acestui cuvânt este $\mathbf{s_1} = 100010111110$.

Pentru că $\mathbf{s_1} = \mathbf{b_6} + \mathbf{f_9} + \mathbf{f_{12}}$, $\mathbf{a}0$ *se decodifică în*

$[001001000000, 111110100000]$,

aşa că $\mathbf{a}$ *este decodificat în* $[001001000000, 11111010000]$.

3.2.4 Codul Golay ternar

Se poate defini o mică generalizare a codului Golay binar, lucrând peste alfabetul de trei caractere Z_3.

Să considerăm matricea

$$S_5 = \begin{pmatrix} 0 & 1 & 2 & 2 & 1 \\ 1 & 0 & 1 & 2 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 2 & 1 & 0 & 1 \\ 1 & 2 & 2 & 1 & 0 \end{pmatrix}$$

Pe baza ei construim $(11,6)$ - codul ternar C (numit *cod Golay ternar*), cu matricea generatoare

$$G = \begin{pmatrix} & & & & & 1 & 1 & 1 & 1 & 1 \\ & I_6 & & & & & & & & \\ & & & & & & S_5 & & & \end{pmatrix}$$

Codul extins $\overline{C}$ este un $(12,6)$- cod ternar de matrice generatoare

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

Teorema 3.8 *Codul Golay ternar are următoarele proprietăţi:*

1. $\overline{C}$ *este auto-dual;*
2. $\overline{C}$ *are distanţa minimă* 6*;*
3. C *este un cod perfect corector de* 2 *erori.*

Demonstraţie: (1) Fiind un cod sistematic, se poate construi imediat matricea de control a codului $\overline{C}$:

$$\overline{H} = \begin{pmatrix} 2 & 0 & 2 & 1 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 2 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 2 & 0 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 1 & 1 & 2 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 2 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Liniile lui $\overline{H}$ sunt combinaţii liniare de linii ale lui $\overline{G}$; cum ele sunt şi liniar independente, rezultă $C^{\perp} = \overline{C}$.

(2) Fiind auto-dual, rezultă că orice produs scalar între două cuvinte - cod din $\overline{C}$ este 0 (calculele se fac în Z_3); deci toate cuvintele - cod au ponderea divizibilă cu 3. Liniile lui $\overline{G}$ au pondere 6. Mai rămâne de arătat că nu există nici un cuvânt - cod de pondere 3.

Orice combinaţie liniară de două linii din $\overline{G}$ are exact 4 zerouri pe primele şase poziţii şi 2 zerouri pe ultimele şase (ceea ce se verifică imediat) – deci ponderea cuvintelor astfel obţinute este 6. Orice combinaţie

de trei linii are trei elemente nenule pe primele şase poziţii şi cel puţin un element nenul pe ultimele şase, deci ponderea este minim 6. O combinaţie de mai mult de trei linii are pe primele şase poziţii cel puţin patru elemente nenule, deci ponderea este minim 6.

(3) Din rezultatul anterior obţinem că C are $d = 5$, deci poate corecta două erori. Calculând, obţinem

$$C_{11}^0 + C_{11}^1 \cdot 2 + C_{11}^2 \cdot 2^2 = 1 + 22 + 220 = 243$$

deci este verificată relaţia $\sum_{i=0}^{2} C_n^i (q-1)^i = q^{n-k}$, care defineşte codurile perfecte corectoare de două erori. □

Se poate arăta ([10]) că orice cod liniar cu $n = 11$, $d = 5$ şi 3^6 cuvinte - cod este echivalent cu codul Golay ternar.

3.3 Unicitatea codurilor perfecte binare

Cele două clase de coduri liniare prezentate până acum (Hamming şi Golay) sunt perfecte. În această secţiune ne vom limita la codurile liniare binare.

Se observă imediat că pentru corectarea unei erori, singurele coduri liniare binare perfecte sunt codurile Hamming.

Vom mai arăta că această singularitate este valabilă şi în cazul codurilor Golay; anume, singurul cod liniar binar perfect corector de 3 erori independente este codul Golay.

Pentru aceasta sunt necesare două leme:

Lema 3.1 *O condiţie necesară pentru existenţa unui (n, k) - cod liniar binar perfect corector de t erori, este $\sum_{i=0}^{t} C_n^i = 2^p$ pentru un anumit p.*

Demonstraţie: Rezultă imediat din relaţia scrisă în demonstraţia Teoremei 3.3, în care se ia $q = 2$. □

Lema 3.2 $$\sum_{i=0}^{t} C_n^i = \frac{n+1}{t!} R_t(n)$$

unde t este număr natural impar, $R_t(X) \in Z[X]$, $gr(R_t(X)) = t - 1$.

Demonstraţie: Pentru $t = 1$ se verifică imediat.

Presupunem adevărată afirmaţia pentru t (impar) şi o demonstrăm pentru $t+2$. Avem $\sum_{i=0}^{t+2} C_n^i = \frac{n+1}{t!} + C_n^{t+1} + C_n^{t+2} = \frac{n+1}{(t+2)!} \cdot S$, unde

$$S = (t+1)(t+2)R_t(n) + \prod_{i=0}^{t}(n-i).$$

S este un polinom de gradul $t+1$; notându-l cu $R_{t+2}(n)$, afirmaţia este demonstrată. □

Să considerăm acum cazul $t = 3$. Pentru ca să existe un cod binar perfect corector de 3 erori, cu lemele de sus, trebuie ca $(n+1)(n^2-n+6) = 3 \cdot 2^s$, sau

$$(n+1)[(n+1)^2 - 3(n+1) + 8] = 3 \cdot 2^s.$$

Considerată ca o ecuaţie în $n+1$, singurele soluţii întregi pozitive sunt de forma $n+1 = 2^r p$ unde $p = 1$ sau $p = 3$. Înlocuind, se ajunge la

$$2^{2r}p^3 - 2^r \cdot 3p^2 + 8p = 2^{s-r} \cdot 3 \qquad (1)$$

Pentru $r \leq 3$ şi $p = 1, 3$ verificările se fac imediat. Pentru $r \geq 4$ se ajunge la contradicţie. Singurele valori care verifică ecuaţia sunt:

$n = 0, 1, 2$	–	nu corespund nici unui cod.
$n = 3$	–	codul trivial cu un singur cuvânt, de lungime 3.
$n = 7$	–	codul (trivial) cu repetiţie $\{0000000.1111111\}$.
$n = 23$	–	codul Golay.

În acest mod am demonstrat teorema:

Teorema 3.9 *Codul binar Golay este singurul cod binar netrivial perfect, corector de* 3 *erori.*

Lemele 3.1 şi 3.2 pot fi folosite şi pentru alte valori impare ale lui t. Cercetările nu au condus la alte coduri perfecte binare, dar nici nu s-a demonstrat că nu există nici un cod perfect binar corector de t ($t > 3$ impar) erori.

Un alt caz interesant de studiu este $q = 2$, $t = 2$. Aici se poate da teorema:

Teorema 3.10 *Nu există nici un cod netrivial binar perfect corector de* 2 *erori.*

Demonstraţie: Lema 3.1 conduce la relaţia

$$(2n+1)^2 = 2^{s+3} - 7.$$

Ecuaţia $x^2+7=2^m$ a fost studiată în multe articole (vezi Math. Rev. 26, #74). Singurele soluţii sunt $x = 1, 3, 5, 11, 181$ cărora le corespund:

$n = 0, 1$ – fără coduri.

$n = 2$ – codul trivial cu un singur cuvânt.

$n = 5$ – codul cu repetiţie $\{00000, 11111\}$.

$n = 90$.

Acest ultim caz este eliminat de următorul rezultat ([10], pag. 95):

Dacă există un cod binar perfect corector de t erori, atunci $\frac{n+1}{t+1}$ *este număr întreg.* □

Pentru 2 erori, în cazul $q = 3$, singurul cod perfect este codul Golay ternar. Acest fapt rezultă din ecuaţia Teoremei 3.3 în care se ia $q = 3$, $t = 2$: $\sum_{i=0}^{2} C_n^i 2^i = 3^{n-k}$; rezolvarea acestei ecuaţii este dată în [10], pag. 114 – 115.

3.4 Coduri Reed - Muller

Vom introduce o nouă clasă de coduri binare, caracterizate printr-o tehnică de decodificare deosebit de simplă: codurile *Reed - Muller* ($R-M$). Ele au fost definite de Reed, iar Muller a construit modalitatea de decodificare şi - implicit - de detectare şi corectare a erorilor. Unul din aceste coduri ($\mathcal{RM}(1,5)$) a fost folosit în 1969 de sonda Mariner pentru transmiterea de imagini de pe Lună. Fiecare pixel din imagine avea asignat una din $2^6 = 64$ grade de umbră, iar cei şase biţi de informaţie erau codificaţi într-un cuvânt de lungime 32. Codul $\mathcal{RM}(1,5)$ poate corecta până la 7 erori independente.

3.4.1 Definirea prin funcţii booleene

Funcţii şi polinoame booleene

Definiţia 3.3 *O funcţie booleană de* m $(m \geq 1)$ *variabile este o aplicaţie* $f : Z_2^m \to Z_2$.

O modalitate simplă folosită pentru definirea unei funcţii booleene este asocierea unei *tabele de adevăr*: un tablou $(m+1) \times 2^m$ care conţine toate combinaţiile posibile de m valori binare, cărora li se asociază valoarea funcţiei (de asemenea o valoare binară). Prin convenţie, primii m biţi de

pe coloana i ($0 \leq i \leq 2^m - 1$) reprezintă scrierea în baza 2 a numărului i.

Exemplul 3.10 *Următoarea tabelă de adevăr defineşte o funcţie booleană de* 3 *variabile:*

$\mathbf{x_0}$	0	1	0	1	0	1	0	1
$\mathbf{x_1}$	0	0	1	1	0	0	1	1
$\mathbf{x_2}$	0	0	0	0	1	1	1	1
$\mathbf{f}$	0	1	1	0	1	1	1	0

Observăm că o astfel de tabelă defineşte un cuvânt binar de lungime 8. *Afirmaţia este adevărată şi invers: orice cuvânt binar de lungime* 8 *este definit printr-o tabelă de adevăr a unei funcţii booleene de* 3 *variabile. Astfel, se pot identifica funcţiile booleene de* 3 *variabile prin cuvintele binare de lungime* 8. *În tabela de sus, cuvântul* 01101110 *este pus în corespondenţă biunivocă cu funcţia* $\mathbf{f}$.

În cele ce urmează, orice cuvânt binar $\mathbf{f} = f_0 f_1 \ldots f_{2^m-1}$ de lungime 2^m este considerat ca o funcţie booleană de m variabile, unde

$$\begin{aligned} \mathbf{f}(0,0,\ldots,0,0) &= f_0, \\ \mathbf{f}(0,0,\ldots,0,1) &= f_1, \\ \mathbf{f}(0,0,\ldots,1,0) &= f_2, \\ &\vdots \\ \mathbf{f}(1,1,\ldots,1,1) &= f_{2^m-1} \end{aligned}$$

În general, $f_i = \mathbf{f}(i_{m-1}, \ldots, i_1, i_0)$, unde $i = \sum_{k=0}^{m-1} i_k 2^k$

($i_{m-1} \ldots i_1 i_0$ reprezintă scrierea în binar a lui i).

Exemplul 3.11 *Există două funcţii booleene constante:*

$$\mathbf{1} = 11\ldots11, \qquad \mathbf{0} = 00\ldots00.$$

Exemplul 3.12 *Orice variabilă poate fi tratată ca o funcţie booleană. De exemplu,* $\mathbf{x_0}$ *este funcţia booleană care asignează fiecărui* m*-tuplu* $(x_0, x_1, \ldots, x_{m-1})$ *valoarea primei coordonate* x_0. *Deci, valoarea este* 0 *pentru toate numerele pare şi* 1 *pentru toate numerele impare:* $\mathbf{x_0} = 0101\ldots01$ *(vezi Exemplul prima linie din 3.10 pentru cazul* $m = 3$*).*

În general,

$\mathbf{x_i}$ *este cuvântul binar în care pe poziţia* k $(0 \leq k \leq 2^m - 1)$ *este* 1 *atunci şi numai atunci când scrierea binară a lui* k *conţine* 1 *pe poziţia* i.

Această observaţie rezultă din modul de scriere al tabelelor de adevăr. De exemplu, $\mathbf{x_1} = 00110011\ldots0011$ *şi* $\mathbf{x_{m-1}} = \underbrace{00\ldots00}_{2^{m-1}}\underbrace{11\ldots11}_{2^{m-1}}$.

Pentru $m = 4$, *cele patru variabile sunt descrise în Tabelul 3.1:*

Tabelul 3.1:

$\mathbf{x_0}$	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$\mathbf{x_1}$	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
$\mathbf{x_2}$	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
$\mathbf{x_3}$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Pe mulţimea funcţiilor booleene se definesc două operaţii:

- **Suma logică** (*sau exclusiv*): $\mathbf{f + g = h}$,
  unde $h_i = f_i + g_i \ (mod\ 2)$, $0 \leq i \leq 2^m - 1$.
- **Produsul logic** (*şi*): $\mathbf{fg = h}$
  unde $h_i = f_i g_i \ (mod\ 2)$, $0 \leq i \leq 2^m - 1$.

Observaţia 3.1

1. *Produsul logic verifică relaţia* $\mathbf{ff = f}$.

   Deci în reprezentarea funcţiilor nu vor apare exponenţi mai mari de 1.

2. *Există şi alte operaţii care pot fi exprimate cu ajutorul sumei şi produsului logic. Astfel,*

   - **Negaţia** $\mathbf{\bar{f} = 1 + f}$;
   - $\vee$ *(sau disjunctiv):* $\mathbf{f \vee g = f + g + fg}$.

Definiţia 3.4 *Un polinom boolean de* m *nedeterminate este o sumă de termeni din mulţimea*

$$\{\mathbf{0}, \mathbf{1}\} \cup \{\mathbf{x_{i_1} x_{i_2} \ldots x_{i_k}} | 0 \leq i_1 < i_2 < \ldots < i_k \leq m-1,\ k \geq 1\}.$$

Funcţia $\mathbf{0}$ este numită *polinom boolean de gradul* -1, funcţia $\mathbf{1}$ este numită *polinom boolean de gradul* 0, iar orice alt polinom boolean are gradul k, unde k este numărul maxim de factori dintr-un termen al lui $\mathbf{f}$.

Exemplul 3.13 *Polinomul boolean* $\mathbf{1} + \mathbf{x_0x_1}$ *de* 3 *nedeterminate are gradul* 2. *El este negaţia polinomului*

$$\mathbf{x_0x_1} = (01010101)(00110011) = 00010001.$$

Deci $\mathbf{1} + \mathbf{x_0x_1} = 11101110$.

Acelaşi polinom, considerat ca funcţie de 4 *nedeterminate, este cuvântul* 1110111011101110.

Exemplul 3.14 *Polinomul* $\mathbf{x_ix_j}$ $(i \neq j)$ *este cuvântul binar în care pe poziţia* k *este* 1 *dacă şi numai dacă reprezentarea binară a lui* k *are* 1 *pe poziţiile* i *şi* j. *Numărul acestor situaţii este* 2^{m-2} *(deoarece celelalte* $m-2$ *valori din reprezentarea binară a lui* k *pe* m *poziţii pot fi alese arbitrar). Deci* $w(\mathbf{x_ix_j}) = 2^{m-2}$.

Mai general, se poate arăta prin inducţie:

Dacă $i_1, i_2, \ldots, i_r$ sunt valori distincte din $[0, m-1]$, atunci

$$w(\mathbf{x_{i_1}x_{i_2}\ldots x_{i_r}}) = 2^{m-r}. \qquad (*)$$

Fiecare polinom boolean de m variabile determină un cuvânt binar de lungime 2^m: pentru o singură nedeterminată, se foloseşte Exemplul 3.12, după care se operează adunările şi multiplicările necesare.

Invers, orice cuvânt binar $\mathbf{f} = f_0f_1\ldots f_{2^m-1}$ poate fi translatat într-un polinom boolean pe baza următoarei propoziţii:

Propoziţia 3.2 *Dacă* $\mathbf{f}$ *este o funcţie booleană de* m *variabile, atunci:*

$$f(x_0,\ldots,x_{m-2},x_{m-1}) = f(x_0,\ldots,x_{m-2},0)+ \\ +[f(x_0,\ldots,x_{m-2},0) + f(x_0,\ldots,x_{m-2},1)]\mathbf{x_{m-1}}.$$

Demonstraţie: Deoarece x_{m-1} poate lua doar două valori $(0,1)$, este suficient să verificăm identitatea pentru acestea. Cazul $x_{m-1} = 0$ se verifică banal. Pentru $x_{m-1} = 1$ avem:

$f(x_0,\ldots,x_{m-2},0) + [f(x_0,\ldots,x_{m-2},0) + f(x_0,\ldots,x_{m-2},1)] =$
$f(x_0,\ldots,x_{m-2},1)$. □

Exemplul 3.15 *Să translatăm* $\mathbf{f} = 01101110$ *într-un polinom boolean de* 3 *variabile. Vom aplica pe etape formula din Propoziţia 3.2:*

$\mathbf{f} = 0110 + [0110 + 1110]\mathbf{x_2} = 0110 + 1000\mathbf{x_2} =$

$= (01 + [01 + 10]\mathbf{x_1}) + (10 + [10 + 00]\mathbf{x_1})\mathbf{x_2} = 01 + 11\mathbf{x_1} + 10\mathbf{x_2} + 10\mathbf{x_1}\mathbf{x_2} = (0 + [0 + 1]\mathbf{x_0}) + (1 + [1 + 1]\mathbf{x_0})\mathbf{x_1} + (1 + [1 + 0]\mathbf{x_0})\mathbf{x_2} + (1 + [1 + 0]\mathbf{x_0})\mathbf{x_1}\mathbf{x_2} = \mathbf{x_0} + \mathbf{x_1} + \mathbf{x_2} + \mathbf{x_0}\mathbf{x_2} + \mathbf{x_1}\mathbf{x_2} + \mathbf{x_0}\mathbf{x_1}\mathbf{x_2}$.

Teorema 3.11 *Spaţiul liniar Z_2^n, $(n = 2^m)$ are o bază formată din toate monoamele booleene:*

$$\begin{array}{ll} \mathbf{1} & \\ \mathbf{x_i} & (0 \leq i \leq m-1) \\ \mathbf{x_i}\mathbf{x_j} & (0 \leq i, j \leq m-1,\ i \neq j) \\ \vdots & \\ \mathbf{x_0}\mathbf{x_1}\ldots\mathbf{x_{m-1}} & \end{array}$$

Demonstraţie: Fiecare cuvânt de lungime $n = 2^m$ este o funcţie booleană de m nedeterminate, care poate fi exprimată printr-un polinom boolean. Monoamele booleene pot fi considerate în Z_2^n şi sunt evident liniar independente. În plus, deoarece pentru fiecare $k = 0, 1, \ldots, m$ există C_m^k monoame de gradul k, numărul lor total va fi $\sum_{k=0}^{m} C_m^k = 2^m = n$, adică dimensiunea spaţiului liniar. □

Coduri Reed - Muller

Definiţia 3.5 *Se numeşte cod Reed - Muller de lungime $n = 2^m$ şi grad r $(0 \leq r \leq m)$, codul liniar $\mathcal{RM}(r, m)$ al tuturor cuvintelor binare de lungime n care au - ca polinoame booleene - gradul maxim r.*

Exemplul 3.16 *$\mathcal{RM}(0, m)$ este format din toate polinoamele de grad cel mult 0, adică $\mathbf{0}$ şi $\mathbf{1}$. Deci $\mathcal{RM}(0, m)$ este codul cu repetiţie de lungime 2^m.*

Exemplul 3.17 *$\mathcal{RM}(1, m)$ are baza $\mathbf{1}, \mathbf{x_0}, \ldots, \mathbf{x_{m-1}}$; orice polinom de grad cel mult 1 se poate scrie ca sumă de o parte din aceste $m + 1$ polinoame (liniar independente). Deci, $\mathcal{RM}(1, m)$ este un $(2^m, m + 1)$ - cod liniar.*

De exemplu, $\mathcal{RM}(1, 3)$ are matricea generatoare:

$$G = \begin{pmatrix} \mathbf{1} \\ \mathbf{x_0} \\ \mathbf{x_1} \\ \mathbf{x_2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

care generează codul Hamming binar extins $(8,4)$.

Similar, $\mathcal{RM}(1,4)$ *este un* $(16,5)$ *- cod liniar binar, iar* $\mathcal{RM}(1,5)$ *este un* $(32,6)$ *- cod liniar binar, folosit în* 1969 *de programul Mariner, după cum s-a menţionat anterior.*

Propoziţia 3.3 $\mathcal{RM}(r,m)$ *are* $k = C_m^0 + C_m^1 + \ldots + C_m^r$ *simboluri de informaţie.*

Demonstraţie: Codul $\mathcal{RM}(r,m)$ are ca bază monoamele booleene

$$\{\mathbf{1}\} \cup \{\mathbf{x_{i_1}x_{i_2}}\ldots\mathbf{x_{i_s}} | s \leq r,\ 0 \leq i_1 < i_2 < \ldots < i_s < m\}.$$

Numărul acestor monoame este $C_m^0 + C_m^1 + \ldots + C_m^r$. Cum ele sunt liniar independente, vor forma liniile matricii generatoare G a codului; ori, numărul de linii este egal cu numărul de simboluri de informaţie. □

Propoziţia 3.4 *Dualul codului* $\mathcal{RM}(r,m)$ *este codul* $\mathcal{RM}(m-r-1,m)$.

Demonstraţie: Trebuie arătat că cele două coduri au baze ortogonale şi suma dimensiunilor lor este egală cu suma întregului spaţiu.

A: Fie $\mathbf{v_{i_1}v_{i_2}}\ldots\mathbf{v_{i_p}}$ $(p \leq r)$ un monom din baza codului $\mathcal{RM}(r,m)$ şi $\mathbf{v_{j_1}v_{j_2}}\ldots\mathbf{v_{j_s}}$ $(s \leq m-r-1)$ un monom din baza codului $\mathcal{RM}(m-r-1,m)$. Produsul lor are t $(t \leq r+m-r-1 = m-1)$ variabile distincte (variabilele comune apar o singură dată), deci ponderea lui este (conform $(*)$) $2^{m-t} \geq 2$. Fiind un număr par, rezultă că produsul scalar (adică suma în binar a termenilor) este 0.

B: Suma dimensiunilor celor două coduri este:

$$\sum_{i=0}^{r} C_m^i + \sum_{i=0}^{m-r-1} C_m^i = \sum_{i=0}^{r} C_m^i + \sum_{i=r+1}^{m} C_m^i = \sum_{i=0}^{m} C_m^i = 2^m = n. \qquad \square$$

Exemplul 3.18 $\mathcal{RM}(m-2,m)$ *este codul Hamming binar extins de lun-gime* 2^m. *Într-adevăr, codul dual este* $\mathcal{RM}(m-(m-2)-1,m) = \mathcal{RM}(1,m)$, *deci* $\mathcal{RM}(m-2,m)$ *are matricea de control*

$$H = \begin{pmatrix} \mathbf{1} \\ \mathbf{x_0} \\ \vdots \\ \mathbf{x_m} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & \ldots & 0 & 1 & 0 & 1 \\ & & & & & & & & \\ 0 & 0 & 0 & 0 & & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Dacă adunăm prima linie la toate celelalte şi apoi o permutăm cu ultima linie, se obţine altă matrice de control a codului:

$$H \sim \left(\begin{array}{cccc|ccc|c} 1 & 0 & 1 & 0 & & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & & 1 & 1 & 0 & 0 \\ & & & & & & & & \\ 1 & 1 & 1 & 1 & \ldots & 0 & 0 & 0 & 0 \\ \hline 1 & 1 & 1 & 1 & \ldots & 1 & 1 & 1 & 1 \end{array}\right).$$

După eliminarea ultimei linii şi coloane se obţine un $(2^m - 1, m)$ *- cod ale cărui coloane sunt nenule şi diferite două câte două, adică matricea de control* H_0 *a unui cod Hamming binar. Deci* H *este matricea de control a unui cod Hamming binar extins.*

Codificarea mesajelor cu un cod $R - M$ se realizează normal, înmulţind mesajul de informaţie cu matricea generatoare. În acest fel biţii de informaţie devin coeficienţii polinomului boolean corespunzător. Astfel, în $\mathcal{RM}(1, m)$ codificarea celor $m + 1$ caractere de informaţie este:

$$(a_1, a_2, \ldots, a_{m+1}) \begin{pmatrix} \mathbf{1} \\ \mathbf{x_0} \\ \vdots \\ \mathbf{x_{m-1}} \end{pmatrix} = a_1\mathbf{1} + a_2\mathbf{x_0} + \ldots + a_{m+1}\mathbf{x_{m-1}}.$$

3.4.2 Definirea recursivă a codurilor R - M

Să introducem o altă modalitate de definire a codurilor Reed - Muller, nu prin polinoame booleene, ci prin construcţie recursivă.

Definiţia 3.6 *Fie* $m \geq 0$ *un număr natural. Se defineşte codul Reed - Muller* $\mathcal{RM}(r, m)$ *de ordin* r $(0 \leq r \leq m)$ *şi lungime* $n = 2^m$ *astfel:*

- $\mathcal{RM}(0, m) = \{00\ldots0, 11\ldots1\}, \quad \mathcal{RM}(m, m) = Z_2^n.$
- $\mathcal{RM}(p, m) = \{[\mathbf{a}, \mathbf{a}+\mathbf{b}] | \mathbf{a} \in \mathcal{RM}(p, m-1), \mathbf{b} \in \mathcal{RM}(p-1, m-1)\}, 0 < p < r.$

S-a notat cu $[\mathbf{x}, \mathbf{y}]$ un cuvânt de lungime 2^m scris ca alăturare de două subcuvinte de lungimi egale (2^{m-1}), separate prin virgulă (similar notaţiei folosite la codul Golay binar).

Exemplul 3.19 $\mathcal{RM}(0,0) = \{0,1\}$

$\mathcal{RM}(0,1) = \{00,11\}$, $\mathcal{RM}(1,1) = \{00,01,10,11\} = Z_2^2$,

$\mathcal{RM}(0,2) = \{0000,1111\}$, $\mathcal{RM}(2,2) = Z_2^4$,

$\mathcal{RM}(1,2) = \{[\mathbf{a}, \mathbf{a}+\mathbf{b}] | \mathbf{a} \in \{00,01,10,11\}, \mathbf{b} \in \{00,11\}\} =$

$= \{0000, 0011, 0100, 0111, 1000, 1011, 1100, 1111\}.$

În mod similar se poate da o definiţie recursivă a matricii generatoare $G(r,m)$ pentru codul $\mathcal{RM}(r,m)$.

- $G(0,m) = (11\ldots1)$;
- $0 < p < r < m \Longrightarrow G(p,m) = \begin{pmatrix} G(p,m-1) & G(p,m-1) \\ \mathbf{0} & G(p-1,m-1) \end{pmatrix}$;
- $G(m,m) = \begin{pmatrix} G(m-1,m) \\ 00\ldots01 \end{pmatrix}$.

Teorema 3.12 *$G(r,m)$ este matrice generatoare pentru codul $\mathcal{RM}(r,m)$.*

Demonstraţie: Se verifică prin inducţie după r. □

Exemplul 3.20 *Să considerăm $r = 2$; atunci lungimea este $n = 2^2 = 4$ şi pentru $r = 1,2$ avem*

$$G(1,2) = \begin{pmatrix} G(1,1) & G(1,1) \\ \mathbf{0} & G(0,1) \end{pmatrix}, \qquad G(2,2) = \begin{pmatrix} G(1,2) \\ 0\,0\,0\,1 \end{pmatrix}.$$

Din definiţie, matricile generatoare pentru $\mathcal{RM}(0,1)$ şi $\mathcal{RM}(1,1)$ sunt

$$G(0,1) = (1\ 1), \qquad G(1,1) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \text{ aşa că}$$

$$G(1,2) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \qquad G(2,2) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Propoziţia 3.5 $\mathcal{RM}(r-1,m) \subseteq \mathcal{RM}(r,m)$.

Demonstraţie: Să considerăm iniţial matricea

$$G(1,m) = \begin{pmatrix} G(1,m-1) & G(1,m-1) \\ \mathbf{0} & G(0,m-1) \end{pmatrix}.$$

Pentru că $\mathbf{1}$ este prima linie a lui $G(1,m-1)$, cuvântul $[\mathbf{1},\mathbf{1}]$ formează prima linie a matricii $(G(1,m-1)\ G(1,m-1))$. Deci $\mathcal{RM}(0,m) = \{\mathbf{0},\mathbf{1}\}$ este conţinut în codul $\mathcal{RM}(1,m)$.

În general, deoarece $G(r-1,m-1)$ este submatrice a lui $G(r,m-1)$ şi $G(r-2,m-1)$ este o submatrice a lui $G(r-1,m-1)$, este evident că

$$G(r-1,m) = \begin{pmatrix} G(r-1,m-1) & G(r-1,m-1) \\ \mathbf{0} & G(r-2,m-1) \end{pmatrix}$$

este o submatrice a lui $G(r,m)$, deci $\mathcal{RM}(r-1,m)$ este subcod al lui $\mathcal{RM}(r,m)$. □

Teorema 3.13 *$\mathcal{RM}(r,m)$ are distanţa $d = 2^{m-r}$.*

Demonstraţie: Vom folosi o inducţie după r:

Pentru $r = 0$, evident ($\mathcal{RM}(0,m)$ fiind codul cu repetiţie, distanţa sa este $d = n = 2^m$).

La pasul inductiv, deoarece

$\mathcal{RM}(r,m) = \{[\mathbf{x}, \mathbf{x}+\mathbf{y}] | \mathbf{x} \in \mathcal{RM}(r,m-1), \mathbf{y} \in \mathcal{RM}(r-1,m-1)\}$ şi $\mathcal{RM}(r-1,m-1) \subseteq \mathcal{RM}(r,m-1)$ (Propoziţia 3.5), rezultă $\mathbf{x}+\mathbf{y} \in \mathcal{RM}(r,m-1)$.

Dacă $\mathbf{x} \neq \mathbf{y}$, conform ipotezei de inducţie $w(\mathbf{x}+\mathbf{y}) \geq 2^{m-1-r}$. Cum şi $w(\mathbf{x}) \geq 2^{m-1-r}$, putem scrie $w([\mathbf{x}+\mathbf{y}, \mathbf{x}]) = w(\mathbf{x}+\mathbf{y}) + w(\mathbf{x}) \geq 2^{m-r}$.

Dacă $\mathbf{x} = \mathbf{y}$, atunci $[\mathbf{x}, \mathbf{x}+\mathbf{y}] = [\mathbf{y}, \mathbf{0}]$; dar $\mathbf{y} \in \mathcal{RM}(r-1,m-1)$ şi deci $w([\mathbf{y}, \mathbf{0}]) = w(\mathbf{y}) \geq 2^{m-r}$.

Cum orice linie a matricii generatoare este cuvânt - cod, iar ultima linie are ponderea exact 2^{m-r}, demonstraţia este încheiată. □

3.4.3 Definirea geometrică a codurilor R-M

Codurile Reed - Muller mai pot fi definite şi geometric prin folosirea spaţiilor afine. Avantajul acestei reprezentări constă în modalitatea mai simplă de aplicare a algoritmilor de decodificare.

Pentru uşurinţa descrierii am construit întâi cazul tri-dimensional. De asemenea, pentru a vedea echivalenţa cu definirea anterioară a codurilor Reed - Muller, vom face permanent legătura cu polinoamele booleene (sau cu funcţiile lor caracteristice).

Cazul 3 -dimensional

Spaţiul euclidian 3 - dimensional binar este mulţimea $\{(a,b,c) | a,b,c \in Z_2\}$. Spre deosebire de spaţiul euclidian obişnuit - unde cele trei coordonate luau valori în $\mathcal{R}$ - aici numărul punctelor este finit: numai 8. Ele pot fi listate, renotându-le astfel:

Punct	Funcţie caracteristică
$\mathbf{p_0} = 000$	00000001
$\mathbf{p_1} = 001$	00000010
$\mathbf{p_2} = 010$	00000100
$\mathbf{p_3} = 011$	00001000
$\mathbf{p_4} = 100$	00010000
$\mathbf{p_5} = 101$	00100000
$\mathbf{p_6} = 110$	01000000
$\mathbf{p_7} = 111$	10000000

Dreptele pot fi definite în geometria euclidiană prin expresii de forma

$$\mathbf{a} + t\mathbf{b} \qquad \mathbf{a}, \mathbf{b} \in Z_2^3,\ \mathbf{b} \neq \mathbf{0}$$

unde t este un parametru binar ($t \in \{0, 1\}$). Deci o dreaptă în spaţiul 3 - dimensional binar are numai 2 puncte: $\mathbf{a}$, $\mathbf{a} + \mathbf{b}$. Invers, orice pereche de două puncte distincte $\mathbf{a}, \mathbf{a}'$ formează o dreaptă, anume $\mathbf{a} + t(\mathbf{a}' - \mathbf{a})$. Putem astfel să considerăm dreptele ca fiind totalitatea celor $C_8^2 = 28$ submulţimi de câte două puncte: $\{\mathbf{p_0}, \mathbf{p_1}\}, \{\mathbf{p_0}, \mathbf{p_2}\}, \ldots, \{\mathbf{p_6}, \mathbf{p_7}\}$.

În mod similar, *planele* din geometria euclidiană sunt definite

$$\mathbf{a} + t_1\mathbf{b} + t_2\mathbf{c}, \quad \mathbf{a}, \mathbf{b}, \mathbf{c} \in Z_2^3, \quad \mathbf{b}, \mathbf{c} \text{ liniar independente},$$

unde t_1, t_2 sunt parametri binari. Un plan este format deci din patru puncte: $\mathbf{a}, \mathbf{a} + \mathbf{b}, \mathbf{a} + \mathbf{c}, \mathbf{a} + \mathbf{b} + \mathbf{c}$. Aparent, deşi numărul planelor ar trebui să fie $C_8^4 = 70$, condiţia de liniar independenţă reduce acest număr la 14 (vezi Tabelul 3.2).

În general, un plan este descris de ecuaţia generală

$$h_0x_0 + h_1x_1 + h_2x_2 = c,$$

care defineşte un subspaţiu 2 - dimensional al lui Z_2^3.

Din faptul că orice dreaptă este o intersecţie de două plane, ea poate fi descrisă printr-un sistem de două ecuaţii liniare:

$$h_0x_0 + h_1x_1 + h_2x_2 = c, \qquad h_0'x_0 + h_1'x_1 + h_2'x_2 = c'.$$

Dreptele şi planele sunt exemple de spaţii afine. Un *spaţiu afin* în Z_2^3 este o mulţime de forma

$$C_a = \mathbf{a} + C = \{\mathbf{a} + \mathbf{b} | \mathbf{b} \in C\},$$

unde $\mathbf{a} \in Z_2^3$ iar C este un subspaţiu liniar din Z_2^3. Dacă C este un spaţiu de dimensiune s, atunci C_a este numit s - *spaţiu afin*.

Tabelul 3.2:

Plan	Funcţie caracteristică	Polinom boolean
$\{p_1, p_3, p_5, p_7\}$	10101010	x_0
$\{p_2, p_3, p_6, p_7\}$	11001100	x_1
$\{p_4, p_5, p_6, p_7\}$	11110000	x_2
$\{p_0, p_2, p_4, p_6\}$	01010101	$1 + x_0$
$\{p_0, p_1, p_4, p_5\}$	00110011	$1 + x_1$
$\{p_0, p_1, p_2, p_3\}$	00001111	$1 + x_2$
$\{p_1, p_2, p_5, p_6\}$	01100110	$x_0 + x_1$
$\{p_1, p_3, p_4, p_6\}$	01011010	$x_0 + x_2$
$\{p_2, p_3, p_4, p_5\}$	00111100	$x_1 + x_2$
$\{p_1, p_2, p_4, p_7\}$	10010110	$x_0 + x_1 + x_2$
$\{p_0, p_3, p_4, p_7\}$	10011001	$1 + x_0 + x_1$
$\{p_0, p_2, p_5, p_7\}$	10100101	$1 + x_0 + x_2$
$\{p_0, p_1, p_6, p_7\}$	11000011	$1 + x_1 + x_2$
$\{p_0, p_3, p_5, p_6\}$	01101001	$1 + x_0 + x_1 + x_2$

În particular, dreptele sunt 1 - spaţii afine, iar planele: 2 - spaţii afine. Pentru fiecare punct $\mathbf{p_i}$ avem un 0 - spaţiu afin, şi - în sfârşit - există un 3 - spaţiu afin unic: Z_2^3.

Orice spaţiu afin L poate fi descris de un cuvânt binar $\mathbf{f_L} = f_7 \dots f_1 f_0$ definit prin

$$f_i = \begin{cases} 1 & \text{dacă } \mathbf{p_i} \in L, \\ 0 & \text{altfel} \end{cases} \qquad (0 \le i \le 7)$$

Cuvântul $\mathbf{f_L}$ (sau funcţia booleană de trei variabile corespunzătoare) se numeşte *funcţia caracteristică* a spaţiului afin L (tabelele anterioare listează aceste funcţii pentru 0 şi 2 - spaţii afine).

Fiind date două spaţii afine L, L', intersecţia lor $L \cap L'$ este caracterizată de produsul logic $\mathbf{f_L f_{L'}}$.

Exemplul 3.21 *Primele două plane din Tabelul 3.2 se intersectează pe dreapta* $\{\mathbf{p_3}, \mathbf{p_7}\}$. *Produsul logic al funcţiilor lor caracteristice este*

$$\mathbf{x_0 x_1} = 10001000,$$

care se poate verifica imediat ca fiind funcţia caracteristică a dreptei $\{\mathbf{p_3}, \mathbf{p_7}\}$.

Cazul m - dimensional

Vom extinde construcţiile anterioare la un cadru mai general, cel al geometriei euclidiene m - dimensionale binare. Punctele (elementele lui Z_2^m) pot fi renotate după extensia binară a indicilor; mai clar, vom scrie $Z_2^m = \{\mathbf{p_0}, \mathbf{p_1}, \ldots, \mathbf{p_{2^m-1}}\}$ unde $\mathbf{p_i}$ reprezintă scrierea în binar (pe m biţi)a numărului i $(0 \leq i \leq 2^m - 1)$. Deci

$$\mathbf{p_0} = 000\ldots00,\ \mathbf{p_1} = 000\ldots01,\ \ldots,\ \mathbf{p_{2^m-1}} = 111\ldots11.$$

Definiţia 3.7 *Fie C un subspaţiu liniar r - dimensional al lui Z_2^m şi $\mathbf{a} \in Z_2^m$. Mulţimea*

$$C_{\mathbf{a}} = \mathbf{a} + C = \{\mathbf{a} + \mathbf{b} | \mathbf{b} \in C\}$$

se numeşte r - spaţiu afin modulo C în geometria euclidiană m - dimensională.

Un $(m-1)$ - spaţiu afin se numeşte "hiperplan".

Dacă $\mathbf{b_1}, \ldots, \mathbf{b_r}$ este o bază a lui C, r - spaţiul afin $C_{\mathbf{a}}$ se notează

$$\mathbf{a} + t_1\mathbf{b_1} + \ldots + t_r\mathbf{b_r}.$$

El are 2^r puncte (date de variantele de alegere ale parametrilor binari t_i, $1 \leq i \leq r$).

O altă modalitate de notare a r - spaţiilor afine se realizează cu ajutorul sistemelor de ecuaţii liniare: astfel, dacă spaţiul liniar C este definit ca mulţimea soluţiilor sistemului $H\mathbf{x}^T = \mathbf{0}^T$, atunci $C_{\mathbf{a}}$ este dat de mulţimea soluţiilor sistemului

$$H\mathbf{x}^T = H\mathbf{a}^T.$$

Acest sistem are $m - r$ ecuaţii. În particular, un hiperplan este definit printr-o singură ecuaţie:

$$h_0x_0 + h_1x_1 + \ldots + h_{m-1}x_{m-1} = c.$$

Exemplul 3.22 *Un 0 - spaţiu afin cuprinde un singur punct. Există deci 2^m 0 - spaţii afine distincte: $\{\mathbf{p_0}\}, \{\mathbf{p_1}\}, \ldots, \{\mathbf{p_{2^m-1}}\}$.*

Similar, orice 1 - spaţiu afin (sau "dreaptă") este o mulţime formată din două puncte

$$\mathbf{a} + t\mathbf{b} \equiv \{\mathbf{a}, \mathbf{a} + \mathbf{b}\},$$

şi invers. orice mulţime de două puncte distincte formează un 1 - spaţiu afin. Există deci $C_{2^m}^2$ 1 - spaţii afine.

Exemplul 3.23 *Fie P_i spaţiul afin definit de ecuaţia $x_i = 1$. Deci P_i este mulţimea punctelor $\mathbf{p_k}$ care au 1 pe poziţia i. De exemplu $P_0 = \{\mathbf{p_1}, \mathbf{p_3}, \ldots, \mathbf{p_{2^m-1}}\}$.*

Fiecare P_i este un hiperplan şi - deoarece $\mathbf{p_{2^i}}$ are un singur 1 pe poziţia i şi 0 în rest, putem scrie

$$P_i \equiv \mathbf{p_{2^i}} + C$$

unde C este spaţiul liniar definit de ecuaţia $x_i = 0$, (deci tot un hiperplan).

Propoziţia 3.6 *Există $2(2^m - 1)$ hiperplane.*

Demonstraţie: Deoarece se pot construi $2^m - 1$ ecuaţii cu coeficienţi binari (nu toţi nuli) şi variabile $x_0, x_1, \ldots, x_{m-1}$, Z_2^m are $2^m - 1$ subspaţii C de dimensiune $m - 1$. Fiecare din ele are 2^{m-1} puncte, deci vor exista $\frac{2^m}{2^{m-1}} = 2$ spaţii afine modulo C. □

Exemplul 3.24 *Pentru $i \neq j$, intersecţia $P_i \cap P_j$ (mulţimea punctelor care au 1 pe poziţiile i şi j) este un $(m-2)$ - spaţiu afin. Într-adevăr, dacă se ia $\mathbf{a} = \mathbf{p_{2^i+2^j}}$, avem*

$$P_i \cap P_j \equiv \mathbf{a} + C,$$

unde C este determinat de ecuaţiile $x_i = 0$, $x_j = 0$ (deci C are dimensiunea $m - 2$).

Definiţia 3.8 *Funcţia caracteristică a unui r - spaţiu afin L este cuvântul binar $\mathbf{f_L} = f_{2^m-1} \ldots f_1 f_0$ definit prin*

$$f_i = \begin{cases} 1 & \text{dacă } \mathbf{p_i} \in L, \\ 0 & \text{altfel} \end{cases} \qquad (0 \leq i \leq 2^m - 1)$$

Funcţia caracteristică poate fi interpretată ca un polinom boolean $f_L(x_0, \ldots, x_{m-1})$. Din proprietăţile acestor polinoame rezultă

$$a_0 a_1 \ldots a_{m-1} \in L \iff f_L(a_0, a_1, \ldots a_{m-1}) = 1$$

Observaţia 3.2

1. *Singurul m - spaţiu afin (Z_2^m) are funcţia caracteristică $\mathbf{1} = 11 \ldots 1$.*
2. *Un hiperplan P_i are funcţia $f_{P_i} = x_i$.*

3. *Fie L un hiperplan definit de ecuaţia $h_0x_0 + \ldots h_{m-1}x_{m-1} = c$. Funcţia sa caracteristică va fi un polinom boolean de gradul 1, anume*

$$f_L(x_0, \ldots, x_{m-1}) = h_0x_0 + \ldots + h_{m-1}x_{m-1} + c + 1.$$

Această relaţie rezultă din faptul că un punct $a_0 \ldots a_{m-1}$ este în plan dacă şi numai dacă $h_0a_0 + \ldots + h_{m-1}a_{m-1} = c$. adică

$$f_L(a_0, \ldots, a_{m-1}) = 1.$$

4. *Pentru două spaţii afine L, L', funcţia caracteristică a intersecţiei $L \cap L'$ este $\mathbf{f_L f_{L'}}$.*

Astfel, pentru $P_i \cap P_j$ (care este un $(m-2)$ - spaţiu afin) funcţia caracteristică este $\mathbf{x_i x_j}$.

Mai general, polinomul boolean $\mathbf{x_{i_1} x_{i_2} \ldots x_{i_s}}$ este funcţia caracteristică a unui $(m-s)$ - spaţiu afin.

Teorema 3.14 *Funcţia caracteristică a unui r - spaţiu afin este un polinom boolean de gradul $m - r$.*

Demonstraţie: Un r - spaţiu afin L este definit ca soluţia sistemului de ecuaţii $H\mathbf{x}^T = \mathbf{c}^T$, sau, detaliind,

$$\sum_{j=0}^{m-1} h_{ij}x_j = c_i, \qquad 1 \leq i \leq m - r.$$

Aceste ecuaţii se pot scrie

$$\sum_{j=0}^{m-1} h_{ij}x_j + c_i + 1 = 1, \qquad 1 \leq i \leq m - r.$$

Atunci, polinomul boolean de grad $m - r$

$$f(x_0, \ldots, x_{m-1}) = \prod_{i=1}^{m-r} \left(\sum_{j=0}^{m-1} h_{ij}x_j + c_i + 1 \right)$$

poate fi considerat funcţia caracteristică a lui L. □

Definiţia 3.9 *$\mathcal{RM}(r, m)$ este codul liniar generat de toate funcţiile caracteristice ale spaţiilor afine de dimensiune cel puţin $m - r$ în geometria euclidiană m - dimensională peste Z_2.*

Faptul că aceasta coincide cu definiţia anterioară a codurilor Reed - Muller rezultă din construţia spaţiilor afine: $\mathcal{RM}(r, m)$ conţine toate

funcţiile caracteristice ale s - spaţiilor afine, unde $s \geq m - r$. Faptul că aceste funcţii generează tot spaţiul $\mathcal{RM}(r, m)$ rezultă din Observaţia 3.2 (4).

Exemplul 3.25 *Codul $\mathcal{RM}(1, 3)$ este generat de funcţiile caracteristice ale tuturor planelor. Orice astfel de funcţie este un polinom de trei variabile de gradul 1.*

· *Codul $\mathcal{RM}(2, 3)$ este generat de funcţiile caracteristice ale tuturor planelor şi liniilor. Cum o linie este intersecţia a două plane, funcţia sa caracteristică este produsul a două polinoame de gradul 1, deci un polinom de gradul 2 (cuvânt - cod din $\mathcal{RM}(2, 3)$).*

3.4.4 Decodificarea codurilor Reed - Muller

Avantajul principal al codurilor Reed - Muller constă în facilitatea decodificării, facilitate bazată pe o tehnică diferită de cea de până acum. Această tehnică, numită *decodificare majoritară* nu apelează la ideea de sindrom, ci corectează direct biţii modificaţi, folosind diverse proprietăţi ale cuvântului recepţionat.

3.4.5 Decodificarea majoritară

Să prezentăm pe scurt principiile generale ale decodificării majoritare. Vom începe cu un exemplu foarte simplu:

Exemplul 3.26 *Fie codul binar cu repetiţie de lungime $2n + 1$:*

$$C = \{\underbrace{00\ldots0}_{2n+1}, \underbrace{11\ldots1}_{2n+1}\}.$$

Ca sistem de ecuaţii de control se poate lua

$$\begin{cases} x_1 + x_2 & = & 0 \\ x_1 + x_3 & = & 0 \\ & \vdots & \\ x_1 + x_{2n+1} & = & 0 \end{cases}$$

Dacă se recepţionează cuvântul $\mathbf{y} = \mathbf{x} + \mathbf{e}$, la înlocuirea lui în sistem, acesta devine

$$y_1 + y_i = e_1 + e_i, \quad 2 \leq i \leq 2n + 1.$$

Dacă mai mult de n din cele $2n$ expresii $y_1 + y_i$ iau valoarea 1, aceasta înseamnă că:

- *Au apărut mai puţin de n erori, printre care şi pe prima poziţie* ($e_1 = 1$), *sau*
- *Au apărut mai mult de n erori, dar nu pe prima poziţie* ($e_1 = 0$).

Din aceste două variante, prima este mai probabilă. Deci valoarea majoritară pe care o iau cele n valori $y_1 + y_i$ *va fi valoarea lui* e_1.

Definiţia 3.10 *Fie* $A_{n,k}$ *un cod liniar. Un set de ecuaţii de control pentru* $A_{n,k}$ *este ortogonal pe mulţimea de poziţii* $P \subset \{1, 2, \ldots, n\}$ *dacă şi numai dacă:*

1. $\forall i \in P$, *termenul* x_i *apare (cu coeficient nenul) în fiecare ecuaţie;*
2. $\forall i \notin P$, *termenul* x_i *apare cel mult într-o ecuaţie.*

Decodificarea se realizează caracter cu caracter, după următorul procedeu:

> Fie $\mathbf{a} = a_1 \ldots a_n$ cuvântul recepţionat şi i ($1 \leq i \leq n$) o poziţie.
> Se construieşte mulţimea maximală de ecuaţii de control ortogonale pe poziţia i.
> Fie e_i valoarea obţinută în majoritatea acestor ecuaţii. Al i-lea caracter decodificat este $a_i + e_i$.

Deci, dacă fiecare ecuaţie de control ortogonală pe poziţia i se scrie $a_i = f(a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$, atunci decodificarea majoritară revine la a decodifica pe a_i prin valoarea care apare cel mai frecvent în evaluările tuturor expresiilor f.

Exemplul 3.27 *Fie dualul* (15, 11) *- codului Hamming cu matricea de control*

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

Pentru fiecare pereche de coloane distincte din H există o a treia coloană astfel încât suma celor trei coloane este **0**. *Deci, fiecare cuvânt de pondere 3 al codului Hamming dă o ecuaţie de control cu trei termeni pentru codul dual; în acest fel se obţine pentru fiecare caracter* x_i *câte un sistem format din 7 ecuaţii ortogonale pe poziţia i. Astfel,*

pentru x_1 ele sunt: $x_1 = x_2 + x_3 = x_4 + x_5 = x_6 + x_7 = x_8 + x_9 = x_{10} + x_{11} = x_{12} + x_{13} = x_{14} + x_{15}$;

pentru x_2 : $x_2 = x_1 + x_3 = x_4 + x_6 = x_5 + x_7 = x_8 + x_{10} = x_9 + x_{11} = x_{12} + x_{14} = x_{13} + x_{15}$, *etc.*

Codul are distanţa minimă $d = 8$; *deci el poate corecta maxim* 3 *erori.*

Dacă în mesajul primit $\mathbf{x}$ *apar cel mult* 3 *erori, atunci un sistem de ecuaţii ortogonal pe* x_1 *va avea cel mult trei ecuaţii care să dea pentru* x_1 *o valoare greşită şi cel puţin cinci cu valoarea calculată corect. Valoarea găsită majoritar va fi cea în care se decodifică primul caracter.*

Procedeul se reia pentru $x_2, x_3, \ldots$.

Să presupunem de exemplu că s-a recepţionat mesajul

$$001011110101011.$$

Pentru decodificarea primului caracter vom calcula sumele

$$0+1,\ 0+1,\ 1+1,\ 1+0,\ 1+0,\ 1+0,\ 1+1.$$

Se obţin 5 *valori de* 1 *şi două de* 0; *deci, primul simbol este* 1.

Procedând similar pentru fiecare (folosind ecuaţiile ortogonale corespunzătoare fiecărei poziţii), se ajunge la cuvântul 10101011010101010.

Această metodă este nu numai uşor de implementat, dar are adesea şi alte completări (gen "bitul de testare a parităţii") care fac posibilă corectarea mai multor erori.

În această lucrare vom construi trei algoritmi de decodificare a codurilor $R - M$. Doi din ei sunt bazaţi pe modurile de reprezentare (geometric şi algebric) ale acestor coduri; al treilea este un algoritm care foloseşte definirea recursivă a codurilor $R - M$ şi este prezentat numai pentru cazul $r = 1$. Toţii algoritmii sunt foarte uşor de implementat, utilizarea unuia sau a altuia depinzând doar de criterii particulare în alegerea parametrilor codului.

3.4.6 Algoritm geometric de decodificare majoritară a codurilor $R - M$

Deoarece distanţa minimă a unui cod $\mathcal{RM}(r, m)$ este $d = 2^{m-r}$, el va fi capabil să corecteze până la $2^{m-r-1} - 1$ erori. În cele ce urmează vom presupune că s-a recepţionat cuvântul $\mathbf{y} = y_{2^m-1} \ldots y_1 y_0$, în care au fost modificate maxim $2^{m-r-1} - 1$ poziţii. Problema este de a determina pentru fiecare i $(0 \leq i \leq 2^m - 1)$ dacă bitul y_i trebuie corectat sau nu. Sau - altfel spus - de a vedea dacă poziţia lui $\mathbf{y}$ corespunzătoare 0 - spaţiului afin $\{\mathbf{p_i}\}$ trebuie corectată sau nu.

Pentru a folosi proprietăţile codurilor Reed - Muller exprimate de spaţiile afine, vom reformula totul în maniera următoare:

Pentru fiecare s - spaţiu afin L $(0 \leq s \leq r+1)$ vom cerceta dacă poziţiile cuvântului recepţionat $\mathbf{y}$ corespunzătoare punctelor lui L (adică acei biţi y_i pentru care $\mathbf{p_i} \in L$) sunt modificaţi sau nu.

Vom da întâi câteva noţiuni şi notaţii ajutătoare pentru a simplifica demonstraţiile.

Definiţia 3.11 *Fie G un grup, C un subgrup al său şi $x \in G$. Se numeşte "subgrup modulo C" al lui G mulţimea*

$$C_x = x + C = \{x + a | a \in C\}.$$

Lema 3.3 *Subgrupurile unui grup modulo un subgrup arbitrar C, au următoarele proprietăţi:*

1. $\forall a \in G,\ \exists x \in G, a \in C_x$;
2. *Dacă $x \neq y$ atunci $C_x \cap C_y = \emptyset$ sau $C_x = C_y$;*
3. *Dacă $a, b \in C_x$ atunci $a - b \in C$;*
4. $\forall x,\ card(C_x) = card(C)$.

Demonstraţie: Este lăsată ca exerciţiu.

Teorema principală a acestui paragraf este:

Teorema 3.15 *Orice s - spaţiu afin din geometria binară m - dimensională este inclus în exact $2^{m-s} - 1$ $(s+1)$ - spaţii afine distincte. În plus, orice punct din afara lui L se află în unul şi numai unul din aceste $(s+1)$ - spaţii afine.*

Demonstraţie: **I**: Să arătăm întâi că orice s - subspaţiu liniar $C \subseteq Z_2^m$ este conţinut în exact $2^{m-s} - 1$ subspaţii distincte de dimensiune $s+1$.

Orice $(s+1)$ - spaţiu care conţine C este de forma

$$\overline{C} = C + t\mathbf{b} \equiv \{\mathbf{a} + t\mathbf{b} | \mathbf{a} \in C,\ t = 0, 1\},$$

unde $\mathbf{b} \notin C$ este un punct arbitrar fixat. Aceasta rezultă imediat din faptul că orice bază a lui C poate fi extinsă la o bază a lui $\overline{C}$.

Pentru două puncte $\mathbf{b}, \mathbf{b}' \notin C$, spaţiile liniare $C + t\mathbf{b}$ şi $C + t\mathbf{b}'$ coincid dacă şi numai dacă $\mathbf{b}$ şi $\mathbf{b}'$ sunt în acelaşi subspaţiu modulo C (Lema 3.3).

Din aceeaşi Lemă rezultă că sunt în total $\dfrac{2^m}{2^s}$ subspaţii modulo C. Unul este chiar C, iar celelalte conţin numai puncte dinafara lui C. Deci există (înafară de C) $2^{m-s} - 1$ spaţii distincte $C + t\mathbf{b}$ pentru $\mathbf{b} \notin C$.

II: Orice s - spaţiu afin $L = \mathbf{a} + C \quad (dim(C) = s)$ este conţinut în $2^{m-s} - 1 \quad (s+1)$ - spaţii afine distincte de forma $\mathbf{a} + \overline{C}$.
Cu **I**, orice spaţiu care conţine L este de forma $\mathbf{b} + \overline{C}$. Deoarece L este de forma $\mathbf{a} + C$, se obţine $\mathbf{a} \in \mathbf{b} + \overline{C}$, deci $\mathbf{a} - \mathbf{b} \in \overline{C}$, de unde rezultă (Lema 3.3) că punctele $\mathbf{a}, \mathbf{b}$ sunt în acelaşi subspaţiu modulo C.

III: Orice punct $\mathbf{b} \notin L = \mathbf{a} + C$ este într-un $(s+1)$ - spaţiu afin care conţine L, anume $\mathbf{a} + \overline{C}$ unde $\overline{C} = C + t(\mathbf{b} - \mathbf{a})$.

Într-adevăr, alegând $t = 1$ şi $\mathbf{0} \in C$, avem $\mathbf{b} = \mathbf{a} + [\mathbf{0} + (\mathbf{b} - \mathbf{a})]$.

Pentru a verifica că $\overline{C}$ are dimensiunea $s+1$ este suficient de arătat că $\mathbf{b} - \mathbf{a} \notin C$; dacă prin absurd $\mathbf{b} - \mathbf{a} \in C$, atunci $\mathbf{a} + (\mathbf{b} - \mathbf{a}) = \mathbf{b} \in C$, contradicţie.

În final, ar mai trebui arătat că acest $(s+1)$ - spaţiu afin care îl conţine pe $\mathbf{b}$ este unic. Orice $(s+1)$ - spaţiu afin care conţine $\mathbf{a} + C$ are forma $\mathbf{a} + \overline{C}$ unde $dim(\overline{C}) = s + 1$. Dacă $\mathbf{b} \in \mathbf{a} + \overline{C}$, atunci $\mathbf{b} - \mathbf{a} \in \overline{C}$, deci $\overline{C}$ conţine spaţiul liniar $C + (\mathbf{b} - \mathbf{a})$. Cum ambele spaţii au aceeaşi dimensiune $(s+1)$, ele coincid. □

Corolarul 3.2 *Dacă numărul de erori din cuvântul recepţionat* $\mathbf{y}$ *este* $t < 2^{m-r-1}$, *atunci pentru fiecare* s *- spaţiu afin* L $(0 \leq s \leq r)$ *majoritatea* $(s+1)$ *- spaţiilor afine care conţin* L *au aceeaşi paritate a erorilor ca* L.

Demonstraţie: Conform Teoremei 3.15, un spaţiu afin L este conţinut în $2^{m-r} - 1 > 2t$ $(s+1)$ - spaţii afine L', fiecare L' fiind unic determinat de un punct din afara lui L. Să considerăm toate punctele $\mathbf{p_i} \notin L$ pentru care caracterul y_i este modificat. Lor le corespund cel mult $t \quad (s+1)$ - spaţii afine L'. Toate celelalte spaţii L' rămase au proprietatea că nu conţin nici un punct $\mathbf{p_i}$ din afara lui L pentru care y_i este greşit. Deci aceste spaţii au aceeaşi paritate de erori ca şi L, iar numărul lor este cel puţin $(2^{m-r} - 1) - t > t$, deci majoritar. □

Algoritm de decodificare pentru $\mathcal{RM}(r, m)$:

1. (*Iniţializare*): La recepţionarea unui cuvânt $\mathbf{y} \in Z_2^m$ se consideră toate $(r+1)$ - spaţiile afine L. Un spaţiu L se numeşte *impar* dacă $\mathbf{yf_L} = 1$ (reamintim, $\mathbf{f_L}$ este funcţia caracteristică a spaţiului afin L). În caz contrar L este *par*.

2. (*Inducţie*): Pentru fiecare $s = r, r-1, \ldots, 0$ unde $(s+1)$ - spaţiile afine au fost definite drept pare sau impare, se consideră toate s - spaţiile afine. Un s - spaţiu afin L este *par* dacă majoritatea $(s+1)$ - spaţiilor afine care conţin pe L sunt pare; altfel L este *impar*.

3. (*Final*): Pentru $i = 0, 1, \ldots, 2^m - 1$, se corectează y_i dacă şi numai dacă 0 - spaţiul afin $\{\mathbf{p_i}\}$ este impar.

Exemplul 3.28 *Să considerăm* $\mathcal{RM}(1,3)$ *în care s-a recepţionat cuvântul* $\mathbf{y} = 11101010$.

Tabelul 3.3: Primul pas al decodificării lui 11101010

Plan	*Paritate*	*Plan*	*Paritate*
$\{\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_5, \mathbf{p}_7\}$	*par*	$\{\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_6\}$	*impar*
$\{\mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_6, \mathbf{p}_7\}$	*impar*	$\{\mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_5\}$	*par*
$\{\mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7\}$	*impar*	$\{\mathbf{p}_0, \mathbf{p}_3, \mathbf{p}_4, \mathbf{p}_7\}$	*par*
$\{\mathbf{p}_0, \mathbf{p}_2, \mathbf{p}_4, \mathbf{p}_6\}$	*impar*	$\{\mathbf{p}_0, \mathbf{p}_2, \mathbf{p}_5, \mathbf{p}_7\}$	*par*
$\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_4, \mathbf{p}_5\}$	*par*	$\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_6, \mathbf{p}_7\}$	*impar*
$\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$	*par*	$\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_4, \mathbf{p}_7\}$	*par*
$\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_5, \mathbf{p}_6\}$	*impar*	$\{\mathbf{p}_0, \mathbf{p}_3, \mathbf{p}_5, \mathbf{p}_6\}$	*impar*

Tabelul 3.4: Al doilea pas al decodificării lui 11101010

Dreaptă	*Paritate*	*Dreaptă*	*Paritate*	*Dreaptă*	*Paritate*
$\{\mathbf{p}_0, \mathbf{p}_1\}$	*par*	$\{\mathbf{p}_1, \mathbf{p}_5\}$	*par*	$\{\mathbf{p}_3, \mathbf{p}_5\}$	*par*
$\{\mathbf{p}_0, \mathbf{p}_2\}$	*par*	$\{\mathbf{p}_1, \mathbf{p}_6\}$	*par*	$\{\mathbf{p}_3, \mathbf{p}_6\}$	*impar*
$\{\mathbf{p}_0, \mathbf{p}_3\}$	*par*	$\{\mathbf{p}_1, \mathbf{p}_7\}$	*par*	$\{\mathbf{p}_3, \mathbf{p}_7\}$	*par*
$\{\mathbf{p}_0, \mathbf{p}_4\}$	*par*	$\{\mathbf{p}_2, \mathbf{p}_3\}$	*par*	$\{\mathbf{p}_4, \mathbf{p}_5\}$	*par*
$\{\mathbf{p}_0, \mathbf{p}_5\}$	*par*	$\{\mathbf{p}_2, \mathbf{p}_4\}$	*par*	$\{\mathbf{p}_4, \mathbf{p}_6\}$	*impar*
$\{\mathbf{p}_0, \mathbf{p}_6\}$	*impar*	$\{\mathbf{p}_2, \mathbf{p}_5\}$	*par*	$\{\mathbf{p}_4, \mathbf{p}_7\}$	*par*
$\{\mathbf{p}_0, \mathbf{p}_7\}$	*par*	$\{\mathbf{p}_2, \mathbf{p}_6\}$	*impar*	$\{\mathbf{p}_5, \mathbf{p}_6\}$	*impar*
$\{\mathbf{p}_1, \mathbf{p}_2\}$	*par*	$\{\mathbf{p}_2, \mathbf{p}_7\}$	*par*	$\{\mathbf{p}_5, \mathbf{p}_7\}$	*par*
$\{\mathbf{p}_1, \mathbf{p}_3\}$	*par*	$\{\mathbf{p}_3, \mathbf{p}_4\}$	*par*	$\{\mathbf{p}_6, \mathbf{p}_7\}$	*impar*
$\{\mathbf{p}_1, \mathbf{p}_4\}$	*par*				

La primul pas trebuie să decidem planele (2 - spaţii afine) pare şi cele impare. De exemplu planul $L = \{\mathbf{p}_1, \mathbf{p}_3, \mathbf{p}_5, \mathbf{p}_7\}$ *este par deoarece* $\mathbf{yf}_L = 11101010 \cdot 10101010 = 0$. *Se obţine Tabelul 3.3.*

La pasul următor trebuie să decidem paritatea celor $C_8^2 = 28$ *drepte (1- spaţiu afin). Orice dreaptă este conţinută în* $2^{3-1} - 1 = 3$ *plane distincte. De exemplu, dreapta* $\{\mathbf{p}_0, \mathbf{p}_1\}$ *este conţinută în trei plane:* $\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_4, \mathbf{p}_5\}$ *(par),* $\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$ *(par),* $\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_6, \mathbf{p}_7\}$ *(impar), deci, prin majoritate, dreapta* $\{\mathbf{p}_0, \mathbf{p}_1\}$ *este pară.*

În mod similar se efectuează calculele pentru toate dreptele, obţinându-se Tabelul 3.4.

Se poate trece acum la corectarea erorilor. $\mathcal{RM}(1,3)$ *poate corecta maxim* $2^{m-r-1}-1 = 2^{3-1-1}-1 = 1$ *erori.* $\{\mathbf{p_0}\}$ *este conţinut în şapte drepte: şase pare şi una impară, deci* y_0 *este corect etc. Singurul bit care trebuie corectat este* y_6 *deoarece din cele şapte drepte care conţin* $\{\mathbf{p_6}\}$*, şase sunt impare. Cuvântul- cod trimis a fost deci* 10101010.

Algoritmul de decodificare se bazează pe următoarea observaţie:

Fie codul $\mathcal{RM}(r,m)$ şi $\mathbf{y} \in Z_2^m$ un cuvânt recepţionat.

Un $(r+1)$ - spaţiu afin L este par $\iff$ $\mathbf{yf_L} = 0$.

Într-adevăr, dacă $\mathbf{y}$ este cuvânt - cod, atunci $\mathbf{yf_L} = 0$ (afirmaţie care se poate verifica uşor). Acum, dacă au fost perturbate un număr par de caractere din $\mathbf{y}$, corespunzătoare unor puncte din L, valoarea produsului scalar $\mathbf{yf_L}$ nu se modifică (deoarece toate calculele se fac într-un corp de caracteristică 2). Pentru un număr impar de poziţii perturbate, valoarea produsului scalar este 1.

3.4.7 Algoritm algebric de decodificare majoritară

În afară de algoritmul prezentat anterior, bazat pe reprezentarea geometrică a codurilor $R-M$, se poate da şi o variantă algebrică de decodificare, folosind direct definiţia decodificării majoritare.

În cele ce urmează să considerăm un cod $\mathcal{RM}(r,m)$ fixat. El are $k = 1 + C_m^1 + \ldots + C_m^r$ simboluri de informaţie şi lungimea $n = 2^m$.

În matricea $U_{m,n} = \begin{pmatrix} \mathbf{x_0} \\ \mathbf{x_1} \\ \vdots \\ \mathbf{x_{m-1}} \end{pmatrix}$ care conţine pe coloane toate elementele spaţiului liniar Z_2^m, vom nota cu $\mathbf{u_i}$ $(0 \le i \le m-1)$ coloanele care reprezintă baza (naturală a) lui Z_2^m. Deci, orice coloană $\mathbf{w_j}$ $(0 \le j \le n-1)$ din Z_2^m se poate scrie

$$\mathbf{w_j} = \sum_{i=0}^{m-1} t_{ij}\mathbf{u_i}, \qquad t_{ij} \in \{0,1\}.$$

Orice cuvânt $\mathbf{v} \in Z_2^n$ poate fi considerat *indicatorul* unei mulţimi de cuvinte din Z_2^m: acele coloane din $U_{m,n}$ care corespund poziţiilor din $\mathbf{v}$ unde se află valoarea 1.

Exemplul 3.29 *Pentru orice* $j = 0, \ldots, n-1$, *cuvântul*

$$\mathbf{e_j} = 00 \ldots 010 \ldots 0,$$

cu 1 *pe poziţia* j, *corespunde cuvântului de pe coloana* j *din* $U_{m,n}$;
Cuvântul $\mathbf{1} = 11 \ldots 1$ *corespunde întregului spaţiu* Z_2^m.

Propoziţia 3.7

$$\mathbf{e_j} = \prod_{i=0}^{m-1} [\mathbf{x_i} + (1 + t_{ij})\mathbf{1}], \ \forall j, \ 0 \leq j \leq 2^m - 1.$$

Demonstraţie: Să arătăm întâi că $\mathbf{x_i} + (1 + t_{ij})\mathbf{1}$ reprezintă indicatorul acelor coloane din U care au aceeaşi componentă i ca şi coloana $\mathbf{w_j} = \sum_{i=0}^{m-1} t_{ij}\mathbf{u_i}$. Într-adevăr:

1. Dacă $t_{ij} = 1$, atunci $\mathbf{x_i} + (1 + t_{ij})\mathbf{1} = \mathbf{x_i} + (1 + 1)\mathbf{1} = \mathbf{x_i}$, care este indicatorul mulţimii coloanelor cu 1 (ca şi $\mathbf{w_j}$) pe linia i.

2. Dacă $t_{ij} = 0$, atunci $\mathbf{x_i} + (1 + t_{ij})\mathbf{1} = \mathbf{x_i} + \mathbf{1}$, care are componentele lui $\mathbf{x_i}$ cu 0 şi 1 permutate, fiind deci indicatorul mulţimii coloanelor care au 0 (ca şi $\mathbf{w_j}$) pe linia i.

Cuvântul $\prod_{i=0}^{m-1} [\mathbf{x_i} + (1 + t_{ij})\mathbf{1}]$ reprezintă indicatorul mulţimii acelor coloane din matricea $U_{m,n}$ care au toate componentele egale cu cele ale coloanei $\mathbf{w_j}$. Cum coloanele matricii $U_{m,n}$ sunt distincte, această mulţime este chiar $\{\mathbf{w_j}\}$, al cărei indicator este $\mathbf{e_j}$. □

Fie mesajul de informaţie $a_1 a_2 \ldots a_k \in Z_2^k$. Ele se codifică cu ajutorul $(2^m, k)$ - codului Reed - Muller, în cuvântul - cod

$$\mathbf{f} = (f_0, f_1, \ldots, f_{n-1}) =$$
$$a_1\mathbf{1} + a_2\mathbf{x_0} + \ldots + a_{m+1}\mathbf{x_{m-1}} + a_{m+2}\mathbf{x_0 x_1} + \ldots + a_k\mathbf{x_{m-r} x_{m-r+1} \ldots x_{m-1}}.$$

Putem enunţa acum rezultatul principal pe baza căruia se construieşte algoritmul algebric de decodificare majoritară pentru codurile $R - M$.

Teorema 3.16 *Pentru fiecare componentă a_s din mesajul de informaţie, care se înmulţeşte la codificare - cu un produs de forma $\mathbf{x_{i_1}x_{i_2}\dots x_{i_r}}$, există 2^{m-r} sume disjuncte (cu termeni diferiţi) egale cu a_s, conţinând fiecare 2^r componente ale cuvântului - cod corespunzător $\mathbf{f} = f_0f_1\dots f_{n-1}$.*

Demonstraţie: Ţinând cont de Propoziţia 3.7, şi deoarece $\mathbf{e_0}, \mathbf{e_1}, \dots, \mathbf{e_{n-1}}$ constituie de asemenea o bază pentru Z_2^n, putem scrie:

$$\mathbf{f} = (f_0, f_1, \dots, f_{n-1}) = \sum_{j=0}^{n-1} f_j \mathbf{e_j} = \sum_{j=0}^{n-1} f_j \prod_{i=0}^{m-1} [\mathbf{x_i} + (1 + t_{ij})\mathbf{1}].$$

Prin urmare, ca un produs de forma $\mathbf{x_{i_1}\dots x_{i_t}}$ să fie în această sumă, trebuie ca $t_{ij} = 0$ pentru $i \notin \{i_1, i_2 \dots, i_t\}$. Dacă notăm

$$C(i_1, \dots, i_j) = \{p |\ p = \sum_{i=0}^{m-1} t_{ip}2^i, t_{ip} = 0 \text{ dacă } i \notin \{i_1, \dots, i_j\}\} =$$

$$= \{p |\ p = t_{i_1p}2^{i_1} + t_{i_2p}2^{i_2} + \dots + t_{i_jp}2^{i_j},\ t_{i_qp} \in \{0,1\},\ q = 1, 2, \dots, j\},$$

vom avea

$$\mathbf{f} = (f_0, f_1, \dots, f_{n-1}) = \sum_{t=1}^{r} \sum_{i_1 < i_2 < \dots < i_t} \left(\sum_{j \in C(i_1, \dots, i_t)} f_j \right) \mathbf{x_{i_1}x_{i_2}\dots x_{i_t}}.$$

Pe de-altă parte,

$$\mathbf{f} = (f_0, \dots, f_{n-1}) =$$

$$a_1\mathbf{1} + a_2\mathbf{x_0} + \dots + a_{m+1}\mathbf{x_{m-1}} + a_{m+2}\mathbf{x_0x_1} + \dots + a_k\mathbf{x_{m-r}\dots x_{m-1}}.$$

Prin identificare, rezultă că elementul a_s, coeficient al produsului $\mathbf{x_{i_1}x_{i_2}\dots x_{i_r}}$ este

$$a_s = \sum_{j \in C(i_1, \dots, i_r)} f_j \tag{1}$$

Fie $z \notin \{i_1, \dots, i_r\}$. Deoarece baza codului $\mathcal{RM}(r, m)$ nu conţine monoame de forma $\mathbf{x_{i_1}\dots x_{i_r}x_{i_{r+1}}}$, vom obţine prin codificare

$$\sum_{j \in C(i_1, \dots, i_r, z)} f_j = 0. \tag{2}$$

Însă, $C(i_1, \dots, i_r, z) = C(i_1, \dots, i_r) \cup [C(i_1, \dots, i_r) + 2^z]$

unde s-a notat $C + t = \{x + t |\ x \in C\}$. Într-adevăr, putem scrie

$$C(i_1,\ldots,i_r,z) = \{j|\ j = t_{i_1 j}2^{i_1} + \ldots + t_{i_r j}2^{i_r} + t_{zj}2^z\} =$$
$$= \{j|\ j = t_{i_1 j}2^{i_1} + \ldots + t_{i_r j}2^{i_r}\} \cup \{j|\ j = t_{i_1 j}2^{i_1} + \ldots + t_{i_r j}2^{i_r} + 2^z\} =$$
$$= C(i_1,\ldots,i_r) \cup [C(i_1,\ldots,i_r) + 2^z],$$

cele două mulţimi ale reuniunii fiind disjuncte.

Deci relaţia (2) se scrie $\sum_{j\in C(i_1,\ldots,i_r)} f_j + \sum_{j\in C(i_1,\ldots,i_r)+2^z} f_j = 0$ şi - ţinând cont de (1), avem

$$a_s + \sum_{j\in C(i_1,\ldots,i_r)+2^z} f_j = 0. \tag{3}$$

Din (1) şi (3) rezultă următoarea afirmaţie: dacă $m = r+1$, pentru orice simbol de informaţie a_s, înmulţit la codificare cu un produs de forma $\mathbf{x_{i_1}}\ldots\mathbf{x_{i_r}}$, există $2^{m-r} = 2$ sume disjuncte ((1) şi (3)), fiecare de câte 2^r componente ale lui $\mathbf{f}$.

Aceste două relaţii se numesc *relaţii de determinare* a componentei a_s.

Fie acum $z_1, z_2 \notin \{i_1,\ldots,i_r\}$. Deoarece baza codului nu are produse de forma $\mathbf{x_{i_1}}\ldots\mathbf{x_{i_r}x_{i_{r+1}}x_{i_{r+2}}}$, vom avea

$$\sum_{j\in C(i_1,\ldots,i_r,z_1,z_2)} f_j = 0. \tag{4}$$

Dar $C(i_1,\ldots,i_r,z_1,z_2) = C(i_1,\ldots,i_r) \cup [C(i_1,\ldots,i_r)+2^{z_1}] \cup [C(i_1,\ldots,i_r)+2^{z_2}] \cup \quad \cup [C(i_1,\ldots,i_r)+2^{z_1}+2^{z_2}]$.

Deci (4) se descompune în

$$\sum_{j\in C(i_1,\ldots,i_r)} f_j + \sum_{j\in C(i_1,\ldots,i_r)+2^{z_1}} f_j + \sum_{j\in C(i_1,\ldots,i_r)+2^{z_2}} f_j + \sum_{j\in C(i_1,\ldots,i_r)+2^{z_1}+2^{z_2}} f_j = 0$$

ceea ce - ţinând cont de (1) şi (3) - devine

$$a_s + a_s + a_s + \sum_{j\in C(i_1,\ldots,i_r)+2^{z_1}+2^{z_2}} f_j = 0, \text{ deci}$$

$$a_s = \sum_{j\in C(i_1,\ldots,i_r)+2^{z_1}+2^{z_2}} f_j. \tag{5}$$

Am arătat astfel că, dacă $m = r+2$. atunci există $2^{m-r} = 2^2 = 4$ sume ((1), de două ori (3) pentru z_1 respectiv z_2, şi (5)) care dau pe a_s, fiecare sumă conţinând 2^r componente ale lui $\mathbf{f}$. Deci, pentru a_s există în acest caz patru relaţii de determinare.

Teorema rezultă în continuare printr-un procedeu de inducţie finită.

□

Să vedem cum se foloseşte Teorema 3.16 la decodificarea majoritară a codurilor Reed - Muller:

Din construcţia anterioară, pentru fiecare componentă a_s corespunzătoare unui produs de forma $\mathbf{x_{i_1}} \ldots \mathbf{x_{i_r}}$ există 2^{m-r} sume disjuncte, conţinând fiecare 2^r componente ale lui $\mathbf{f}$ care - în absenţa erorilor - vor fi toate egale cu a_s.

Deoarece sumele sunt disjuncte (deci termenii lor sunt elemente diferite ale cuvântului - cod), o eroare singulară va afecta numai o relaţie de determinare a lui a_s. Prin urmare, dacă în transmiterea cuvântului - cod $\mathbf{f}$ apar cel mult $2^{m-r-1} - 1$ erori independente, putem determina - prin majoritate - pe a_s, identificându-l cu valoarea a *jumătate plus unu* din relaţiile de determinare corespunzătoare.

După ce se determină în acest mod coeficienţii tuturor produselor $\mathbf{x_{i_1}} \ldots \mathbf{x_{i_r}}$, se scade din vectorul recepţionat combinaţia liniară

$$a_p \mathbf{x_0} \ldots \mathbf{x_{r-1}} + \ldots + a_k \mathbf{x_{m-r}} \ldots \mathbf{x_{m-1}}.$$

În acest fel problema se reduce în continuare la decodificarea într-un cod $\mathcal{RM}(r-1, m)$; procedeul este similar, fiind determinate aici componentele de informaţie care se codifică prin înmulţire cu produsele de forma $\mathbf{x_{j_1}} \ldots \mathbf{x_{j_{r-1}}}$ $(j_1 < j_2 < \ldots < j_{r-1})$, ş. a. m. d.

Exemplul 3.30 *Fie codul $\mathcal{RM}(2,4)$. El are lungimea $n = 16$, ordinul $r = 2$, $k = 1 + C_4^1 + C_4^2 = 11$ simboluri de informaţie şi $n - k = 5$ simboluri de control. Deci el este un $(16, 11)$ - cod liniar, capabil să corecteze cel mult o eroare (pentru că $2^{4-2-1} - 1 = 1$).*

Orice mesaj de informaţie $(a_1, a_2, \ldots, a_{11}) \in Z_2^{11}$ se codifică în cuvântul $\mathbf{f} = (f_0, \ldots, f_{15}) = a_1\mathbf{1} + a_2\mathbf{x_0} + a_3\mathbf{x_1} + a_4\mathbf{x_2} + a_5\mathbf{x_3} + a_6\mathbf{x_0x_1} + a_7\mathbf{x_0x_2} + a_8\mathbf{x_0x_3} + a_9\mathbf{x_1x_2} + a_{10}\mathbf{x_1x_3} + a_{11}\mathbf{x_2x_3}$.

Să stabilim de exemplu relaţiile de determinare pentru a_9, care se codifică prin înmulţire cu $\mathbf{x_1x_2}$. Trebuiesc calculate mulţimile:

$$C(1,2) = \{j \mid j = t_{1j}2^1 + t_{2j}2^2,\ t_{ij} \in \{0,1\}\} = \{0,2,4,6\}$$
$$C(1,2) + 2^0 = \{1,3,5,7\}$$
$$C(1,2) + 2^3 = \{8,10,12,14\}$$
$$C(1,2) + 2^0 + 2^3 = \{9,11,13,15\}.$$

Deci relaţiile de determinare corespunzătoare lui a_9 devin

$$a_9 = f_0 + f_2 + f_4 + f_6 \qquad a_9 = f_1 + f_3 + f_5 + f_7$$
$$a_9 = f_8 + f_{10} + f_{12} + f_{14} \qquad a_9 = f_9 + f_{11} + f_{13} + f_{15}$$

Dacă intervine o eroare în timpul transmiterii cuvântului - cod $\mathbf{f}$, *va fi afectată o singură relaţie de determinare, celelalte trei relaţii dând valoarea corectă a lui* a_9.

După ce s-au determinat valorile $a_6, a_7, a_8, a_9, a_{10}, a_{11}$, *se trece la etapa a doua a decodificării, scăzând din cuvântul recepţionat combinaţia*

$$a_6\mathbf{x_0x_1} + a_7\mathbf{x_0x_2} + a_8\mathbf{x_0x_3} + a_9\mathbf{x_1x_2} + a_{10}\mathbf{x_1x_3} + a_{11}\mathbf{x_2x_3}$$

şi aplicând cuvântului obţinut procedeul de corectare a erorilor pentru codul $\mathcal{RM}(1,4)$.

În final, a_1 *se decodifică în codul* $\mathcal{RM}(0,4)$, *ceea ce înseamnă că el este egal cu valoarea majoritară (*0 *sau* 1*) luată de biţii* $f_0, f_1, \ldots, f_{15}$.

3.4.8 Decodificarea codurilor $\mathcal{RM}(1,m)$

Pentru cazul $r = 1$ se poate da şi un alt algoritm de decodificare, extrem de eficient. El foloseşte transformarea Hadamard pentru aflarea celui mai apropiat cuvânt - cod (în distantă Hamming).

Vom da iniţial câteva noţiuni ajutătoare.

Definiţia 3.12 *Fie* $A_{m,n}, B_{p,q}$ *două matrici. Se defineşte produsul Kronecker* $A \times B$ *ca fiind matricea* $C_{mp,nq} = [a_{ij}B]$.

Acest produs este evident asociativ şi necomutativ; el va fi utilizat pe larg şi în cazul altor coduri.

Exemplul 3.31 *Fie* $H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, $I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. *Atunci*

$$I_2 \times H = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}, \qquad H \times I_2 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}.$$

Se consideră şirul de matrici definit

$$H_m^i = I_{2^{m-i}} \times H \times I_{2^{i-1}}, \quad i = 1, 2, \ldots$$

unde H este matricea (Hadamard) definită în Exemplul 3.31.

Exemplul 3.32 *Fie* $m = 2$. *Atunci*

$$H_2^1 = I_2 \times H \times I_1 = I_2 \times H, \qquad H_2^2 = I_1 \times H \times I_2 = H \times I_2.$$

Exemplul 3.33 *Fie* $m = 3$. *Atunci:*

$$H_3^1 = I_4 \times H \times I_1 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix},$$

$$H_3^2 = I_2 \times H \times I_2 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \end{pmatrix},$$

$$H_3^3 = I_1 \times H \times I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{pmatrix}.$$

Modalitatea recursivă de construcţie a codurilor $\mathcal{RM}(r, m)$ sugerează existenţa unui algoritm similar de decodificare. Acest algoritm există numai pentru $\mathcal{RM}(1, m)$ şi îl prezentăm fără a demonstra corectitudinea lui.

Algoritm de decodificare a codurilor $\mathcal{RM}(1,m)$:

Fie $\mathbf{y} \in Z_2^n$ $(n = 2^m)$ cuvântul recepţionat şi $G(1,m)$ matricea generatoare a codului.

1. Se înlocuieşte 0 cu -1 în $\mathbf{y}$; fie $\mathbf{y}'$ noul cuvânt;
2. Se calculează $\mathbf{y_1} = \mathbf{y}'H_m^1$, $\mathbf{y_i} = \mathbf{y_{i-1}}H_m^i$ $(2 \le i \le m)$;
3. Se determină poziţia j a celei mai mari componente (în valoare absolută) a lui $\mathbf{y_m}$.

   Fie $v(j) \in Z_2^n$ reprezentarea binară a lui j pe m biţi (începând cu biţii cei mai semnificativi). Mesajul decodificat este:

   $(1, v(j))$ dacă a j - a componentă a lui $\mathbf{y_m}$ este pozitivă,

   $(0, v(j))$ dacă a j - a componentă a lui $\mathbf{y_m}$ este negativă.

Exemplul 3.34 *Fie* $m = 3$ *şi* $G(1,3)$ *matricea generatoare a codului* $\mathcal{RM}(1,3)$. *Să presupunem că s-a recepţionat cuvântul* $\mathbf{y} = 10101011$.
Primul pas al algoritmului transformă acest cuvânt în $\mathbf{y}' = (1,-1,1,-1,1,-1,1,1)$. *Se calculează apoi:*

$$\mathbf{y_1} = \mathbf{y}'H_3^1 = (0,2,0,2,0,2,2,0)$$
$$\mathbf{y_2} = \mathbf{y_1}H_3^2 = (0,4,0,0,2,2,-2,2)$$
$$\mathbf{y_3} = \mathbf{y_2}H_3^3 = (2,6,-2,2,-2,2,2,-2).$$

Cea mai mare componentă a lui $\mathbf{y_3}$ *este* 6 *pe poziţia* 1. *Cum* $v(1) = 100$ *şi* $6 > 0$, *rezultă că a fost codificat mesajul de informaţie* 1100.
Dacă s-a recepţionat cuvântul $\mathbf{y} = 10001111$, *atunci* $\mathbf{y}' = (1,-1,-1,-1,1,1,1,1)$ *şi*

$$\mathbf{y_1} = \mathbf{y}'H_3^1 = (0,2,-2,0,2,0,2,0),$$
$$\mathbf{y_2} = \mathbf{y_1}H_3^2 = (-2,2,2,2,4,0,0,0).$$
$$\mathbf{y_3} = \mathbf{y_2}H_3^3 = (2,2,2,2,-6.2,2,2).$$

Cea mai mare componentă a lui $\mathbf{y_3}$ *(în valoare absolută) este* -6 *aflată pe poziţia* 4. *Deoarece* $v(4) = 001$ *şi* $-6 < 0$, *mesajul transmis a fost* 0001.

3.5 Coduri MacDonald

Fie $A_{n,k}$ un cod liniar peste Z_2 şi G matricea sa generatoare.

Vom nota cu $M_{k,2^k-1}$ o matrice care conţine toate coloanele nenule posibile de k elemente, care se pot construi peste Z_2. Ordinea acestor coloane este arbitrară dar fixată. Coloana i din matricea M se va numi *coloană de tip* i. Pentru uşurinţa notaţiei, se admite aserţiunea că tipul de coloană i este reprezentarea în binar a numărului zecimal i.

Deoarece codul $A_{n,k}$ nu este influenţat de operaţiile elementare efectuate asupra coloanelor din matricea generatoare (în particular permutarea de coloane), este suficient să definim acest cod indicând doar numărul coloanelor de fiecare tip care intră în componenţa matricii generatoare.

Se obţine astfel *reprezentarea modulară a codului* $A_{n,k}$:

$$N = (n_1, n_2, \ldots, n_{2^k-1})$$

unde n_i este numărul coloanelor de tip i care apar (sau nu) în matricea G,

$$\sum_{i=1}^{2^k-1} n_i = n.$$

Exemplul 3.35 *Să luăm* $k = 2$, $n = 3$ *şi matricea generatoare* $G = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$.

Avem matricea $M = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$ *(coloanele sunt reprezentările binare ale numerelor* $1, 2, 3$*).*

Reprezentarea modulară a codului generat de matricea G *este atunci* $N = (2, 0, 1)$ *(pentru că în* G *există două coloane de tipul* 1*, nici una de tipul* 2 *şi una de tipul* 3*).*

Să introducem matricea

$$K_{2^k-1,n} = M^T G.$$

K are drept linii toate cuvintele nenule ale codului $A_{n,k}$.

Fie – de asemenea – matricea pătrată simetrică:

$$C_{2^k-1,2^k-1} = M^T M.$$

Astfel, pentru Exemplul 3.35,

$$K = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}. \qquad C = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}.$$

Teorema 3.17 *O listă a ponderilor cuvintelor codului $A_{n,k}$ se obţine prin înmulţirea (în $\mathcal{R}$) a matricii de reprezentare modulară cu matricea C:*

$$W_{2^k-1} = NC. \tag{6}$$

Demonstraţie: Să scriem matricea M sub forma unei matrici linie

$$M = (\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{2^k-1}}),$$

unde $\mathbf{v_i}$ $(1 \leq i \leq 2^k - 1)$ este coloana de tip i din matricea M.

Matricea generatoare corespunzătoare reprezentării modulare $N = (n_1, n_2, \ldots, n_{2^k-1})$ este

$$G = (\underbrace{\mathbf{v_1}, \ldots, \mathbf{v_1}}_{n_1}, \underbrace{\mathbf{v_2}, \ldots, \mathbf{v_2}}_{n_2}, \ldots, \underbrace{\mathbf{v_{2^k-1}}, \ldots, \mathbf{v_{2^k-1}}}_{n_{2^k-1}}),$$

unde $n_1 + n_2 + \ldots + n_{2^k-1} = n$. Se obţine deci

$$C = \begin{pmatrix} \mathbf{v_1} \\ \mathbf{v_2} \\ \vdots \\ \mathbf{v_{2^k-1}} \end{pmatrix} \left(\mathbf{v_1}, \ldots, \mathbf{v_{2^k-1}}\right) =$$

$$= \begin{pmatrix} \mathbf{v_1 v_1} & \mathbf{v_1 v_2} & & \mathbf{v_1 v_{2^k-1}} \\ \mathbf{v_2 v_1} & \mathbf{v_2 v_2} & & \mathbf{v_2 v_{2^k-1}} \\ & & & \\ \mathbf{v_{2^k-1} v_1} & \mathbf{v_{2^k-1} v_2} & \cdots & \mathbf{v_{2^k-1} v_{2^k-1}} \end{pmatrix}.$$

unde produsele scalare sunt efectuate în Z_2.

Acum, $W = (w_1, w_2, \ldots, w_{2^k-1}) = (n_1, n_2, \ldots, n_{2^k-1})C$, înmulţire efectuată în $\mathcal{R}$.

Prin identificare rezultă $w_i = n_1 \mathbf{v_1 v_i} + n_2 \mathbf{v_2 v_i} + \ldots + n_{2^k-1} \mathbf{v_{2^k-1} v_i}$.

Deoarece toate produsele scalare $\mathbf{v_j v_i}$ sunt 0 sau 1, w_i este un număr natural. El este chiar ponderea cuvântului - cod care ocupă linia i în matricea K. Într-adevăr, avem

$$K_{2^k-1,n} = \begin{pmatrix} \mathbf{v_1 v_1} & \mathbf{v_1 v_1} & & \mathbf{v_1 v_{2^k-1}} \\ \mathbf{v_2 v_1} & \mathbf{v_2 v_1} & & \mathbf{v_2 v_{2^k-1}} \\ & & & \\ \mathbf{v_{2^k-1} v_1} & \mathbf{v_{2^k-1} v_1} & & \mathbf{v_{2^k-1} v_{2^k-1}} \end{pmatrix},$$

produsele scalare fiind în Z_2.

Deci ponderea cuvântului situat pe linia i în matricea K este:

$$\underbrace{\mathbf{v_i v_1} + \ldots + \mathbf{v_i v_1}}_{n_1} + \ldots + \underbrace{\mathbf{v_i v_{2^k-1}} + \ldots + \mathbf{v_i v_{2^k-1}}}_{n_{2^k-1}} = n_1 \mathbf{v_i v_1} + \ldots + n_{2^k-1} \mathbf{v_i v_{2^k-1}}.$$ □

Ne punem problema de a determina un cod atunci când se dă vectorul ponderilor cuvintelor sale (deci şi ponderea minimă, adică distanţa Hamming). Formal, din (6) se obţine

$$N = WC^{-1}, \tag{7}$$

înmulţire efectuată în $\mathcal{R}$. Formula este adevărată numai dacă se demonstrează că matricea C este inversabilă. În acest sens avem:

Teorema 3.18 *Matricea C este nesingulară. Inversa ei se calculează înlocuind în C pe 0 cu -1 şi împărţind toate elementele cu 2^{k-1}.*

Demonstraţie: **I**: Fiecare linie (deci şi coloană) din C conţine 1 pe 2^{k-1} poziţii. Două linii (coloane) distincte din C au în comun 1 în 2^{k-2} poziţii.

Într-adevăr, liniile lui C - la care se adaugă linia nulă - formează un spaţiu liniar cu 2^k cuvinte binare. Să considerăm submulţimea liniilor care au 1 în componentele i şi j. Ele formează un subspaţiu liniar cu $\frac{2^k}{2^2} = 2^{k-2}$ elemente deoarece, factorizând cu acest spaţiu se obţin patru clase de resturi, corespunzătoare componentelor (i, j) de forma $(0,0), (0,1), (1,0), (1,1)$.

Pentru a arăta că 1 apare pe fiecare linie din C de 2^{k-1} ori se raţionează similar.

II: În aceste condiţii, putem scrie

$$C^2 = \begin{pmatrix} 2^{k-1} & 2^{k-2} & & 2^{k-2} \\ 2^{k-2} & 2^{k-1} & \ldots & 2^{k-2} \\ & & \vdots & \\ 2^{k-2} & 2^{k-2} & & 2^{k-1} \end{pmatrix} = 2^{k-2}(I + J),$$

unde I este matricea unitate iar J este matricea pătrată cu toate elementele egale cu 1, ambele de ordin 2^k.

Se mai observă că $CJ = 2^{k-1}J$. Deci

$$I = \frac{1}{2^{k-1}}(2C^2 - 2^{k-1}J) = \frac{1}{2^{k-1}}(2C^2 - CJ) = C\frac{2C - J}{2^{k-1}} = CC^{-1}.$$

De aici rezultă că C este inversabilă şi $C^{-1} = \frac{1}{2^{k-1}}(2C - J)$. □

Exemplul 3.36 *Să luăm* $k = 3$, $n = 5$. *Vom avea:*

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix},$$

$$C = M^T M = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix},$$

$$C^{-1} = \frac{1}{4} \begin{pmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 \end{pmatrix}.$$

Dacă luăm codul generat de matricea $G = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$, *el are reprezentarea modulară* $N = (1,1,0,1,1,1,0)$. *Vectorul ponderilor cuvintelor - cod este* $W = (2,2,4,3,3.3,3)$. *Într-adevăr, cuvintele - cod sunt aşezate pe liniile matricii*

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

Să determinăm acum reprezentarea modulară a unui cod liniar binar în care toate cuvintele - cod au aceeaşi pondere w (şi deci distanţa minimă va fi $d = w$). Vom avea prin ipoteză $W = (\underbrace{w, w, \ldots, w}_{2^k - 1})$.

După cum am remarcat în demonstraţia Teoremei 3.18, 1 apare pe fiecare linie din C în 2^{k-1} poziţii. Rezultă (Teorema 3.18) că în C^{-1}, 1 va apare pe fiecare linie (coloană) tot în 2^{k-1} poziţii, în rest fiind -1. Deci

$$N = WC^{-1} = (\underbrace{w2^{-(k-1)}, \ldots, w2^{-(k-1)}}_{2^k-1})$$

Problema enunţată are soluţie dacă w se divide cu 2^{k-1}; în acest caz, reprezentarea modulară este de forma $N = (\underbrace{r, r, \ldots, r}_{2^k-1})$, adică fiecare tip de coloană apare în matricea generatoare de $r = w \cdot 2^{-(k-1)}$ ori.

Plecând de la aceste considerente, MacDonald a introdus o clasă de coduri pentru care reprezentarea modulară este de forma $0^i 1^j$. Mai exact,

n	d	N
$2^k - 1$	2^{k-1}	$(1, 1, 1, \ldots, 1)$
$2^k - 2$	$2^{k-1} - 1$	$(0, 1, 1, \ldots, 1)$
$2^k - 3$	$2^{k-1} - 2$	$(0, 0, 1, \ldots, 1)$
$\ldots$	$\ldots$	$\ldots$
$2^k - 2^u$	$2^{k-1} - 2^{u-1}$	$(\underbrace{0, \ldots, 0}_{2^u-1}, \underbrace{1, \ldots, 1}_{2^k-2^u})$

Aceste coduri au proprietatea că prezintă o distanţă minimă foarte apropiată de marginea Plotkin.

Să luăm de exemplu al treilea cod din tabelul de sus. Reamintim, marginea Plotkin este

$$d \leq \frac{nq^{k-1}(q-1)}{q^k - 1}.$$

În cazul codului MacDonald avem $q = 2$, $n = 2^k - 3$, $d = 2^{k-1} - 2$, deci

$$\frac{nq^{k-1}(q-1)}{q^k - 1} - d = \frac{(2^k - 3)2^{k-1}}{2^k - 1} - (2^{k-1} - 2) = \frac{2^k - 2}{2^k - 1} < 1.$$

3.6 Coduri Hadamard

Am definit anterior matricile Hadamard pentru a putea construi un algoritm de decodificare al codurilor $\mathcal{RM}(1, m)$. De fapt matricile Hadamard pot fi utilizate în mod direct la construcţia de coduri.

Definiţia 3.13 *Se numeşte matrice Hadamard $\mathcal{H}_n$ o matrice pătrată de ordinul n cu elemente 1 şi -1, ale cărei linii sunt ortogonale două câte două.*

Teorema 3.19 *Fiind dată matricea Hadamard $\mathcal{H}_n$, (n par) se poate defini un cod de distanţă minimă $d = \frac{n}{2}$, format din $2n$ cuvinte de lungime n.*

Demonstraţie: În matricea $\mathcal{H}_n$ vom înlocui 1 cu 0 şi -1 cu 1. Vom nota $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}$ vectorii care formează liniile noii matrici. Codul căutat este

$$C = \{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}, -\mathbf{v_1}, -\mathbf{v_2}, \ldots, -\mathbf{v_n}\}.$$

Acest cod (care nu este neapărat liniar) are $2n$ cuvinte de lungime n. Mai trebuie arătat că distanţa sa minimă este $n/2$.

Deoarece $\mathbf{v}$ şi $-\mathbf{v}$ diferă în toate componentele, distanţa Hamming dintre ele este n. Pe de-altă parte, din proprietatea matricilor Hadamard rezultă

$$\mathbf{v_i}(\pm\mathbf{v_j}) = 0,\ \forall i, j \in \{1, \ldots, n\},\ i \neq j.$$

Pentru ca această egalitate să fie adevărată, este necesar ca $\mathbf{v_i}$ şi $\mathbf{v_j}$ să coincidă în jumătate din componente şi să difere în cealaltă jumătate (atenţie, calculele se fac în $\mathcal{R}$, nu în Z_2 !). Rezultă că distanţa Hamming dintre cele două cuvinte este $\frac{n}{2}$. Deci $d = \frac{n}{2}$. □

Pentru $n = 32$ un astfel de cod a fost folosit de programul spaţial Mariner.

Teorema 3.20 *Dacă $\mathcal{H}_n$ este matrice Hadamard, atunci*

$$\mathcal{H}_{2n} = \begin{pmatrix} \mathcal{H}_n & \mathcal{H}_n \\ \mathcal{H}_n & -\mathcal{H}_n \end{pmatrix}$$

este matrice Hadamard.

Demonstraţie: Evident, $\mathcal{H}_{2n}$ este o matrice $2n \times 2n$ cu elemente ± 1. Mai trebuie arătat că liniile sale sunt ortogonale două câte două.

Avem: $[\mathbf{v_i}, \mathbf{v_i}][\mathbf{v_i}, -\mathbf{v_i}] = \mathbf{v_i}\mathbf{v_i} - \mathbf{v_i}\mathbf{v_i} = n - n = 0$.
De asemenea, pentru $i \neq j$, $[\mathbf{v_i}, \mathbf{v_i}][\mathbf{v_j}, \pm\mathbf{v_j}] = [\mathbf{v_i}, \mathbf{v_j}] \pm [\mathbf{v_i}, \mathbf{v_j}] = 0 \pm 0 = 0$. □

Pe baza acestei teoreme, putem construi coduri Hadamard a căror distanţă să fie oricât de mare. Este suficient să găsim o matrice Hadamard iniţială, pe care (folosind Teorema 3.20) să o "dilatăm" ulterior cât este nevoie. O astfel de matrice Hadamard iniţială a fost dată în cadrul construcţiei codurilor RM: ea este

$$\mathcal{H}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Pentru construcţia altor matrici (şi coduri) Hadamard mai generale au fost elaborate diverse tehnici. Una din cele mai cunoscute este cea a lui Paley şi Hall.

Definiţia 3.14 *O matrice pătrată M de ordin n cu 0 pe diagonala principală şi ± 1 înafara ei, astfel ca $MM^T = (n-1)I_n$, se numeşte matrice de conjunctură (conference matrix).*

Exemplul 3.37 *Matricile* $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & 1 & 1 & 1 \\ -1 & 0 & -1 & 1 \\ -1 & 1 & 0 & -1 \\ -1 & -1 & 1 & 0 \end{pmatrix}$

sunt matrici de conjunctură.

Fie q un număr prim impar: pe $Z_q = \{0, 1, \ldots, q-1\}$ definim funcţia reziduu χ astfel:

$$\chi(x) = \begin{cases} 0 & \text{dacă } x = 0 \\ 1 & \text{dacă } x \text{ este un pătrat nenul} \\ -1 & \text{în rest} \end{cases}$$

Teorema 3.21 *Matricea Paley $S_q = (s_{ij})$ de ordin q definită prin $s_{ij} = \chi((q+i-j) \bmod q)$ $(0 \le i, j < q)$ verifică proprietăţile:*

1. $S_q J_q = J_q S_q = \mathbf{0}$;
2. $S_q S_q^T = qI_q - J_q$;
3. $S_q^T = (-1)^{\frac{q-1}{2}} S_q$.

Toate calculele se fac în $\mathcal{Z}$; s-a notat cu I_q matricea unitate de ordin q şi cu J_q matricea pătrată de ordin q având toate elementele egale cu 1.

Demonstraţie: Este lăsată ca exerciţiu.

Exemplul 3.38 *Pentru $q = 3$, matricea Paley este*

$$S_3 = \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix},$$

iar pentru $q = 5$,

$$S_5 = \begin{pmatrix} 0 & 1 & -1 & -1 & 1 \\ 1 & 0 & 1 & -1 & -1 \\ -1 & 1 & 0 & 1 & -1 \\ -1 & -1 & 1 & 0 & 1 \\ 1 & -1 & -1 & 1 & 0 \end{pmatrix}.$$

De remarcat că $S_3^T = -S_3$, $S_5^T = S_5$ *(conform Teoremei 3.21).*

Matricea S *folosită în definirea codurilor Golay ternare este matricea Paley* S_5 *(în care, deoarece se lucrează în* Z_3, $-1 = 2$*).*

Teorema 3.22 *Fie* q *un număr prim impar astfel încât* $q \equiv 3 \ (mod\ 4)$. *Atunci există o matrice Hadamard de ordin* $q + 1$.

Demonstraţie: Din ipoteză şi Teorema 3.21, se poate defini matricea Paley S_q de ordin q. Construim matricea M de ordin $q + 1$ astfel:

$$M = \begin{pmatrix} 0 & 1 & & 1 \\ -1 & & & \\ -1 & & S_q & \\ -1 & & & \end{pmatrix}$$

Se verifică faptul că M este o matrice de conjunctură cu proprietatea $M^T = -M$ ($q \equiv 3\ mod\ 4$ conduce la relaţia $S_q^T = -S_q$). Fie $H = I_{q+1} + M$.

Avem $HH^T = (I_{q+1} + M)(I_{q+1} + M)^T = (I_{q+1} + M)(I_{q+1} + M^T) = I_{q+1} + M + M^T + MM^T = I_{q+1} + qI_{q+1} = (q+1)I_{q+1}$, deci orice două linii distincte ale lui H sunt ortogonale. Rezultă că H este matrice Hadamard de ordin $q + 1$. □

Folosind acum Teorema 3.19, se pot construi coduri Hadamard de lungime $q+1$ unde q este număr prim care verifică condiţia $q \equiv 3 \ (mod\ 4)$.

Plecând de la matricile Paley se pot construi şi alte coduri Hadamard, modificând puţin Teorema 3.19 (pentru a admite şi coduri de lungime impară).

Fie S_n o matrice Paley de ordin n (conform Teoremei 3.21). Un cod Hadamard de ordin n poate fi definit cu $\mathbf{0}, \mathbf{1}$ şi cu liniile matricilor $\frac{1}{2}(S_n + I_n + J_n)$, $\frac{1}{2}(-S_n + I_n + J_n)$. Acesta este un cod care conţine $2n + 2$ cuvinte - cod de lungime n şi are distanţa minimă $d = \frac{n-1}{2}$.

Exemplul 3.39 *Să considerăm* $q = 11$. *Matricea Paley asociată este:*

$$S_{11} = \begin{pmatrix} 0 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 \\ 1 & 0 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\ -1 & 1 & 0 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 0 & -1 & 1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 0 & -1 & 1 & -1 & -1 & -1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 0 & -1 & 1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 & -1 & 1 & 0 & -1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 & 1 & -1 & 1 & 0 & -1 & 1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 & 0 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 & 0 & -1 \\ -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 & 0 \end{pmatrix}$$

Pe baza ei se pot construi două coduri Hadamard. Primul se bazează pe o matrice Hadamard construită conforma Teoremei 3.22:

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

Acest cod are 24 *cuvinte - cod de lungime* 12 *şi distanţă minimă* 6 *(pentru construcţie, vezi Teorema 3.19).*

Al doilea cod Hadamard posibil este de lungime 11 *şi are tot* 24 *cuvinte - cod:*

{00000000000, 10100011101, 11010001110, 01101000111, 10110100011, 11011010001, 11101101000, 01110110100, 00111011010, 00011101101, 10001110110, 01000111011, 00100011101, 10010001110, 01001000111, 10100100011, 11010010001, 11101001000, 01110100100, 00111010010, 00011101001, 10001110100, 01000111010, 11111111111,}

Distanţa sa minimă este 5.

3.7 Coduri produs

Să considerăm un cod de lungime $n = n_1 n_2$. În loc să scriem cuvintele sale sub formă de linii de lungime n, le putem reprezenta ca matrici cu n_1 linii şi n_2 coloane. Un mod simplu de a realiza acest lucru este de a scrie cuvântul cod $\mathbf{a} = a_0 a_1 \ldots a_{n-1}$ ca o matrice $X = [x_{ij}], 0 \leq i \leq n_1 - 1,\ 0 \leq j \leq n_2 - 1$, unde $x_{ij} = a_{in_2+j}$. Aceasta va fi numită *scriere canonică.*

Definiţia 3.15 *Fie C_1, C_2 două coduri liniare de lungime n_1 şi respectiv n_2. Se construieşte codul C de lungime $n_1 \cdot n_2$ ca mai sus (folosind scrierea canonică). Spunem că $C = C_1 \times C_2$ este codul produs al lui C_1 cu C_2 dacă şi numai dacă C conţine toate cuvintele - cod în care scrierea canonică verifică proprietăţile:*

- *Fiecare coloană a unei matrici este un cuvânt - cod din C_1;*
- *Fiecare linie a unei matrici este un cuvânt - cod din C_2.*

De remarcat că prin permutarea lui C_1 cu C_2 se obţine un cod echivalent.

Teorema 3.23 *Dacă C_1 este un (n_1, k_1) - cod liniar şi C_2 este un (n_2, k_2) - cod liniar, atunci $C_1 \times C_2$ este un $(n_1 n_2, k_1 k_2)$ - cod liniar.*

Demonstraţie: Fie G_i $(i = 1, 2)$ matricile generatoare ale celor două coduri, pe care le presupunem în forma eşalonat canonică, deci $G_i = [I_{k_i} B_{n_i - k_i}]$. Vom nota cu $\mathbf{g_i^{(1)}}$ $(1 \leq i \leq k_1)$ respectiv $\mathbf{g_i^{(2)}}$ $(1 \leq i \leq k_2)$ liniile celor două matrici.

Definim matricile X_{ij} $(1 \leq i \leq k_1,\ 1 \leq j \leq k_2)$ de dimensiune $n_1 \times n_2$, astfel:

- Primele k_1 linii sunt $\mathbf{0}$ cu excepţia liniei i care este $\mathbf{g_j^{(2)}}$;
- Primele k_2 coloane sunt $\mathbf{0}$ cu excepţia coloanei j care este $\mathbf{g_i^{(1)^T}}$.
- Pentru $k > k_1$, linia k este $g_{ik}^{(1)} \mathbf{g_j^{(2)}}$;
- Pentru $k > k_2$, coloana k este $g_{jk}^{(2)} \mathbf{g_i^{(1)^T}}$.

Schematic, o astfel de matrice arată astfel:

$$X_{ij} = \left(\begin{array}{ccc|ccc} 0 & & 0 & 0 & & 0 \\ & \dots & \dots & \dots & \dots & \dots \\ & 1 & & x & x & x \\ \dots & & \dots & \dots & & \dots \\ 0 & \dots & 0 & 0 & \dots & 0 \\ \hline 0 & x & 0 & & & \\ \dots & x & \dots & & X & \\ 0 & x & 0 & & & \end{array}\right)$$

Evident, X_{ij} reprezintă un cuvânt - cod din $C_1 \times C_2$ şi orice alt cuvânt - cod este o combinaţie liniară formată din aceste matrici. Deci X_{ij} $(1 \leq i \leq k_1,\ 1 \leq j \leq k_2)$ constituie o bază a codului $C_1 \times C_2$.

Submatricile formate din primele k_1 linii şi k_2 coloane formează mulţimea mesajelor de informaţie. □

Exemplul 3.40 *Să considerăm codurile liniare binare* C_1, C_2 *generate de matricile*

$$G_1 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \quad \textit{respectiv} \quad G_2 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Deci $n_1 = 3$, $n_2 = 4$, $k_1 = 2$, $k_2 = 3$. *Sunt* 6 *matrici* X_{ij}, *toate de dimensiune* 3×4*:*

$$X_{11} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}, \ X_{12} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \ X_{13} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

$$X_{21} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}, \ X_{22} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \ X_{23} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Să considerăm mesajul de informaţie 101110 *pe care îl aşezăm sub forma unei matrici* 2×3 : $\begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$. *Matricea - cod va fi*

$$X_{11} + X_{13} + X_{21} + X_{22} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} +$$

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

Se observă că liniile sunt cuvinte - cod din C_2, iar coloanele sunt cuvinte - cod din C_1.

Propoziţia 3.8 *Matricea generatoare a codului produs $C_1 \times C_2$ este $G_1 \times G_2$ unde G_1 şi G_2 sunt matricile generatoare ale celor două coduri, iar "$\times$" este produsul Kronecker.*

Demonstraţie: Fie $(n_1, k_1), (n_2, k_2)$ cele două coduri, cu matricile generatoare G_1 respectiv G_2, şi să considerăm matricea $G_1 \times G_2$ obţinută prin produsul Kronecker. Dacă se ia linia $k_2 i + j$ din această matrice şi se reprezintă sub formă canonică ca o matrice $n_1 \times n_2$, se obţine exact matricea X_{ij} din construcţia Teoremei 3.23. Cum X_{ij} reprezintă o bază pentru codul $C_1 \times C_2$, rezultă că $G_1 \times G_2$ este matricea sa generatoare.

Din acest motiv, în primii ani, codurile produs erau numite şi coduri Kronecker. □

Exemplul 3.41 *Matricea generatoare a codului produs $C_1 \times C_2$ din Exemplul 3.40 este:*

$$G = G_1 \times G_2 = \begin{pmatrix} G_2 & \mathbf{0} & G_2 \\ \mathbf{0} & G_2 & G_2 \end{pmatrix} =$$

$$\left(\begin{array}{cccc|cccc|cccc} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}\right).$$

Se observă imediat că prin "plierea" celor patru linii din această matrice se obţin matricile X_{ij} din Exemplul 3.40.

Să considerăm din nou mesajul de informaţie 101110. *Prin înmulţirea cu matricea G se ajunge la cuvântul - cod* 101111000111 *care - scris ca o matrice 3×4 - coincide cu rezultatul din Exemplul 3.40.*

Teorema 3.24 *Distanţa minimă a codului $C_1 \times C_2$ este egală cu $d_1 \cdot d_2$ (d_1 şi d_2 fiind distanţele minime ale codurilor C_1 respectiv C_2).*

Demonstraţie: Dacă C_i are lungimea n_i $(i = 1, 2)$, să considerăm o matrice X reprezentând un cuvânt - cod nenul din $C_1 \times C_2$ (am văzut că asemenea cuvinte există). Putem găsi deci cel puţin o linie cu minim d_2 elemente nenule. Fiecare astfel de element se află pe o coloană cu cel puţin d_1 elemente nenule; deci matricea X are minim $d_1 \cdot d_2$ elemente nenule. □

Exemplul 3.42 *Cele două coduri din Exemplul 3.40 au* $d_1 = d_2 = 2$ *(se găsesc imediat cuvintele - cod şi se observă că ponderea lor nenulă minimă este 2). Deci codul produs va avea distanţa* $d = 2 \cdot 2 = 4$. *De remarcat că - deşi* C_1 *şi* C_2 *nu pot corecta erori,* $C_1 \times C_2$ *este capabil să corecteze o eroare.*

Codurile produs - cu toate că au fost definite de la începutul anilor '60, au devenit foarte importante odată cu folosirea lor în transmisiile telefonice fără fir şi codificarea informaţiei pe CD - uri. Vom relua ulterior - pe larg - analiza acestor coduri.

3.8 Coduri optimale

Fie mulţimea $C \subseteq Z_q^n$. Se numeşte *diametrul* mulţimii C numărul

$$d(C) = min\{d(\mathbf{x}, \mathbf{y}) | \ \mathbf{x}, \mathbf{y} \in C, \ \mathbf{x} \neq \mathbf{y}\}.$$

De remarcat că, dacă C este cod liniar, atunci $d(C)$ este chiar distanţa minimă a codului.

Definiţia 3.16 *C este mulţime optimală de distanţă minimă d dacă:*

- $d(C) = d$;
- $\forall \mathbf{x} \in Z_q^n \setminus C, \ d(C \cup \{\mathbf{x}\}) < d$.

Exemplul 3.43 *În* Z_2^6, *următoarele mulţimi sunt optimale de distanţă minimă 3:*

$$A = \{000000, 010101, 101010, 111111\}$$
$$B = \{000000, 101010, 011001, 000111, 110100, 111111\}$$

Definiţia 3.17 *Se numeşte mulţime optimală cardinal maximală de distanţă d o mulţime optimală din Z_q^n, având un număr maxim de elemente.*

Vom nota acest număr cu $a(n, d, q)$.

Exemplul 3.44 *Mulţimile date în Exemplul 3.43 sunt optimale dar nu cardinal maximale pentru distanţa 3. O mulţime optimală cardinal maximală din* Z_2^6 *pentru* $d = 3$ *este:*
$C = \{000000, 001011, 010101, 011110, 100110, 101101, 110011, 111000\}$, *deci* $a(6, 3, 2) = 8$.

Definiţia 3.18 *Un cod liniar C de distanţă minimă d este optimal dacă C este mulţime optimală cardinal maximală de distanţă d.*

Spunem că un astfel de cod satisface *condiţia max - min*, deoarece are un număr maxim de vectori aflaţi la distanţa minimă d.

Nu se cunosc algoritmi de construcţie a unor astfel de coduri. În această secţiune vom da numai câteva condiţii necesare pentru definirea codurile optimale.

Teorema 3.25 *Într-un cod optimal C de lungime n şi distanţă minimă d,*

$$a(n,d,q) \leq q \cdot a(n-1,d,q).$$

Demonstraţie: Putem presupune – fără a micşora generalitatea – că primul caracter al codului este caracter de informaţie.

Fie K mulţimea cuvintelor din C care au 0 pe prima componentă. Cum C este cod liniar, iar pe prima componentă pot apare q caractere distincte, va rezulta

$$card(K) = \frac{a(n,d,q)}{q}.$$

K este un cod liniar cu distanţa minimă d, în care toate cuvintele - cod au 0 pe prima poziţie. Dacă eliminăm această componentă, se obţine un cod din Z_q^{n-1}, tot de distanţă d - cu acelaşi număr de cuvinte. Acest cod nu este neapărat optimal, deci condiţia va fi

$$a(n-1,d,q) \geq \frac{a(n,d,q)}{q}$$

Mai trebuie făcută observaţia că $a(n-1,n-1,q) = q$ pentru că un cod cu repetiţie este optimal, având distanţa minimă egală cu lungimea cuvintelor - cod. □

Corolarul 3.3 *Dacă $j \geq d$, atunci $q^{n-j}a(j,d,q) \geq a(n,d,q)$.*

Demonstraţie: Din Teorema 3.25 rezultă inegalităţile:

$$\begin{aligned} &q \cdot a(n-1,d,q) \geq a(n,d,q) \\ &q \cdot a(n-2,d,q) \geq a(n-1,d,q) \\ &\dots \\ &\underline{q \cdot a(n-i,d,q) \geq a(n-i+1,d,q)} \end{aligned}$$

Prin înmulţire se obţine $q^i \cdot a(n-i,d,q) \geq a(n,d,q)$, după care se face notaţia $n-i=j$. □

Teorema 3.26 *Fie $A_{n,k} \subseteq Z_q^n$ un cod liniar optimal de distanţă minimă d. Dacă $n < \frac{qd}{q-1}$, atunci numărul k al simbolurilor de informaţie verifică inegalitatea*

$$k \leq n - \frac{dq-1}{q-1} + 1 + log_q d.$$

Demonstraţie: Deoarece $A_{n,k}$ este cod liniar, numărul de simboluri de informaţie va fi dat de relaţia $a(n,d,q) = q^k$.

Rescriem inegalitatea lui Plotkin (Teorema 2.10) sub forma $d(q^k - 1) \leq nq^{k-1}(q-1)$, sau

$$q^{k-1}(dq + n - nq) \leq n.$$

Deoarece - din ipoteză - $dq + n - nq > 0$, avem $q^k \leq \frac{dq}{dq+n-nq}$, sau

$$a(n,d,q) \leq \frac{dq}{dq+n-nq}.$$

Fie $\frac{dq-1}{q-1} = i + f$ unde $i = \left[\frac{dq-1}{q-1}\right]$, $f = \left\{\frac{dq-1}{q-1}\right\}$.

Deci $qd - 1 = (q-1)i + (q-1)f$ şi avem

$$a(i,d,q) \leq \frac{dq}{dq+i-iq} = \frac{dq}{dq-(q-1)i} = \frac{dq}{1+(q-1)f}.$$

Folosind Corolarul 3.3 şi $i \geq d$ (deoarece $\left[\frac{dq-1}{q-1}\right] = \left[d + \frac{d-1}{q-1}\right] \geq d$), rezultă

$$a(n,d,q) \leq q^{n-i}a(i,d,q) \leq \frac{q^{n-\frac{dq-1}{q-1}+f}}{1+(q-1)f}qd.$$

Dezvoltând în serie Taylor, $q^f \leq 1 + (q-1)f \quad (f < 1)$ şi deci

$$a(n,d,q) \leq q^{n-\frac{dq-1}{q-1}}qd,$$

sau $q^k \leq q^{n-\frac{dq-1}{q-1}}qd$, de unde - prin logaritmare - se obţine relaţia din enunţ. □

3.9 Exerciţii

3.1 *Fie* $(8,4)$ *- codul Hamming binar extins. Decodificaţi cuvintele*
10101010 11010110 11111111.

3.2 *Construiţi codurile Hamming ternare pentru* $r = 2,\ 3$ *şi determinaţi decodificarea pe baza sindromurilor.*

3.3 *Construiţi* $(5,3)$ *- codul Hamming peste* Z_4. *Determinaţi toate cuvintele - cod şi tabela de decodificare cu sindromuri. Decodificaţi cuvintele:* 11223, 32101 2222 1100.

3.4 *Demonstraţi afirmaţiile din Corolarul 3.1.*

3.5 *Arătaţi că* C_{24} *conţine un cuvânt cu toate componentele egale cu* 1 *şi nici un cuvânt de pondere* 20.

Demonstraţi că numărătorul de ponderi al lui C_{24} *este:*

$$1 + 759X^8 + 2576X^{12} + 759X^{16} + X^{24}.$$

3.6 *De ce în Algoritmul D se face extensia cuvintelor la cuvinte de pondere impară ? Justificaţi prin exemple corectitudinea afirmaţiei.*

3.7 *În codul Golay extins* C_{24} *să se decodifice - dacă este posibil, cuvintele:*

[111000000000, 011011011011]	[111111000000, 100011100111]
[111111000000, 101011100111]	[111111000000, 111000111000]
[111000000000, 110111001101]	[110111001101, 111000000000]
[000111000111, 101000101101]	[110000000000, 101100100000]

3.8 *Folosind* C_{23}, *decodificaţi cuvintele:*

[101011100000, 10101011011]	[101010000001, 11011100010]
[100101011000, 11100010000]	[011001001001, 01101101111]

3.9 *Fie* H *matricea de control a unui* $(4,2)$ *- cod Hamming ternar (Exemplul 3.4),* I *matricea unitate de ordin* 4 *şi* J *matricea de ordin* 4 *cu toate elementele* 1. *Să se arate că*

$$G = \begin{pmatrix} J - I & I & I \\ \mathbf{0} & H & -H \end{pmatrix}$$

generează un $(12,6)$ *- cod liniar cu* $d = 6$ *(echivalent cu codul Golay ternar extins).*

3.10 *Ce polinom boolean are ultima linie a tabelei de adevăr:*

(a) 10100110
(b) 1010011010100110
(c) 0101001110011100

3.11 *Scrieţi tabela de adevăr a polinomului boolean* $\mathbf{1 + x_0 + x_1x_2}$*:*

1. *Ca funcţie de trei variabile;*

2. *Ca funcţie de patru variabile.*

3.12 *Fiind dat codul* $\mathcal{RM}(1,3)$*, codificaţi toate mesajele de informaţie posibile.*

Acelaşi lucru pentru codul $\mathcal{RM}(2,3)$.

3.13 *Calculaţi numărul de* 2 *- spaţii afine în geometria euclidiană peste* Z_2^m.

3.14 *Este orice funcţie booleană funcţia caracteristică a unui anumit spaţiu afin ? Caracterizaţi astfel de funcţii.*

3.15 *Orice funcţie caracteristică a unui* $(r+1)$ *- spaţiu afin aparţine codului dual lui* $\mathcal{RM}(r,m)$.

3.16 *Să se arate că* $\mathcal{RM}(2,5)$ *este auto - dual.*
Să se determine toate codurile $\mathcal{RM}$ *auto - duale.*

3.17 *Reluaţi Exemplul 3.27. De ce sunt* 7 *ecuaţii ortogonale pentru fiecare* x_i *?*

3.18 *În codul Hamming* $(15,11)$ *să se decodifice mesajele*
111100000000000, 011101000111001.

3.19 *Să se decodifice cuvântul* 01111100 *în* $\mathcal{RM}(1,3)$*. Verificaţi corectitudinea decodificării.*

3.20 *Să se scrie toate relaţiile de determinare din codul* $\mathcal{RM}(2,4)$.

3.21 *Să se decodifice* 1111111011111111 *în* $\mathcal{RM}(2,4)$.

3.22 *Arătaţi că orice hiperplan L în geometria euclidiană peste Z_2^m are drept complement $Z_2^m \setminus L$ tot un hiperplan.*

3.23 *Să se calculeze H_4^i pentru $i = 1, 2, 3, 4$.*

3.24 *În codul $\mathcal{RM}(1,4)$, să se decodifice cuvintele:*
1011011001101001, 1111000001011111.

3.25 *Să se construiască vectorii ponderilor pentru codurile Hamming $(7,4)$ şi $\mathcal{RM}(1,3)$.*

3.26 *Folosind vectorul ponderilor, să se construiască numărătorul de ponderi al unui cod liniar.*

3.27 *Să se construiască matricile generatoare şi matricile K pentru codurile McDonald cu $k = 2, 3, 4$.*

3.28 *Să se arate că dacă $\mathcal{H}_n$ este o matrice Hadamard, atunci $\mathcal{H}_n \mathcal{H}_n^T = nI_n$.*

3.29 *Să se construiască o matrice Hadamard de ordin 20.*

3.30 *Să se arate că matricea Hadamard $\mathcal{H}_8$ generează $(8,4)$ - codul Hamming binar extins.*

3.31 *Să se construiască codurile produs pentru codul cu repetiţie de lungime n (C_1) şi codul cu repetiţie de lungime p (C_2).*

3.32 *Să se construiască matricea generatoare a codului produs $C \times C$ unde C este codul Hamming $(7,4)$.*
Aceeaşi problemă pentru codul $\mathcal{RM}(1,3)$.

3.33 *Demonstraţi Teorema 3.21.*

3.34 *Folosind matricile Paley, să se construiască codurile Hadamard de ordin 5, 7 şi respectiv 10.*

3.35 *Să se arate că un cod binar liniar optimal de distanţă minimă d are cel puţin $2d - 1 - log_2 d$ simboluri de control.*

Capitolul 4

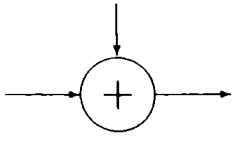
Circuite liniare şi extensii Galois

4.1 Definirea circuitelor liniare

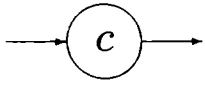
În această secţiune presupunem că toate calculele se realizează într-un corp de caracteristică q (q număr prim).

Un *circuit liniar* este un graf orientat, cu trei tipuri de noduri (conectori):

- *sumator*: este un conector cu cel puţin două intrări şi o ieşire. Efectul este acela de a scoate suma modulo q a elementelor aflate la intrare. Operaţia se execută instantaneu. Un sumator (cu două intrări) se notează astfel:

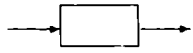


- *Multiplicator*: este un conector cu o singură intrare şi o ieşire unde se obţine rezultatul înmulţirii cu $c \in Z_q$ a elementului de la intrare. Operaţia se execută (de asemenea) instantaneu. Notaţie:



- *Element de înmagazinare*: este un registru buffer cu o intrare şi o ieşire; efectul este întârzierea cu un tact a intrării: elementul

$a \in Z_q$ intră în momentul (tactul) k şi iese în momentul (tactul) $k+1$. Notaţie:



Aceste trei tipuri de conectori se pot grupa arbitrar - folosind eventual şi noduri obişnuite. Utilizarea lor conduce la obţinerea de circuite liniare care pot realiza automat operaţii uzuale cu polinoame: adunări, scăderi, înmulţiri şi împărţiri.

În cele ce urmează, vom identifica un vector cu $n+1$ componente
$$\mathbf{a} = (a_0, a_1, \ldots, a_n)$$
cu polinomul de gradul n cu o variabilă
$$a(x) = a_0 + a_1 X + \ldots + a_n X^n.$$

În cursul transmisiei mesajului **a**, ordinea de prelucrare (transmisie) se face după puterile descrescătoare ale lui X; deci, sensul de circulaţie a semnalelor va fi:

$$a_0 \quad a_1 \qquad\qquad a_n \longrightarrow$$

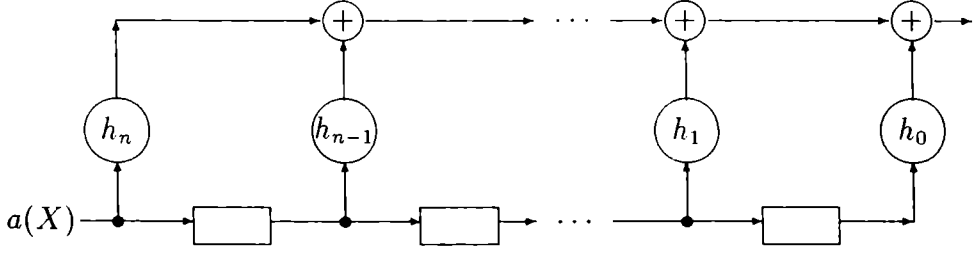
4.1.1 Construcţia unor circuite liniare uzuale

A. *Circuit de înmulţire cu un polinom.*

Fie $h(X) = h_0 + h_1 X + \ldots + h_n X^n$ un polinom fixat şi $a(X) = a_0 + a_1 X + \ldots + a_k X^k$ un polinom arbitrar. Coeficienţii ambelor polinoame pot fi luaţi într-un inel arbitrar (din considerente pur aplicative, acesta este de obicei Z_q, dar construcţiile rămân valabile şi în cazul general).

Un circuit de înmulţire cu $h(X)$ este:

Figura 4.1:



Iniţial elementele de înmagazinare conţin 0; coeficienţii polinomului $a(X)$ intră prin stânga jos - după puterile descrescătoare ale lui X - câte

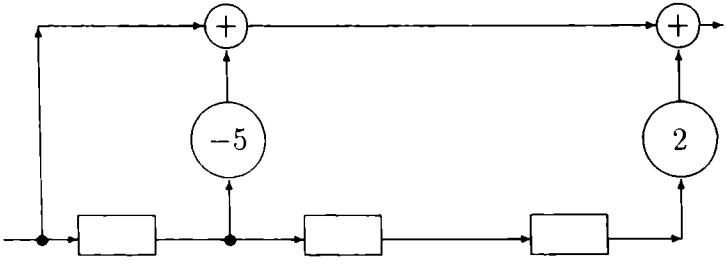
unul la un tact, iar coeficienţii produsului ies în dreapta sus. Circuitul funcţionează $n+k+1$ tacţi; după $k+1$ tacţi, la intrare se introduc zerouri în cei n tacţi rămaşi, până ce elementele de înmagazinare conţin din nou numai 0.

De remarcat că un element de înmagazinare funcţionează similar unei variabile de memorie (introducerea unei valori face să dispară vechea valoare din locaţie).

Circuitul lucrează după formula de înmulţire obişnuită:

$a(X)h(X) = a_0h_0+(a_0h_1+a_1h_0)X+\ldots+(a_{k-1}h_n+a_kh_{n-1})X^{k+n-1}+a_kh_nX^{n+k}$.

Exemplul 4.1 *Să luăm polinomul* $h(X) = X^3 - 5X^2 + 2 \in Z[X]$ *(vectorul corespunzător este* $\mathbf{h} = (2, 0, -5, 1)$*). Circuitul de înmulţire cu* $h(X)$ *va fi:*



De remarcat că multiplicarea cu 1 *se face prin arc simplu (fără conector de înmulţire), iar multiplicarea cu* 0 *revine la suprimarea arcului. În cazul operaţiilor cu polinoame din* $Z_2[X]$*, toţi multiplicatorii se reduc la aceste două cazuri.*

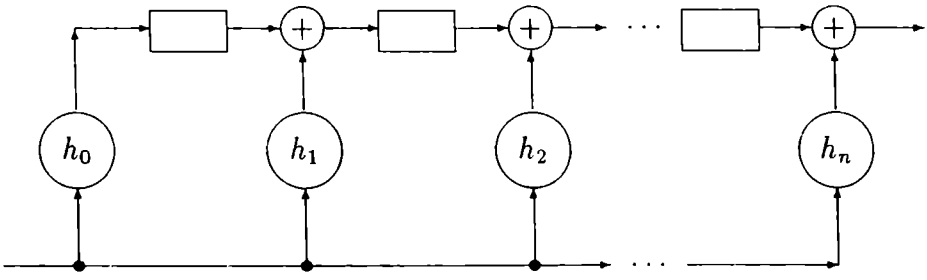
Dacă dorim să înmulţim $h(X)$ *cu polinomul* $a(X) = 2X + 8$ *(vector* $\mathbf{a} = (8, 2)$*), circuitul liniar va funcţiona* 5 *tacţi (până când toate elementele de înmagazinare revin la* 0*); comportarea sa este relatată de tabelul:*

Nr. tact	*Intrare*	*Înmagazinare*			*Ieşire*
0	0	0	0	0	0
1	2	0	0	0	2
2	8	2	0	0	−2
3	0	8	2	0	−40
4	0	0	8	2	4
5	0	0	0	8	16
6	0	0	0	0	0

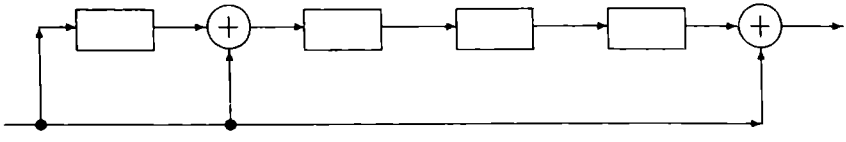
Deci, polinomul produs este $2X^4 - 2X^3 - 40X^2 + 4X + 16$*; sau - în termeni de vectori,* $(16, 4, -40, -2, 2)$.

Aceeaşi operaţie - de înmulţire cu un polinom fixat $h(X)$ este realizată şi cu circuitul liniar:

Figura 4.2:



Exemplul 4.2 *Circuitul liniar care realizează înmulţirea cu polinomul* $h(X) = 1 + X + X^4$ *peste* Z_2 *este*

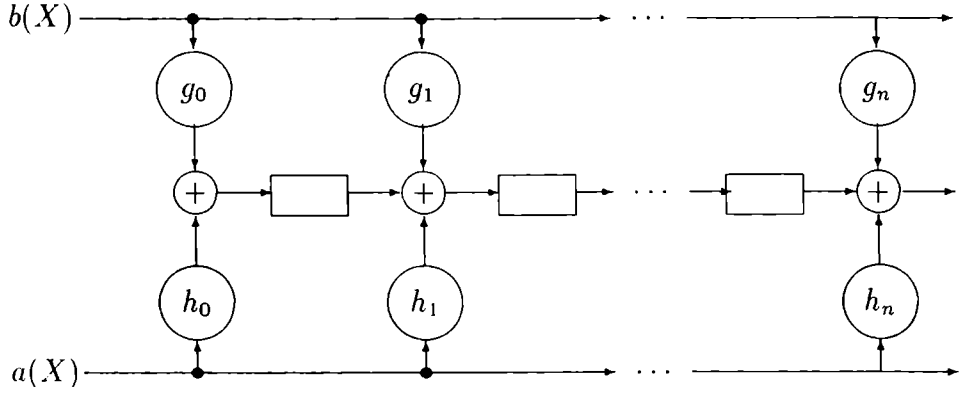


deoarece vectorul corespunzător polinomului $h(X)$ *este* $\mathbf{h} = (1, 1, 0, 0, 1)$.

B. Fie $h(X) = h_0 + h_1X + \ldots + h_nX^n$ şi $g(X) = g_0 + g_1X + \ldots + g_nX^n$ două polinoame fixate. Un circuit liniar care să realizeze operaţia

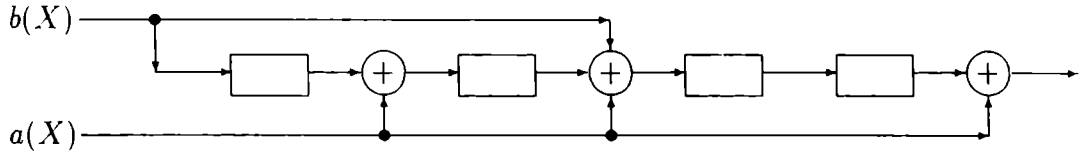
$$a(X)h(X) + b(X)g(X)$$

pentru două polinoame arbitrare $a(X), b(X)$, este:



De remarcat că polinoamele $g(X)$ şi $h(X)$ pot avea grade diferite; în acest caz se alege n ca fiind gradul cel mai mare şi se completează celălalt polinom cu coeficienţi nuli până la gradul n.

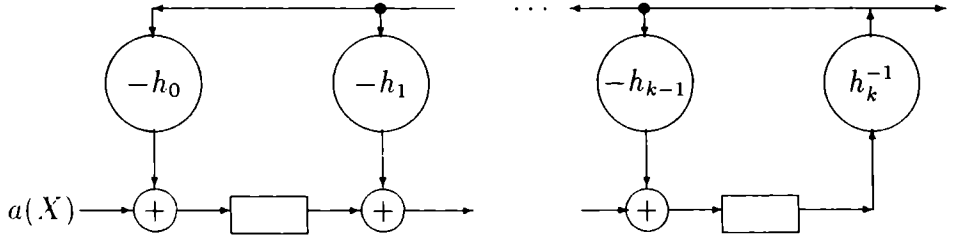
Exemplul 4.3 *Fie polinoamele* $g(X) = 1 + X^2$, $h(X) = X + X^2 + X^4$ *cu coeficienţi în* Z_2*: vectorii corespunzători vor fi* $\mathbf{h} = (0,1,1,0,1)$, $\mathbf{g} = (1,0,1,0,0)$, *iar circuitul liniar care realizează expresia* $a(X)h(X) + b(X)g(X)$ *este:*



C. *Circuit liniar de împărţire cu un polinom.*

Fie polinomul (fixat) $h(X) = h_0 + h_1X + \ldots + h_kX^k$, unde $h_k \neq 0$. Un circuit liniar care realizează împărţirea unui polinom arbitrar $a(X) = a_0 + a_1X + \ldots + a_nX^n$ la $h(X)$ este:

Figura 4.3:

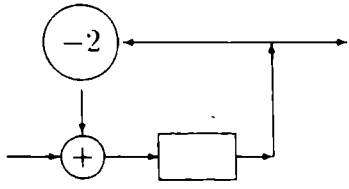


Iniţial, toate cele k elemente de înmagazinare conţin 0. La intrare, timp de $n+1$ tacţi se introduc coeficienţii polinomului $a(X)$ (în ordinea descrescătoare a puterilor). În primii k tacţi, ieşirea va scoate numai 0. Primul coeficient nenul apare la ieşire la momentul $k+1$ şi este $a_nh_k^{-1}$ (operaţii în inelul coeficienţilor celor două polinoame) - coeficientul termenului X^{n-k} din cât.

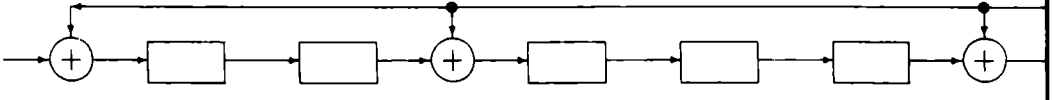
Pentru fiecare coeficient q_j al câtului, polinomul $q_jh(X)$ trebuie scăzut din deîmpărţit; această operaţie se realizează printr-un procedeu de conexiune inversă (*feedback*).

După momentul $n+1$ (când au intrat toţi coeficienţii polinomului $a(X)$) la ieşire s-a obţinut câtul $q(X)$ al împărţirii lui $a(X)$ la $h(X)$, iar în elementele de înmagazinare se găsesc coeficienţii restului.

Exemplul 4.4 *Un circuit liniar de împărţire (peste Q) la polinomul $h(X) = 2 + X$ este:*



Exemplul 4.5 *Fie polinomul $h(X) = 1 + X^2 + X^5 + X^6 \in Z_2[X]$. Circuitul care realizează împărţirea cu acest polinom este:*



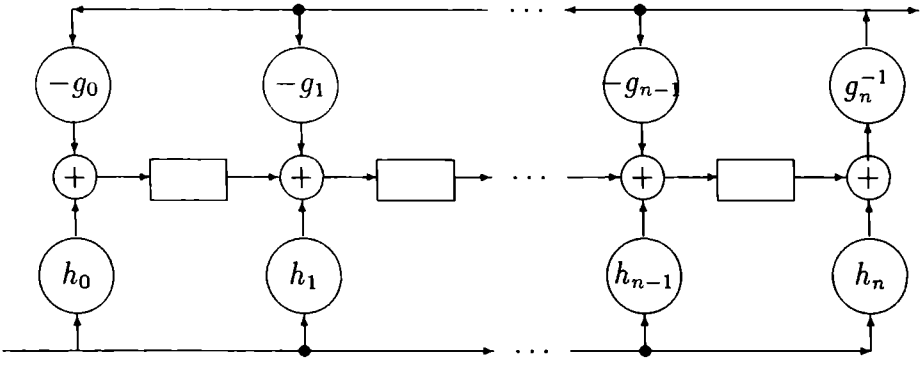
Să luăm polinomul $a(X) = 1 + X^4 + X^5 + X^7 + X^9 + X^{10}$ pe care-l vom împărţi (în Z_2) la polinomul $h(X)$. Se obţine câtul $X^4 + X$ şi restul $X^5 + X^3 + X + 1$.

Circuitul funcţionează 11 tacţi. Etapele prin care trece circuitul pentru efectuarea acestei împărţiri sunt reprezentate de tabelul următor:

Nr. tact	*Intrare*	*Elemente de înmagazinare*	*Ieşire*
0	–	0 0 0 0 0 0	–
1	1	1 0 0 0 0 0	0
2	1	1 1 0 0 0 0	0
3	0	0 1 1 0 0 0	0
4	1	1 0 1 1 0 0	0
5	0	0 1 0 1 1 0	0
6	1	1 0 1 0 1 1	0
7	1	0 1 1 1 0 0	**1**
8	0	0 0 1 1 1 0	**0**
9	0	0 0 0 1 1 1	**0**
10	**0**	1 0 1 0 1 0	**1**
11	1	1 1 0 1 0 1	**0**

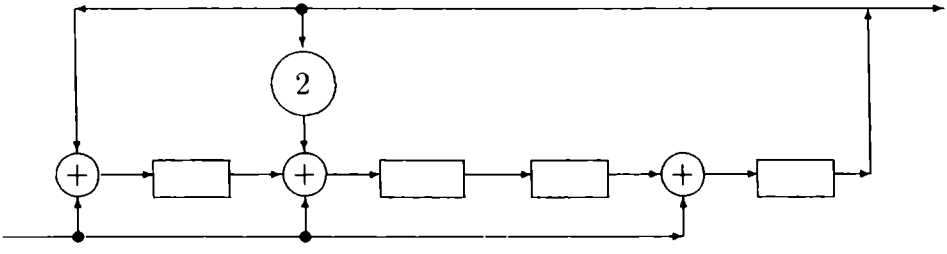
Vectorii $(1,1,0,1,0,1)$ - de pe ultima linie - şi $(0,1,0,0,1)$ (scris mai gros şi în ordine inversă pe ultima coloană) reprezintă restul $1+X+X^3+X^5$ respectiv câtul $X + X^4$ împărţirii lui $a(X)$ la $h(X)$.

D. Circuitul liniar care realizează înmulţirea cu polinomul $h(X) = h_0 + h_1X + \ldots + h_nX^n$, urmată de împărţirea la polinomul $g(X) = g_0 + g_1X + \ldots + g_nX^n$, este:



Când gradele celor două polinoame sunt diferite, se construieşte întâi circuitul liniar corespunzător polinomului de grad maxim, după care se completează cu circuitul corespunzător celuilalt polinom.

Exemplul 4.6 *Să luăm polinoamele* $h(X) = 1 + X + X^3$, $g(X) = 2 + X + X^4$ *cu coeficienţi în* Z_3. *Circuitul liniar care realizează înmulţirea cu* $h(X)$ *şi împărţirea la* $g(X)$ *este*



Deoarece calculele se fac în Z_3, $-2 = 1, -1 = 2$, *şi* $1^{-1} = 1$.

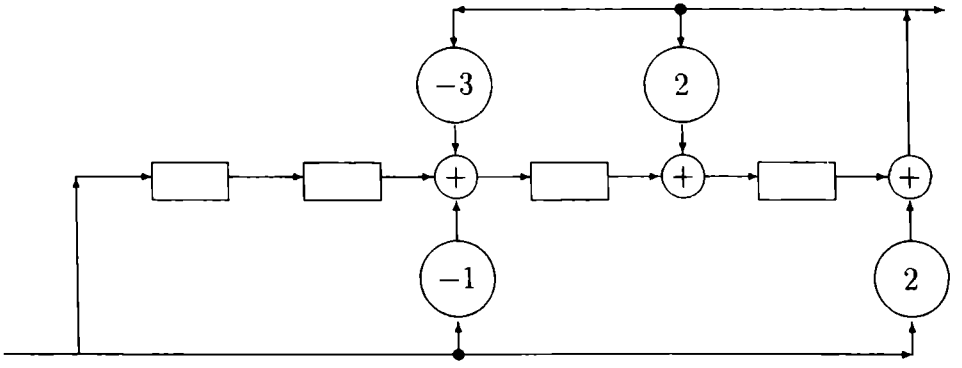
Pentru intrarea $a(X) = 1 + 2X + 2X^2$, *comportarea circuitului este următoarea:*

–	0	0	0	0	–
2	2	2	0	2	–
2	1	2	2	2	2
1	0	0	2	0	2

ceea ce corespunde câtului $2 + 2X$ *şi restului* $2X^2$.

De remarcat că primul coeficient (de grad maxim) al câtului este scos de circuit cu o întârziere de $grad(g(X)) - grad(h(X)) = 1$ *tact.*

Exemplul 4.7 *Un circuit liniar pentru înmulţirea cu* $h(X) = 1 - X^2 + 2X^4$ *şi împărţirea la* $g(X) = 3 - 2X + X^2$, *ambele polinoame în* $Z[X]$, *este:*



Fie intrarea $a(X) = -2 + X$; *comportarea circuitului este:*

−	0	0	0	0	−
1	1	0	−7	4	2
−2	−2	1	2	−7	0
0	0	−2	22	−12	−7
0	0	0	34	−2	−12

deci câtul este $2X^3 - 7X - 12$ *iar restul* $34 - 2X$.

De remarcat că circuitul este lăsat să funcţioneze $grad(h(X)) - grad(g(X)) = 2$ *tacţi în plus (prin introducerea de* 0*-uri), pentru golirea elementelor de înmagazinare necontrolate de împărţitor.*

4.2 Extensii Galois

Fie q un număr prim şi $h(X) = h_0 + h_1X + \ldots + h_nX^n \in Z_q[X]$. Vom considera algebra polinoamelor modulo $h(X)$.

Fiind dat un polinom oarecare din $Z_q[X]$, clasa sa de resturi modulo $h(X)$ se găseşte împărţind polinomul respectiv cu $h(X)$. Restul împărţirii specifică clasa de resturi respectivă.

Două polinoame cărora le corespunde acelaşi rest la împărţirea cu $h(X)$ sunt echivalente, intrând în aceeaşi clasă de resturi modulo $h(X)$.

Fiecare clasă de resturi modulo $h(X)$ se reprezintă prin polinomul (unic) de grad strict mai mic decât n, care aparţine acelei clase. Toţi aceşti reprezentanţi pot fi consideraţi ca polinoame de gradul $n - 1$, având eventual coeficienţi nuli.

Vom nota cu $\{f(X)\}$ clasa de resturi a polinomului $f(X)$ modulo $h(X)$. Deci $\{f(X)\}$ va corespunde unui vector (distinct) de lungime n, cu componente din Z_q (prin completare eventual cu zerouri).

Operaţiile în algebra claselor de resturi sunt:

$$\{a(X)\} + \{b(X)\} = \{a(X) + b(X)\}$$
$$c\{a(X)\} = \{ca(X)\},\ c \in Z_q$$
$$\{a(X)\}\{b(X)\} = \{a(X)b(X)\}$$

toate calculele fiind făcute modulo $h(X)$.

Evident, $\{0\} = \{h(X)\}$, unde $\{0\}$ este polinomul de grad $n-1$ cu toţi coeficienţii nuli (sau vectorul cu n componente zero). Vom nota $\{0\} = \mathbf{0}$.

Pentru un polinom $s(X) \in Z_q[X]$, clasa de resturi modulo $h(X)$ se obţine foarte uşor. Teorema împărţirii cu rest dă:

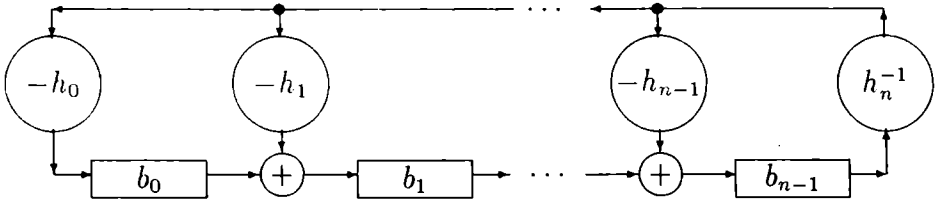
$s(X) = a(X)h(X) + r(X)$ unde $grad(r(X)) < grad(h(X)) = n$. Atunci:

$\{s(X)\} - \{r(X)\} = \{s(X) - r(X)\} = \{a(X)h(X)\} = \{a(X)\}\{h(X)\} = \mathbf{0}$, deci $\{s(X)\} = \{r(X)\}$.

Să reluăm circuitul liniar de împărţire definit în Figura 4.3, în care facem următoarele modificări (vezi Figura 4.4):

- Se neglijează intrarea şi ieşirea;
- La momentul iniţial elementele de înmagazinare conţin coeficienţii unui polinom $b(X) = b_0 + b_1X + \ldots + b_{n-1}X^{n-1}$.

Figura 4.4:



La următorul tact, în elementele de înmagazinare se obţin coeficienţii polinomului:

$b'(X) = -h_0h_n^{-1}b_{n-1} + (b_0 - h_1h_n^{-1}b_{n-1})X + (b_1 - h_2h_n^{-1}b_{n-1})X^2 + \ldots + (b_{n-2} - h_{n-1}h_n^{-1}b_{n-1})X^{n-1} = b_0X + b_1X^2 + \ldots + b_{n-1}X^n - b_{n-1}X^n - h_n^{-1}b_{n-1}(h_0 + h_1X + \ldots + h_{n-1}X^{n-1}) = Xb(X) - h_n^{-1}b_{n-1}h(X)$.

Dacă notăm $\alpha = \{X\}$ ($\{X\}$ este clasa de resturi modulo $h(X)$ a polinomului X), din relaţia de mai sus se obţine:

$b'(\alpha) = \{b'(X)\} = \{Xb(X)\} = \{X\}\{b(X)\} = \{X\}b(\{X\}) = \alpha b(\alpha)$.

Deci circuitul de împărţire cu polinomul $h(X)$ poate fi utilizat pentru înmulţirea lui $b(\alpha)$ cu α.

Teorema 4.1 *În algebra polinoamelor modulo* $h(X)$, $grad(h(X)) = n$, *avem:*

1. $h(\alpha) = \mathbf{0}$ *unde* $\alpha = \{X\}$;

2. h *este polinomul de grad minim care are pe* α *ca rădăcină.*

Demonstraţie:

1. Fie $h(X) = h_0 + h_1X + \ldots + h_nX_n \in Z_q[X]$, $h_n \neq 0$. Vom avea $h(\alpha) = h_0 + h_1\alpha + \ldots + h_n\alpha^n = h_0 + h_1\{X\} + \ldots + h_n\{X\}^n = h_0 + h_1\{X\} + \ldots + h_n\{X^n\} = \{h_0 + h_1X + \ldots + h_nX^n\} = \{h(X)\} = \mathbf{0}$.

2. Folosind acelaşi raţionament, se obţine pentru orice polinom $f(X)$ cu $grad(f(X)) < n$, că $f(\alpha) = \{f(X)\} \neq \mathbf{0}$, deoarece toate polinoamele neidentic nule de grad strict mai mic decât n aparţin la clase de resturi distincte. □

Să presupunem acum că $h(X) \in Z_q[X]$ este un polinom ireductibil de gradul n. Conform Teoremei 4.1, $\alpha = \{X\}$ este rădăcină a lui $h(X)$. Acest element α poate fi considerat fie ca un polinom de gradul $n-1$ (cu toţi coeficienţii nuli înafară de cel al lui X), fie ca un vector cu n componente $(0, 1, 0, \ldots, 0)$.

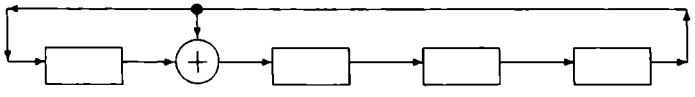
Vom nota cu $GF(q^n)$ mulţimea vectorilor din Z_q^n, sau – echivalent:
mulţimea polinoamelor de grad $n-1$ cu coeficienţi în Z_q, sau
– mulţimea claselor de resturi modulo polinomul $h(X) \in Z_q[X]$ de gradul n, ireductibil peste Z_q.
$GF(q^n)$ se numeşte *extensia Galois de grad* n a lui Z_q.

Spunem că $GF(q^n)$ se obţine prin adăugarea la Z_q a unei rădăcini a polinomului $h(X) \in Z_q[X]$ de grad n, ireductibil peste corpul de bază Z_q.

Observaţia 4.1 *Procedeul de trecere de la Z_q la $GF(q^n)$ este similar trecerii de la corpul numerelor reale $\mathcal{R}$ la cel complex $\mathcal{C}$, prin adăugarea rădăcinii $i = \{X\}$ în algebra claselor de resturi modulo polinomul $1+X^2$, ireductibil peste $\mathcal{R}$. Putem considera mulţimea $\mathcal{C}$ a numerelor complexe fie ca mulţimea polinoamelor de grad $2-1=1$, $a+bi$ cu $a,b \in \mathcal{R}$, fie ca mulţimea vectorilor (a,b) cu două componente reale.*

Dacă polinomul $h(X) \in Z_q[X]$ este ireductibil şi primitiv peste Z_q, puterile lui $\alpha = \{X\}$ vor genera toate elementele lui $GF(q^n) \setminus \{0\}$, unde $n = grad(h(X))$. Acest lucru se poate realiza cu circuitul liniar din Figura 4.4, introducând la momentul iniţial în elementele de înmagazinare coeficienţii polinomului $b(X) = 1$ (adică vectorul $(1,0,\ldots,0)$). Conform celor observate mai sus, la momentul următor se obţine $\{Xb(X)\} = \{X\} = \alpha$, după care urmează pe rând $\alpha^2, \alpha^3, \ldots$ Cum $h(X)$ a fost ales primitiv, el va genera toate elementele nenule din $GF(q^n)$.

Exemplul 4.8 *Fie polinomul $1+X+X^4$ ireductibil peste Z_2 şi $\alpha = \{X\}$ în algebra claselor de resturi modulo $1+X+X^4$. Circuitul liniar care va genera toate elementele nenule din $GF(2^4)$ este:*



α^0	=	1								1	0	0	0
α^1	=			α						0	1	0	0
α^2	=					α^2				0	0	1	0
α^3	=							α^3		0	0	0	1
α^4	=	1	+	α						1	1	0	0
α^5	=			α	+	α^2				0	1	1	0
α^6	=					α^2	+	α^3		0	0	1	1
α^7	=	1	+	α			+	α^3		1	1	0	1
α^8	=	1			+	α^2				1	0	1	0
α^9	=			α			+	α^3		0	1	0	1
α^{10}	=	1	+	α	+	α^2				1	1	1	0
α^{11}	=			α	+	α^2	+	α^3		0	1	1	1
α^{12}	=	1	+	α	+	α^2	+	α^3		1	1	1	1
α^{13}	=	1			+	α^2	+	α^3		1	0	1	1
α^{14}	=	1					+	α^3		1	0	0	1
α^{15}	=	1								1	0	0	0

unde, la momentul iniţial, în elementele de înmagazinare se află vectorul $(1,0,0,0)$. *Conform Teoremei 4.1,* α *este o rădăcină a polinomului* $1 + X + X^4$, *adică*

$$1 + \alpha + \alpha^4 = 0.$$

Lăsând circuitul să funcţioneze, în elementele de înmagazinare se obţin toate elementele lui $GF(2^4)$, *pe care le interpretăm fie ca vectori cu patru componente peste* Z_2, *fie ca polinoame de gradul* 3 *în* α *cu coeficienţi în* Z_2.

Vedem deci că polinomul $1 + X + X^4$ *este primitiv, puterile lui* α *epuizând toţi vectorii nenuli din* $GF(2^4) = \{\mathbf{0}, \alpha^0, \alpha, \ldots, \alpha^{14}\}$.

Exemplul 4.9 *Tabelul din exemplul precedent descrie operaţia de înmulţire din extensia Galois* $GF(2^4)$, *generată de rădăcina* α *a polinomului primitiv* $1 + X + X^4$. *Dacă s-ar alege* α *ca rădăcină a altui polinom primitiv - de exemplu* $1 + X^3 + X^4$, *se vor genera tot elementele lui* $GF(2^4)$, *dar în altă ordine. Lăsăm ca exerciţiu această generare.*

În $GF(2^4)$ *se poate defini şi operaţia de adunare, însumând modulo* 2 *componentele celor doi operanzi. Astfel, dacă reluăm* $GF(2^4)$ *generat în Exemplul 4.8 de rădăcina polinomului* $1 + X + X^4$, *tabela de adunare va fi:*

+	$\mathbf{0}$	α^0	α	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}
$\mathbf{0}$	$\mathbf{0}$	α^0	α	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}
α^0	α^0	$\mathbf{0}$	α^4	α^8	α^{14}	α	α^{10}	α^{13}	α^9	α^2	α^7	α^5	α^{12}	α^{11}	α^6
α	α	α^4	$\mathbf{0}$	α^5	α^9	α^0	α^2	α^{11}	α^{14}	α^{10}	α^3	α^8	α^6	α^{13}	α^{12}
α^2	α^2	α^8	α^5	$\mathbf{0}$	α^6	α^{10}	α	α^3	α^{12}	α^0	α^{11}	α^4	α^9	α^7	α^{14}
α^3	α^3	α^{14}	α^9	α^6	$\mathbf{0}$	α^7	α^{11}	α^2	α^4	α^{13}	α	α^{12}	α^9	α^{10}	α^8
α^4	α^4	α	α^0	α^{10}	α^7	$\mathbf{0}$	α^8	α^{12}	α^3	α^5	α^{14}	α^2	α^{13}	α^6	α^{11}
α^5	α^5	α^{10}	α^2	α	α^{11}	α^8	$\mathbf{0}$	α^9	α^{13}	α^4	α^6	α^0	α^3	α^{14}	α^7
α^6	α^6	α^{13}	α^{11}	α^3	α^2	α^{12}	α^9	$\mathbf{0}$	α^{10}	α^{14}	α^3	α^7	α	α^4	α^0
α^7	α^7	α^9	α^{14}	α^{12}	α^4	α^3	α^{13}	α^{10}	$\mathbf{0}$	α^{11}	α^0	α^6	α^8	α^2	α^9
α^8	α^8	α^2	α^{10}	α^0	α^{13}	α^5	α^4	α^{14}	α^{11}	$\mathbf{0}$	α^{12}	α	α^7	α^9	α^3
α^9	α^9	α^7	α^3	α^{11}	α	α^{14}	α^6	α^5	α^0	α^{12}	$\mathbf{0}$	α^{13}	α^2	α^8	α^{10}
α^{10}	α^{10}	α^5	α^8	α^4	α^{12}	α^2	α^0	α^7	α^6	α	α^{13}	$\mathbf{0}$	α^{14}	α^3	α^9
α^{11}	α^{11}	α^{12}	α^6	α^9	α^5	α^{13}	α^3	α	α^8	α^7	α^2	α^{14}	$\mathbf{0}$	α^0	α^4
α^{12}	α^{12}	α^{11}	α^{13}	α^7	α^{10}	α^6	α^{14}	α^4	α^2	α^9	α^8	α^3	α^0	$\mathbf{0}$	α
α^{13}	α^{13}	α^6	α^{12}	α^{14}	α^8	α^{11}	α^7	α^0	α^5	α^3	α^{10}	α^9	α^4	α	$\mathbf{0}$
α^{14}	α^{14}	α^3	α^7	α^{13}	α^0	α^9	α^{12}	α^8	α	α^6	α^4	α^{11}	α^{10}	α^5	α^2

Fie acum $a(X) = a_0 + a_1 X + \ldots + a_{k-1} X^{k-1} \in Z_q[X]$. Valoarea $a(\alpha)$ se obţine folosind circuitul de împărţire dat în Figura 4.3, cu următoarele observaţii:

- Se ignoră ieşirea;

- Coeficienţii lui $a(X)$ se introduc la intrare în ordinea descrescătoare a puterilor;

- Iniţial, elementele de înmagazinare conţin valorile $(0, 0, \ldots, 0)$.

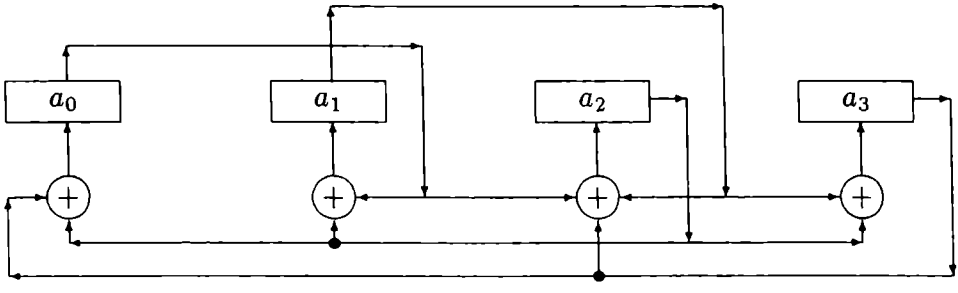
Pentru calculul unei valori de forma $a(\alpha)\alpha^j$ ($j \geq 0$ fixat), se pot construi circuite liniare care să realizeze direct acest produs.

Exemplul 4.10 *Fie $\alpha \in GF(2^4)$ element primitiv, rădăcină a ecuaţiei $1 + X + X^4 = 0$. Polinomul $a(X)$ va avea atunci gradul maxim 3. Fie $a(X) = a_0 + a_1X + a_2X^2 + a_3X^3$. Să construim un circuit liniar care să efectueze $a(\alpha)\alpha^5$.*

Folosind tabelul din Exemplul 4.8, avem:

$(a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3)\alpha^5 = a_0\alpha^5 + a_1\alpha^6 + a_2\alpha^7 + a_3\alpha^8 = a_0(\alpha + \alpha^2) + a_1(\alpha^2 + \alpha^3) + a_2(1 + \alpha + \alpha^2) + a_3(1 + \alpha^2) = (a_2 + a_3) + (a_0 + a_2)\alpha + (a_0 + a_1 + a_3)\alpha^2 + (a_1 + a_2)\alpha^3$

Pentru acest polinom se va construi circuitul liniar:



La momentul iniţial, în elementele de înmagazinare se găsesc coeficienţii lui $a(\alpha)$; după un tact, aici vor fi coeficienţii polinomului $a(\alpha)\alpha^5$.

4.3 Exerciţii

4.1 *Să se realizeze circuite liniare (cu ambele modalităţi de construcţie) care să efectueze înmulţiri cu polinoamele:*

$$h(X) = 1 + 2X + 3X^2 + 4X^3 \in Z[X],$$
$$h(X) = 1 + 2X + X^5 \in Z_7[X].$$

4.2 *Să se reprezinte sub formă de tabel comportamentul circuitelor realizate anterior, la înmulţirea cu polinoamele:*

$a(X) = 2 + 5X + X^3$, $a(X) = X - X^3 - 2X^4$, $a(X) = 0$.

4.3 *Se dă vectorul* $\mathbf{h} = (1, 0, 0, -1, 0, 1)$. *Să se realizeze circuitul de înmulţire cu polinomul corespunzător şi să se reprezinte comportamentul circuitului la înmulţirea cu polinoamele* $\mathbf{a} = (1)$ *şi* $\mathbf{a} = (1, 1, 1, 0, 0, 0, 1)$.

4.4 *Să se realizeze un circuit liniar care realizează produsul* $a(X)g(X) + b(X)h(X)$ *pentru polinoamele:*

$$g(X) = 1 + X + X^3, \quad h(X) = 1 - 5X - 2X^2 + 3X^3;$$
$$g(X) = -4X^3 + X^5 - 2X^8, \quad h(X) = X^2 + 2X^4 + X^6;$$
$$g(X) = X - X^3 + 3X^4, \quad h(X) = 2X^3 + X^5 + X^9.$$

4.5 *Să se efectueze împărţirea în* $Q[X]$ *prin polinomul* $h(X) = X^2 - 2X + 1$, *a polinoamelor*

$$a(X) = X^5 - 4X^3 - 2X + 5, \qquad a(X) = X + 7.$$

4.6 *Aceeaşi problemă pentru polinoamele* $h(X) = 2X - 1$ *şi respectiv* $a(X) = X^3 - 3$.

(1) *În* $Q[X]$;

(2) *În* $Z_3[X]$.

4.7 *Să realizeze un circuit care să efectueze (în* Q*) înmulţirea cu polinomul* $g(X) = 1 - 2X + X^3$ *şi împărţirea cu polinomul* $h(X) = g(X)$.

4.8 *Aceeaşi problemă pentru polinoamele:*

$g(X) = X - 2X^3 - X^4$, $h(X) = 2 + X + X^3$ *în* $Z_5[X]$;

$g(X) = -1 + X - X^6$, $h(X) = X^2 - 4X^3 - 2X^5 + 3X^7$ *în* $Z_{11}[X]$.

4.9 *Să se rezolve problema enunţată în Exemplul 4.9.*

4.10 *Să se genereze toate puterile lui* α, *rădăcină a polinomului*

(1) $1 + X^2 + X^5 \in Z_2[X]$

(2) $1 + X + X^3 \in Z_3[X]$.

4.11 *Folosind elementele din Exemplul 4.10, să se construiască circuite de generare pentru* $a(\alpha)\alpha$, $a(\alpha)\alpha^2$ *şi* $a(\alpha)\alpha^7$.

4.12 *Să se construiască circuite de generare în* $GF(2^5)$ *pentru* $a(\alpha)$ *şi* $a(\alpha)\alpha^{10}$, *unde* α *este rădăcină a polinomului* $1 + X^2 + X^5 \in Z_2[X]$.

Capitolul 5

Coduri ciclice

5.1 Relaţii de recurenţă liniară

Teorema 5.1 *Fie q un număr prim şi $f(X) \in Z_q[X]$, $grad(f(X)) = n$. În algebra polinoamelor modulo $f(X)$, fie I un ideal şi $g(X)$ polinomul de grad minim a cărui clasă de resturi aparţine lui I : $\{g(X)\} \in I$. Atunci:*

1. $\{s(X)\} \in I \iff g(X)|s(X)$;
2. $g(X)|f(X)$.

Demonstraţie: (1): Fie $\{s(X)\} \in I$. Folosind identitatea împărţirii, avem

$$s(X) = b(X)g(X) + r(X), \quad grad(r(X)) < grad(g(X)).$$

Trecând la clasele de resturi respective, se obţine $\{s(X)\} = \{b(X)\}\{g(X)\} + \{r(X)\}$. Cum $\{s(X)\} \in I$, rezultă $\{b(X)\}\{g(X)\} \in I$ (deoarece I este ideal), deci $\{r(X)\} \in I$, absurd, deoarece $g(X)$ este polinomul de grad minim a cărui clasă de resturi aparţine idealului I. Singura variantă rămâne $r(X) \equiv 0$, adică $s(X) = b(X)g(X)$.

Reciproca este imediată, pentru că din $s(X) = b(X)g(X)$ rezultă $\{s(X)\} = \{b(X)\}\{g(X)\} \in I$.

(2): Prin împărţirea polinomului $f(X)$ la $g(X)$ se obţine

$$f(X) = c(X)g(X) + r(X) \quad \text{unde} \quad grad(r(X)) < grad(g(X)).$$

Avem $\mathbf{0} = \{f(X)\} = \{c(X)\}\{g(X)\} + \{r(X)\}$, de unde rezultă $\{r(X)\} \in I$, absurd; deci $r(X) \equiv 0$. □

Teorema 5.2 *În algebra polinoamelor modulo $f(X)$, pentru orice ideal I există un polinom normat unic $g(X)$ de grad minim, cu $\{g(X)\} \in I$.*

Reciproc, pentru orice divizor normat al lui $f(X)$ există un ideal generat de acel divizor.

$g(X)$ se numeşte *generatorul* idealului I şi scriem $I = (\{g(X)\})$.

Demonstraţie: Fie $h(X)$ un polinom de grad minim cu $\{h(X)\} \in I$; vom nota $g(X) = h_n^{-1}h(X)$ unde h_n este coeficientul termenului de grad maxim din polinomul $h(X)$).

Să presupunem că $g(X)$, $g'(X)$ sunt două astfel de polinoame de grad minim, ale căror clase de resturi sunt în I. Conform Teoremei 5.1, $g(X)$ şi $g'(X)$ se divid unul pe altul şi - având acelaşi grad - diferă printr-o constantă. Polinoamele fiind normate, această constantă este 1. Deci $g(X) = g'(X)$.

Reciproc, fie $g(X)$ un divizor normat al polinomului $f(X)$ şi $(\{g(X)\})$ idealul generat de el. Un element al acestui ideal este $\{a(X)\} \in (\{g(X)\})$, deci are forma $\{a(X)\} = \{b(X)\}\{g(X)\} = \{b(X)g(X)\}$. □

Teorema 5.3 *Fie q un număr prim, $f(X) \in Z_q[X]$ un polinom normat cu $grad(f(X)) = n$ şi fie $f(X) = g(X)h(X)$. Dacă $grad(h(X)) = k$, atunci în algebra polinoamelor modulo $f(X)$, idealul $(\{g(X)\})$ are dimensiunea k.*

Demonstraţie: Să observăm că vectorii (asociaţi polinoamelor)

$$\{g(X)\},\ \{Xg(X)\},\ \ldots,\ \{X^{k-1}g(X)\}$$

sunt liniar independenţi. Într-adevăr, pentru orice k elemente $c_0, \ldots, c_{k-1} \in Z_q$, nu toate nule, avem $c_0\{g(X)\} + c_1\{Xg(X)\} + \ldots + c_{k-1}\{X^{k-1}\} = \{(c_0 + c_1X + \ldots + c_{k-1}X^{k-1})g(X)\} \neq 0$
deoarece s-a obţinut un polinom de grad cel mult $n-k+k-1 = n-1 < n$.

În acelaşi timp, pentru orice $\{s(X)\} \in (\{g(X)\})$ avem
$\{s(X)\} = \{a(X)g(X)\} = \{(a_0 + a_1X + \ldots + a_{k-1}X^{k-1})g(X)\} = a_0\{g(X)\} + a_1\{Xg(X)\} + \ldots + a_{k-1}\{X^{k-1}g(X)\}$, unde unii din coeficienţii $a_0, a_1, \ldots, a_{k-1}$ pot fi nuli.

Deci vectorii de sus formează o bază a subspaţiului liniar $(\{g(X)\})$, care are deci dimensiunea k. □

Teorema 5.4 *Fie $f(X) \in Z_q[X]$ un polinom normat care se poate scrie ca produs de două polinoame: $f(X) = g(X)h(X)$.*

În algebra polinoamelor modulo $f(X)$,
$\{a(X)\} \in (\{g(X)\}) \Longleftrightarrow \{a(X)\}$ *este în spaţiul nul al idealului* $(\{h(X)\})$.

Demonstraţie:
"$\Longrightarrow$": Fie $\{a(X)\} \in (\{g(X)\})$, deci $a(X) = v(X)g(X)$.

Fie de asemenea $\{b(X)\} \in (\{h(X)\})$, deci $b(X) = w(X)h(X)$. Vom avea $a(X)b(X) = v(x)w(X)g(X)h(X) = v(X)w(X)f(X)$, deci

$$\{a(X)\}\{b(X)\} = \{v(X)w(X)\}\{f(X)\} = \mathbf{0}.$$

" $\Longleftarrow$": Să considerăm $\{a(X)\}$ în spaţiul nul al idealului $(\{h(X)\})$. Atunci $\{a(X)\}\{h(X)\} = \mathbf{0}$, adică $a(X)h(X) = c(X)f(X) = c(X)g(X)h(X)$ de unde rezultă $a(X) = c(X)g(X)$. Deci $\{a(X)\} \in (\{g(X)\})$. □

Definiţia 5.1 *Se numeşte relaţie de recurenţă liniară egalitatea*

$$\sum_{j=0}^{k} h_j a_{i+j} = 0, \tag{1}$$

unde $i \geq 0$, $h_k = 1$, $h_0 \neq 0$, $h_j \in Z_q$, $a_{i+j} \in Z_q$.

Relaţiile de recurenţă liniară se pot scrie şi

$$a_{i+k} = -\sum_{j=0}^{k-1} h_j a_{i+j}, \quad i = 0, 1, \ldots$$

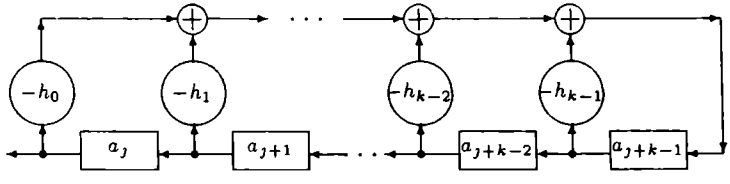
O soluţie a unei astfel de relaţii de recurenţă liniară este orice succesiune infinită de forma $a_0, a_1, \ldots, a_p, \ldots$ care verifică relaţia dată, cu $h_0, h_1, \ldots, h_k$ fixate, $h_0 \neq 0$, $h_k = 1$. Relaţia va determina succesiv pe a_k din $a_0, \ldots, a_{k-1}$, apoi pe a_{k+1} din $a_1, \ldots, a_k$ ş.a.m.d.

Altfel spus, "condiţiile iniţiale" $a_0, a_1, \ldots, a_{k-1}$ determină o soluţie a relaţiei de recurenţă liniară.

Observaţia 5.1

- *Orice combinaţie liniară de soluţii ale unei relaţii de recurenţă liniară este tot o soluţie.*
- *Soluţiile pentru care condiţiile iniţiale sunt respectiv*
  $(1, 0, \ldots, 0), (0, 1, \ldots, 0), \ldots, (0, 0, \ldots, 1)$
  determină orice altă soluţie. Deci, spaţiul soluţiilor relaţiei de recurenţă (1) *are dimensiunea cel mult* k.

Fiind date condiţiile iniţiale (arbitrare) $a_0, a_1, \ldots, a_{k-1}$, soluţia relaţiei de recurenţă liniară (1) corespunzătoare lor se poate obţine folosind circuitul liniar



în care, la momentul inițial, în elementele de înmagazinare se găsesc $a_0, a_1, \ldots, a_{k-1}$.

Să considerăm acum polinomul (fixat) $h(X) \in Z_q[X]$, $h(X) = h_0 + h_1X + \ldots + h_kX^k$, $h_0 \neq 0, h_k = 1$, și fie n cel mai mic număr natural pentru care $X^n - 1$ se divide cu $h(X)$. Notăm

$$g(X) = \frac{X^n - 1}{h(X)}. \tag{2}$$

Pe baza acestor date construim relația de recurență liniară

$$a_{i+k} = -\sum_{j=0}^{k-1} h_j a_{i+j},\ i \geq 0 \tag{3}$$

Propoziția 5.1 *Fie $h(X) \in Z_q[X]$ un polinom normat Atunci există un număr natural n astfel ca $h(X)|X^n - 1$.*

Demonstrație: Afirmația se bazează pe "*principiul cutiei*": fie n_1, n_2 ($n_1 > n_2$) cele mai mici numere cu proprietatea că $X^{n_1} - 1$ și $X^{n_2} - 1$ dau același rest la împărțirea cu $h(X)$. Atunci $X^{n_2}(X^{n_1-n_2} - 1)$ se divide cu $h(X)$. Deoarece $h(X)$ are termenul liber 1, el va divide $X^{n_1-n_2} - 1$, și se ia $n = n_1 - n_2$.

Evident, prin construcție n este minim cu această proprietate.

Teorema 5.5 (*a*) *Soluțiile relației de recurență* (3) *sunt periodice, de perioadă n.*

(*b*) *Mulțimea formată din prima perioadă a fiecărei soluții, scrisă ca polinom $a(X) = a_0X^{n-1} + \ldots + a_{n-2}X + a_{n-1}$, constituie idealul $(\{g(X)\})$ în algebra polinoamelor modulo $X^n - 1$.*

Demonstrație: (*a*): Vom arăta că oricărui element $\{a(X)\} \in (\{g(X)\})$ cu $a(X) = a_0X^{n-1} + \ldots + a_{n-2}X + a_{n-1}$ îi corespunde o soluție periodică

$$a_0, a_1, \ldots, a_{n-1}, a_0, a_1, \ldots, a_{n-1}, a_0, \ldots$$

a relației de recurență (3). Evident, nu este obligatoriu ca toți coeficienții a_i să fie nenuli, gradul real al polinomului $f(X)$ putând fi chiar zero.

Fie $\{a(X)\}\{h(X)\} = \{c(X)\}$. Vom nota $c(X) = c_0 + c_1X + \ldots + c_{n-1}X^{n-1}$. Avem:

- pentru $k \leq p \leq n-1$: $c_p = h_0a_{n-1-p} + h_1a_{n-1-p+1} + \ldots + h_ka_{n-1-p+k}$; (4)

- pentru $0 \leq p < k$: $c_p = h_0a_{n-1-p} + h_1a_{n-1-p+1} + \ldots + h_pa_{n-1} + h_{p+1}a_0 + \ldots + h_ka_{k-p-1}$. (5)

Conform Teoremei 5.4, dacă $\{a(X)\} \in (\{g(X)\})$ atunci $\{a(X)\}\{h(X)\} = \mathbf{0}$, adică $c_p = 0$ $(0 \leq p \leq n-1)$.

Considerând $(a_i)_i$ ca o succesiune periodică, vom avea evident $a_{i+n} = a_i$, $i = 0, 1, \ldots$. Ţinând cont de aceasta, după introducerea lui $c_p = 0$ în (4) şi (5) se obţine (3).

(b): Idealul $(\{g(X)\})$ are dimensiunea $k = grad(h(X))$ (Teorema 5.3) şi fiecare element al acestui ideal (polinom sau vector - după cum este scris) va da (conform cu (a)) o soluţie a relaţiei de recurenţă liniară (3). Dar spaţiul soluţiilor relaţiei (3) are dimensiunea cel mult k. Deci idealul $(\{g(X)\}$ va da toate soluţiile relaţiei de recurenţă liniară (3). Mai trebuie arătat că perioada acestor soluţii este chiar n.

Din cele de până acum a rezultat că perioada este cel mult n. Vom arăta că soluţia corespunzătoare clasei de resturi $\{g(X)\}$ are perioada n.

Să presupunem prin absurd că soluţia corespunzătoare lui $\{g(X)\}$ are perioada $m < n$. Dar n este divizibil cu m şi deci coeficienţii polinomului $g(X)$ - considerat ca fiind de grad $n-1$ (prin completare cu coeficienţi nuli) formează $\frac{n}{m}$ blocuri care se repetă.

Deci $g(X) = q(X)(1 + X^m + X^{2m} + \ldots + X^{n-m})$ unde $q(X)$ este un polinom de gradul $m-1$. Relaţia se poate scrie şi $g(X) = q(X)\frac{X^n - 1}{X^m - 1}$.

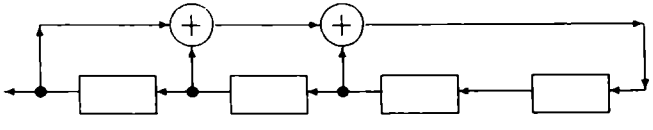
Vom avea acum $(X^n - 1)(X^m - 1) = g(X)h(X)(X^m - 1) = q(X)h(X)(X^n - 1)$ adică $X^m - 1 = q(X)h(X)$, ceea ce contrazice condiţia că n este cel mai mic număr care verifică Propoziţia 5.1.

Deci $m = n$. □

Exemplul 5.1 *Să considerăm polinomul* $h(X) = 1 + X + X^2 + X^4 \in Z_2[X]$. *Relaţia de recurenţă liniară asociată este*

$$a_{i+4} = a_{i+2} + a_{i+1} + a_i \quad (i \geq 0)$$

Circuitul liniar corespunzător are forma



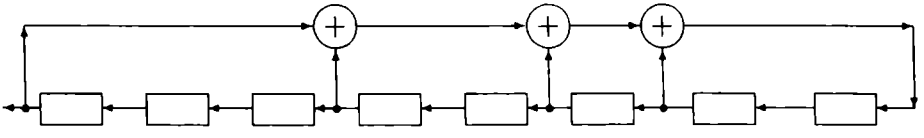
Cel mai mic n *pentru care* $X^n - 1$ *se divide cu* $h(X)$ *este* $n = 7$. *Cum* $g(X) = \frac{X^7 - 1}{h(X)} = X^3 + X + 1$, *rezultă că circuitul va genera cuvintele idealului* $(\{g(X)\})$. *Fiecare cuvânt este caracterizat de cele patru valori binare iniţiale din elementele de înmagazinare. Vor fi deci* $2^4 = 16$ *cuvinte de lungime* 7. *Ele corespund tuturor polinoamelor de grad maxim* 6 *din* $Z_2[X]$, *care se divid cu* $g(X)$.

Ieşire				
–	1	0	1	1
1	0	1	1	0
0	1	1	0	0
1	1	0	0	0
1	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	1	0	1	1

De exemplu, pentru valorile iniţiale $(1,0,1,1)$, *funcţionarea circuitului timp de şapte tacţi este dată în tabelul de mai sus.*

În elementele de înmagazinare se regăsesc valorile iniţiale, iar la ieşire s-a obţinut polinomul $f(X) = X^6 + X^4 + X^3 = X^3 g(X)$.

Exemplul 5.2 *Circuitul liniar corespunzător polinomului* $h(X) = 1 + X^3 + X^5 + X^6 + X^8 \in Z_2[X]$ *este*



El dă soluţiile relaţiei de recurenţă liniară

$$a_i + a_{i+3} + a_{i+5} + a_{i+6} + a_{i+8} = 0,$$

care formează idealul generat de polinomul $g(X) = \dfrac{X^{255} - 1}{h(X)}$ *ideal compus din* $2^8 = 256$ *cuvinte de lungime* 255.

5.2 Definirea codurilor ciclice

Fie n ($n \geq 2$) un număr natural. Să notăm cu A_n algebra polinoamelor din $Z_q[X]$, modulo $X^n - 1$. După cum s-a convenit, identificăm cuvântul $a_0 a_1 \ldots a_{n-1}$ cu vectorul $(a_0, a_1, \ldots, a_{n-1})$ şi cu polinomul $a(X) = a_0 + a_1 X + \ldots + a_{n-1} X^{n-1} \in Z_q[X]$.

În cadrul dualismului *vector - polinom* vom face o deosebire: anularea produsului a două polinoame nu înseamnă ortogonalitatea vectorilor corespunzători. Pentru această situaţie se foloseşte următoarea propoziţie:

Propoziţia 5.2 *Produsul a două polinoame este zero dacă şi numai dacă toate produsele scalare dintre vectorul unui polinom şi permutările ciclice ale vectorului celuilalt polinom, sunt zero.*

Demonstraţie: Fie $a(X), b(X) \in Z_q[X]$, $a(X) = \sum_{i=0}^{n-1} a_i X^i$, $b(X) = \sum_{i=0}^{n-1} b_i X^i$. Atunci $\{c(X)\} = \{a(X)\}\{b(X)\} = \{c_0 + c_1 X + \ldots + c_{n-1} X^{n-1}\}$

unde $c_j = \sum_{i=0}^{j} a_i b_{j-i} + \sum_{i=j+1}^{n-1} a_i b_{j+n-i} =$

$= (a_0, a_1, \ldots, a_{n-1}) \cdot (b_j, \ldots, b_0, b_{n-1}, \ldots, b_{j+1})^T,$

şi Propoziţia rezultă din faptul că $c_j = 0 \ \forall j$. □

Definiţia 5.2 *Un subspaţiu liniar $V_n \subseteq A_n$ se numeşte ciclic dacă*

$$(a_0, a_1, \ldots, a_{n-1}) \in V_n \quad \Longrightarrow \quad (a_{n-1}, a_0, \ldots, a_{n-2}) \in V_n.$$

Teorema 5.6 *Un subspaţiu liniar $V_n \subseteq A_n$ este ciclic dacă şi numai dacă este ideal.*

Demonstraţie: Înmulţirea cu clasa de resturi $\{X\}$ înseamnă de fapt permutarea ciclică a componentelor cu o unitate spre dreapta, pentru că:

$$\{X\}\{a_0 + a_1 X + \ldots + a_{n-1} X^{n-1}\} = \{a_{n-1} + a_0 X + \ldots + a_{n-2} X^{n-1}\}.$$

"$\Longleftarrow$": Dacă $V_n \subseteq A_n$ este ideal, atunci pentru orice $\mathbf{v} \in V_n$ avem $\mathbf{v}' = \{X\}\mathbf{v} \in V_n$ (s-a notat tot cu $\mathbf{v}$ clasa de resturi modulo $X^n - 1$ corespunzătoare polinomului ai cărui coeficienţi sunt componentele vectorului $\mathbf{v}$). Deci V_n este ciclic.

"$\Longrightarrow$": Presupunem că subspaţiul liniar $V_n \subseteq A_n$ este ciclic şi fie $\mathbf{v} \in V_n$. Atunci, din $\{X\}\mathbf{v} \in V_n$ rezultă $\{X^j\}\mathbf{v} \in V_n$, $\forall j = 1, \ldots, n-1$. Deci $\{c_0 + c_1 X + \ldots + c_{n-1} X^{n-1}\}\mathbf{v} \in V_n$, adică V_n este ideal în A_n. □

Definiţia 5.3 *Se numeşte cod ciclic un ideal propriu $I \subset A_n$.*

Codurile ciclice au fost introduse de Prange (1957), care a evidenţiat bogăţia de informaţie rezultată din structura lor algebrică.

5.3 Generarea codurilor ciclice

Pentru a construi un cod ciclic se foloseşte structura idealelor în algebra polinoamelor modulo $X^n - 1$. Fie I un ideal proprie în A_n şi $g(X)$ polinomul normat de grad minim cu $\{g(X)\} \in I$. Atunci (Teorema 5.1) $\{a(X)\} \in I$ dacă şi numai dacă $g(X)|a(X)$.

De asemenea, $g(X)|X^n - 1$.

Oricărui divizor $g(X)$ al lui $X^n - 1$ ($grad(g(X)) < n$) îi corespunde un ideal propriu în A_n, generat de $g(X)$.

Pentru a da un cod ciclic este suficient să dăm generatorul său $g(X)$, divizor al lui $X^n - 1$. Fie $g(X) = g_0 + g_1X + \ldots + g_{n-k}X^{n-k}$ ($k < n$). O bază a idealului ($\{g(X)\}$) se poate alege (Teorema 5.3)

$$\{g(X)\},\ \{Xg(X)\},\ldots,\{X^{k-1}g(X)\}.$$

Matricea generatoare corespunzătoare codului ciclic va fi

$$G_{k,n} = \begin{pmatrix} g_0 & g_1 & \ldots & g_{n-k} & 0 & 0 & & 0 \\ 0 & g_0 & \ldots & g_{n-k-1} & g_{n-k} & 0 & & 0 \\ & & & & \vdots & & & \\ 0 & 0 & \ldots & 0 & g_0 & g_1 & \ldots & g_{n-k} \end{pmatrix}$$

Rezultă că un cod ciclic poate fi organizat ca un (n, k) - cod liniar, unde n este gradul polinomului $X^n - 1$ iar k este gradul polinomului $h(X) = \dfrac{X^n - 1}{g(X)}$.

Exemplul 5.3 *Codul cu repetiţie este un cod ciclic al cărui polinom generator este* $g(X) = 1 + X + X^2 + \ldots + X^{n-1}$.

Exemplul 5.4 *Să considerăm codul ciclic binar de lungime 7, cu polinomul generator* $g(X) = 1 + X^2 + X^3$ *(vezi şi Exemplul 5.1). El are matricea generatoare*

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

Exemplul 5.5 *Să considerăm* $n = 6$ *şi* $q = 3$. *Deoarece* $X^6 - 1 = (X + 1)^3(X + 2)^3$, *există 14 polinoame în* $Z_3[X]$ *care divid* $X^6 - 1$, *deci sunt posibile 14 coduri ciclice ternare de lungime 6.*

Unul din ele este de exemplu codul ciclic generat de polinomul $g(X) = (X + 1)(X + 2) = 2 + X^2$, *având matricea generatoare*

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

El este deci un $(6,4)$ *- cod ternar.*

Idealul generat de $\{g(X)\}$ este spaţiul nul al idealului $(\{h(X)\})$ unde $h(X) = \dfrac{X^n - 1}{g(X)}$. Acest ideal $(\{h(X)\})$ se construieşte luând ca bază polinoamele

$$\{h(X)\},\ \{Xh(X)\} \ldots, \{X^{n-k-1}h(X)\}.$$

Ţinând cont de Propoziţia 5.2, putem construi matricea de control H a codului ciclic $(\{g(X)\})$ luând ca linii ale matricii cele $n-k$ polinoame de sus, cu ordinea componentelor inversată.

Exemplul 5.6 *Să reluăm codul din Exemplul 5.4. Deoarece* $X^7 - 1 = (1+X)(1+X+X^3)(1+X^2+X^3)$, *pentru acest cod, avem*

$h(X) = (1+X)(1+X+X^3) = 1+X^2+X^3+X^4$.

Matricea generatoare a codului $(\{h(X)\})$ *este*

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix},$$

deci matricea de control asociată codului din Exemplul 5.4 va fi

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

Deoarece coloanele sale sunt nenule şi distincte două câte două, codul astfel construit este $(7,4)$ *- codul Hamming binar.*

Exemplul 5.7 $(6,4)$ *- codul ternar din Exemplul 5.5 are drept cod dual* $(6,2)$ *- codul generat de polinomul* $h(X) = (1+X)^2(2+X)^2 = 1+X^2+X^4$. *Deci matricea sa de control va fi*

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

Pentru a obţine un cod ciclic sistematic, este convenabil să alegem o altă bază pentru idealul $(\{g(X)\})$. Să observăm că pentru $i = n-k, n-k+1, \ldots, n-1$, putem scrie

$$X^i = q_i(X)g(X) + r_i(X) \text{ cu } grad(r_i(X)) < grad(g(X)) = n-k.$$

Deci $\{X^i - r_i(X)\} \in (\{g(X)\}), \quad i = n-k, n-k+1, \ldots, n-1.$

Această mulţime de vectori este liniar independentă şi conduce la o matrice generatoare de forma

$$G = [-R\ I]$$

unde linia j din matricea R este vectorul coeficienţilor lui $r_j(X)$ pentru $j = n-k, \ldots, n-1$. Codul obţinut este deci sistematic, iar matricea sa de control se scrie imediat:

$$H = [I\ R^T].$$

De remarcat că matricea H^T are pe linii componentele resturilor $r_i(X)$ pentru $i = 0, 1, \ldots, n-1$.

Exemplul 5.8 *Să luăm din nou* $X^7 - 1 = g(X)h(X)$ *peste* Z_2, *unde* $g(X) = 1 + X^2 + X^3$, $h(X) = 1 + X^2 + X^3 + X^4$ *(deci* $n = 7$, $k = 4$*).* *Avem*

$$\begin{aligned}
X^0 &= 0g(X) + 1 \\
X^1 &= 0g(X) + X \\
X^2 &= 0g(X) + X^2 \\
X^3 &= g(X) + 1 + X^2 \\
X^4 &= (1+X)g(X) + 1 + X + X^2 \\
X^5 &= (1+X+X^2)g(X) + 1 + X \\
X^6 &= (X+X^2+X^3)g(X) + X + X^2
\end{aligned}$$

Baza corespunzătoare idealului $(\{g(X)\})$ *este*

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

care conduce la matricea generatoare a unui cod sistematic:

$$G_{4,7} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = [-R_{4,3}\ I_4].$$

Matricea de control corespunzătoare se scrie simplu:

$$H_{3,7} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} = [I_3\ R_{3,4}^T].$$

5.4 Generarea automată a codurilor ciclice

Codurile ciclice sunt coduri a căror implementare este mult facilitată de circuitele liniare. Cu ajutorul acestora se poate realiza automat codificarea, calculul sindromului, detectarea şi corectarea erorilor.

În construcţia practică vom distinge două cazuri:

5.4.1 Circuit cu k elemente de înmagazinare

Este o metodă de generare a codurilor ciclice, avantajoasă dacă sunt mai puţine simboluri de informaţie decât de control : $k < n - k$.

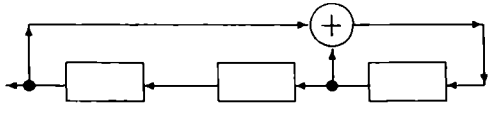
Fie codul ciclic $(\{g(X)\}) \subset A_n$ şi $h(X) = \dfrac{X^n - 1}{g(X)} = h_0 + h_1 X + \ldots + h_k X^k$ cu $h_0 \neq 0$, $h_k = 1$. Conform Teoremei 5.5, idealul $(\{g(X)\})$ este generat de circuitul liniar care construieşte soluţiile relaţiei de recurenţă liniară

$$\sum_{j=0}^{k} h_j a_{i+j} = 0, \quad i = 0, 1, \ldots$$

Mesajul de informaţie care trebuie codificat - conţinând k simboluri pe fiecare bloc - se introduce la momentul iniţial în elementele de înmagazinare ale circuitului, sub formă de "condiţii iniţiale". Lăsând circuitul să funcţioneze, obţinem după n momente cuvântul - cod corespunzător, aparţinând idealului $(\{g(X)\})$ şi având pe primele k poziţii elementele de informaţie.

Exemplul 5.9 *Circuitul liniar construit în Exemplul 5.1 este un circuit de codificare pentru codul generat de polinomul* $g(X) = 1 + X + X^3$. *Exemplul descrie şi un mod de funcţionare pentru cuvântul de informaţie* 1011.

Dacă s-ar lua drept polinom generator $1 + X^2 + X^3 + X^4$, *idealul generat de el este dat de circuitul liniar*



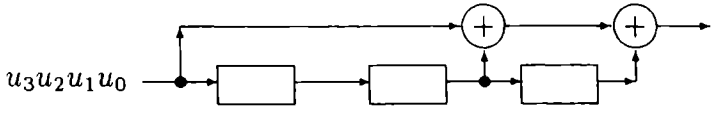
deoarece $h(X) = \dfrac{X^7 - 1}{1 + X^2 + X^3 + X^4} = 1 + X^2 + X^3$.

5.4.2 Circuit cu $n-k$ elemente de înmagazinare

Este o strategie avantajos de utilizat în cazul $n-k<k$.

Dacă interpretăm mesajul de informaţie ca un polinom de gradul $k-1$, atunci codificarea se poate face utilizând un circuit de înmulţire cu polinomul generator $g(X)$ (vezi Capitolul 4). La decodificare se recapătă mesajul iniţial (dacă nu au apărut erori) folosind un circuit de împărţire cu $g(X)$.

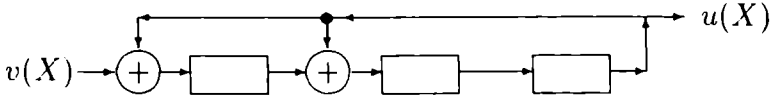
Exemplul 5.10 *Codul ciclic de polinom generator* $g(X)=1+X+X^3$ *poate codifica mesajele de informaţie folosind circuitul liniar:*



Astfel, mesajul de informaţie $\mathbf{u}=1001$ *se codifică în* $\mathbf{v}=1100101$, *conform calculelor:*

$$v(X)=u(X)g(X)=(1+X^3)(1+X+X^3)=1+X+X^4+X^6$$

Pentru decodificare se foloseşte circuitul

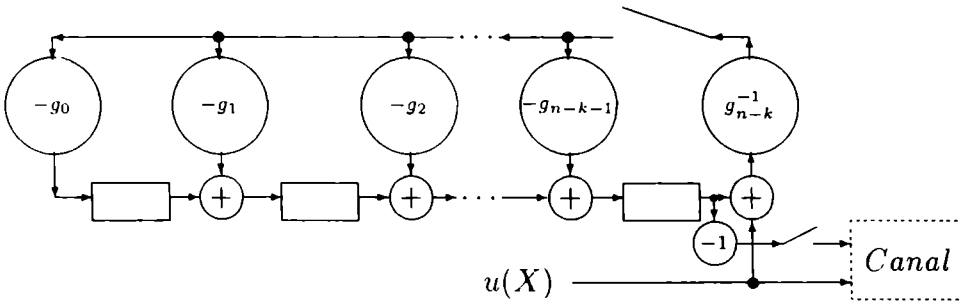


Dezavantajul unei asemenea metode îl constituie faptul că, nefiind un cod sistematic, prin codificare poziţiile de informaţie se pierd. Pentru a obţine un cod sistematic, procedăm în felul următor: mesajul de informaţie este considerat un polinom $u_0(X)$ de gradul $n-1$ având poziţiile de informaţie drept coeficienţii lui $X^{n-k},\ldots,X^{n-1}$, iar restul coeficienţilor sunt 0. Atunci avem $u_0(X)=q(X)g(X)+r(X)$ cu $grad(r(X))<grad(g(X))=n-k$, de unde

$$\{u_0(X)-r(X)\}\in(\{g(X)\}).$$

$\{u_0(X)-r(X)\}$ reprezintă un cuvânt - cod în care coeficienţii lui X^{n-k}, $\ldots,X^{n-1}$ sunt poziţiile de informaţie nealterate, iar coeficienţii lui $r(X)$ cu semnul schimbat (deci coeficienţii lui $X^0,X^1,\ldots,X^{n-k-1}$) sunt caracterele de control.

Circuitul care realizează această codificare este următorul:

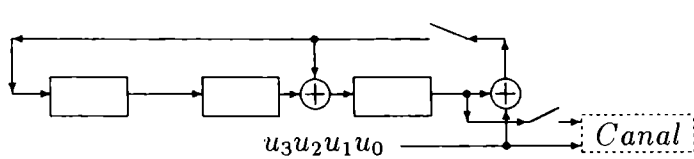


El funcţionează astfel:

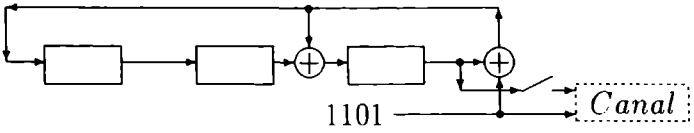
- Iniţial, cele $n-k$ elemente de înmagazinare conţin 0, întrerupătorul de ieşire este decuplat iar cel de feedback - cuplat.
- Se introduc cele k elemente ale mesajului de informaţie atât în circuit cât şi în canalul de comunicaţie. După k momente, în elementele de înmagazinare avem coeficienţii restului $r(X)$.
- Se decuplează feedbackul şi se cuplează circuitul la canalul de comunicaţie.
- Coeficienţii restului - cu semn schimbat - se transmit în canal imediat după poziţiile de informaţie.

Deoarece provine din circuitul de împărţire cu $g(X)$, acelaşi circuit poate fi folosit şi pentru detectarea erorilor la recepţie. Pentru aceasta, după decuplarea întrerupătorului de ieşire şi cuplarea celui de feedback, se introduce în circuit cuvântul recepţionat. Dacă la momentul n restul nu este nul (cel puţin un element de înmagazinare conţine un element nenul) înseamnă că a apărut cel puţin o eroare în transmisie.

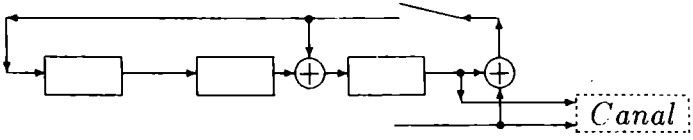
Exemplul 5.11 *Codul ciclic binar de lungime* 7, *generat de* $g(X) = 1 + X^2 + X^3$, *poate fi construit folosind circuitul liniar:*



Să presupunem că vrem să transmitem mesajul de informaţie 1011. *Pentru codificare, primii patru tacţi vor lucra cu următorul circuit liniar:*



Următorii trei taçti vor lucra cu acelaşi circuit liniar, puţin modificat:

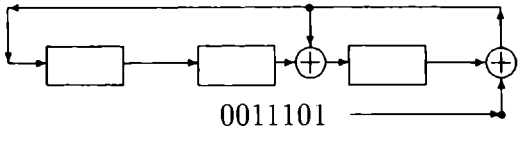


Funcţionarea circuitului este descrisă de următorul tabel:

Nr. crt.	*Intrare*				*Ieşire*
1	1	1	0	1	1
2	0	1	1	1	0
3	1	0	1	1	1
4	1	0	0	1	1
5	–	0	0	0	1
6	–	0	0	0	0
7	–	0	0	0	0

Deci prin canal este transmis cuvântul - cod 1011100 *(corespunzător polinomului* $a(X) = X^6 + X^4 + X^3 + X^2$*). Se observă că primii patru biţi sunt biţii de informaţie.*

La recepţie, cuvântul 1011100 *este introdus în circuitul liniar (în care se ignoră legătura cu canalul de comunicaţie) :*



După ce funcţionează 7 *taçti, în elementele de înmagazinare ale circuitului rămâne* 000*, deci cuvântul a fost transmis fără erori. Pentru decodificare – codul fiind sistematic – se păstrează primii patru biţi, adică* 1011*.*

5.5 Exerciţii

5.1 *Să se determine toate codurile ciclice de lungime* $n = 5$
(*a*) *peste* Z_2*;*
(*b*) *peste* Z_3*.*

5.2 *Aceeaşi problemă pentru codurile de lungime* $n = 6$.

5.3 *Construiţi o bază pentru cel mai mic cod ciclic de lungime* n *care conţine cuvântul - cod* $\mathbf{v}$*:*

(a) $\mathbf{v} = 1101000$, $n = 7$;
(b) $\mathbf{v} = 010101$, $n = 6$;
(c) $\mathbf{v} = 11011000$, $n = 8$.

5.4 *Pentru fiecare din cuvintele de mai jos, găsiţi polinomul generator al celui mai mic cod ciclic care conţine cuvântul respectiv:*

010010, 01100110, 0101100,
001000101110000, 000001001000000, 010111010000000.

5.5 *Să se afle polinomul generator al codului ciclic* C*, ştiind o bază* S *a sa:*

(a) $S = \{010, 011, 111\}$; (b) $S = \{1010, 0101, 1111\}$;
(c) $S = \{0101, 1010, 1100\}$; (d) $S = \{1000, 0100, 0010, 0001\}$.

5.6 *Într-o codificare sistematică, codificaţi mesajele de informaţie date de polinoamele:* $1 + X^2$, X, $X + X^2 + X^3$ *în cuvânte - cod al unui cod ciclic binar de lungime* 7, *cu polinomul generator* $g(X) = 1 + X^2 + X^3$.

5.7 *Fie codul definit mai sus.*

Fiind date cuvintele - cod $X^2 + X^4 + X^5$, $1 + X + X^2 + X^4$, $X^2 + X^3 + X^4 + X^6$, *găsiţi mesajele de informaţie corespunzătoare.*

5.8 *Construiţi circuite liniare pentru codificare/decodificare ale codului ciclic binar de lungime* 7 *generat de polinomul* $g(X) = (1 + X^2 + X^3)(1 + X)$.

5.9 *Care este lungimea minimă a unui cod ciclic binar de polinom generator* $g(X) = 1 + X^2 + X^3 + X^4$ *?*

Construiţi circuite liniare pentru codificarea şi decodificarea sa.

5.10 *Construiţi un circuit liniar pentru codificarea* $(15, 11)$ *- codului Hamming binar.*

5.11 *Construiţi circuite liniare pentru codificarea codului Golay binar (generat de* $g(X) = 1 + X + X^5 + X^6 + X^7 + X^9 + X^{11}$*) şi ternar (generat de* $g(X) = 2 + X^2 + 2X^3 + X^4 + X^5$*).*

5.12 *Construiţi matricea de control a codului ciclic binar de lungime n şi polinom generator $g(X)$:*

$n = 6$, $g(X) = 1 + X^2$;
$n = 8$, $g(X) = 1 + X^2$;
$n = 9$, $g(X) = 1 + X^3 + X^6$;
$n = 15$, $g(X) = 1 + X + X^4$;
$n = 15$, $g(X) = 1 + X^4 + X^6 + X^7 + X^8$;
$n = 23$, $g(X) = 1 + X + X^5 + X^7 + X^9 + X^{11}$.

5.13 *Sunt ciclice codurile liniare binare definite de matricile generatoare definite mai jos ?*

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

5.14 *Arătaţi că dacă polinomul generator al unui cod binar se divide cu polinomul $1 + X$, atunci toate cuvintele sale au pondere pară.*

Reciproca este adevărată ?

5.15 *Arătaţi că dacă $g(X) \in Z_q[X]$ generează un (n, k) - cod ciclic, atunci $g(X^p)$ generează un (pn, pk) - cod ciclic.*

5.16 *Arătaţi că intersecţia a două coduri ciclice de aceeaşi lungime este tot un cod ciclic.*

Care este polinomul său generator ?

5.17 *Daţi o condiţie necesară şi suficientă pentru polinomul generator al unui cod ciclic de lungime impară în care $11\ldots1$ este cuvânt - cod.*

Capitolul 6

Decodificarea codurilor ciclice

6.1 Corectarea erorilor independente

După cum se ştie de la codurile liniare, o modalitate de caracterizare a tipurilor de erori care pot apare în transmisia mesajelor este dată pe baza sindromului. În cazul codurilor ciclice este convenabil să lucrăm cu un *sindrom polinomial.*

Definiţia 6.1 *Fie C un cod ciclic de lungime n, cu polinomul generator $g(X)$. Se numeşte "sindrom polinomial" $s(X)$ al cuvântului* **a** *de lungime n, restul împărţirii polinomului corespunzător $a(X)$ la $g(X)$.*

Deci, dacă se transmite cuvântul - cod $v(X) = q(X)g(X)$ şi se primeşte $a(X)$, sindromul va fi

$$s(X) = a(X) - v(X), \quad grad(s(X) < grad(g(X)).$$

Observaţii:

- Fie **e** eroarea generată prin transmiterea cuvântului - cod **v**. Deoarece $a(X) - e(X)$ este divizibil cu $g(X)$, rezultă că resturile împărţirii lui $a(X)$ şi $e(X)$ la $g(X)$ coincid. Deci se poate vorbi despre "*eroarea - tip*" **e**.

- Similar cu raţionamentul folosit la coduri liniare, vor fi considerate sindromuri toate polinoamele din $Z_q[X]$ de grad $< n - k$; pentru fiecare sindrom $s(X)$ putem alege o eroare - tip $e(X)$ de pondere minimă care corespunde acelui sindrom.

Structura algebrică a codurilor ciclice permite construcţia de metode noi de decodificare (cu corectarea posibilelor erori aferente), adesea mai eficiente decât algoritmii similari de la codurile liniare.

Vom începe cu prezentarea unui exemplu.

Exemplul 6.1 *Fie C codul Hamming binar de lungime 7, care are polinomul generator $g(X) = 1 + X + X^3$ (Capitolul 5. Exemplele 5.4 şi 5.6).*

Eroare - tip	*Sindrom*
0	0
1	1
X	X
X^2	X^2
X^3	$1 + X$
X^4	$X + X^2$
X^5	$1 + X + X^2$
X^6	$1 + X^2$

Să considerăm că s-a recepţionat cuvântul $\mathbf{a} = 1011001$*; sindromul lui este restul împărţirii polinomului* $X^6 + X^3 + X^2 + 1$ *la* $g(X)$*, adică* $s(X) = X + 1$*. Se ştie că orice cod Hamming corectează o eroare, deci va fi util să avem un tabel cu sindromurile tuturor erorilor - tip* $e(X) = X^i$ $(0 \leq i \leq 6)$*. Pentru că* $X + 1$ *este sindromul lui* $e(X) = X^3$*, tragem concluzia că a fost perturbat al treilea bit şi vom decodifica*

$$a(X) - e(X) = 1011001 - 0001000 = 1010001.$$

O dificultate în această metodă de decodificare - bazată pe sindrom - constă în necesitatea de a lista toate erorile - tip corespunzătoare sindromurilor, şi de a le parcurge la recepţia fiecărui cuvânt. O simplificare este însă propusă de Meggitt (1960), bazată pe structura ciclică a codurilor. Vom face corecţia numai pentru ultimul caracter (coeficientul termenului de grad maxim) al cuvântului **a** recepţionat. După aceea se efectuează o permutare ciclică şi se studiază din nou ultimul caracter. În acest fel, după n permutări ciclice, toate poziţiile au fost corectate.

Metoda prezintă două avantaje majore:

1. Se foloseşte doar lista erorilor - tip reprezentate prin polinoame de grad $n - 1$. (Astfel, în Exemplul 6.1, o singură eroare - tip are gradul polinomului asociat $n - 1 = 6$).

2. Calculul sindromului se efectuează o singură dată- la început - folosind circuitul liniar de împărţire cu $g(X)$. După ce a funcţionat n momente, în elementele de înmagazinare se găseşte sindromul. După aceea se neglijează ieşirea şi se lasă să funcţioneze circuitul. La fiecare pas (Capitolul 4, Figura 4.4) sindromul se înmulţeşte cu X, această operaţie având ca efect permutarea sa ciclică cu o poziţie.

Propoziţia 6.1 *Dacă un cuvânt* $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1})$ *are sindromul* $s(X)$, *atunci permutarea sa ciclică are sindromul* $s'(X)$, *care este restul împărţirii lui* $Xs(X)$ *la polinomul generator* $g(X)$.

Demonstraţie: Sindromul $s(X)$ este definit prin identitatea împărţirii

$$a(X) = q(X)g(X) + s(X),$$

unde $q(X)$ este câtul împărţirii lui $a(X)$ la $g(X)$. Deoarece se lucrează în algebra polinoamelor modulo $X^n - 1$, permutarea ciclică a lui $\{a(X)\}$ este $\{a'(X)\} = \{Xa(X)\}$, construit astfel:

$a'(X) = Xa(X) - a_{n-1}X^n + a_{n-1} = Xa(X) - a_{n-1}(X^n - 1) = Xa(X) - a_{n-1}g(X)h(X) = Xq(X)g(X) + Xs(X) - a_{n-1}g(X)h(X) = Xs(X) + g(X)[Xq(X) - a_{n-1}h(X)]$.

Rezultă de aici că resturile împărţirii lui $a'(X)$ şi $Xs(X)$ la polinomul generator $g(X)$ sunt aceleaşi. □

Se poate da acum

Algoritmul de decodificare Meggitt pentru (n, k) **- codurile ciclice binare**:

1. Se listează toate sindromurile corespunzătoare erorilor - tip **e**, reprezentate prin polinoame de grad $n - 1$;

2. Cuvântul recepţionat **a** este introdus în circuitul liniar de împărţire cu $g(X)$, care va funcţiona n tacţi;

3. Dacă în elementele de înmagazinare se obţine un polinom care se regăseşte în lista de sindromuri, se modifică bitul cel mai din dreapta al cuvântului primit;

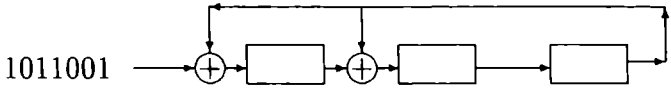
4. Se permută ciclic cuvântul primit şi - în acelaşi timp - se lasă să funcţioneze un tact circuitul liniar (neglijând ieşirea), după care se reia Pasul (3).

   Se repetă acest procedeu de $n - 1$ ori.

Exemplul 6.2 *Reluând codul Hamming binar din Exemplul 6.1, algoritmul va lucra astfel:*

- Singurul sindrom din listă este $1 + X^2$ *(de vector* 101*).*

- Dacă se recepţionează de exemplu $\mathbf{a} = 1011001$*, el se introduce în circuitul liniar*



După 7 *tacţi, în elementele de înmagazinare se obţine sindromul* 110*.*

- 110 *este diferit de* 101 *9singurul sindriom din listă0, deci* a_6 *a fost recepţionat corect.*

- Se permută circular $\mathbf{a}$ *în* 1101100 *şi se lasă circuitul să funcţioneze un tact; elementele de înmagazinare vor conţine acum* 011*.*

- Nici acest sindrom nu este pe listă, deci a_5 *a fost şi el corect ş.a.m.d.*

Paşii de lucru ai algoritmului sunt reprezentaţi în tabloul:

Pas	*Bit*	*Sindrom*
7	a_6	110
8	a_5	011
9	a_4	111
10	a_3	**101**
11	a_2	100
12	a_1	010
13	a_0	001

Deci singurul bit corectat (unde se transformă 1 *în* 0*) este* a_3*. Mesajul primit se decodifică în* 1010001*.*

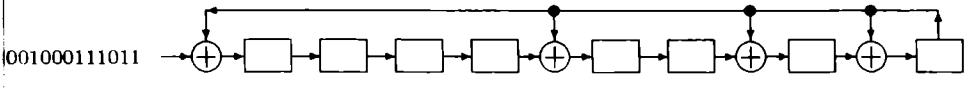
Exemplul 6.3 *Să considerăm codul ciclic binar generat de polinomul* $g(X) = 1 + X^4 + X^6 + X^7 + X^8$*. El este un* (15, 7) *- cod ciclic, de distanţă minimă* $d = 5$ *(după cum vom vedea mai târziu) deci poate corecta cel mult* 2 *erori independente. Lista completă a sindromurilor (deci a tuturor erorilor - tip de pondere* 0, 1, 2*) are* $C_{15}^0 + C_{15}^1 + C_{15}^2 =$ 121 *elemente. Folosind algoritmul Meggitt, rămân numai* 15 *sindromuri, listate în Tabelul 6.1*

Deci, la primirea unui cuvânt $\mathbf{a}$ *de lungime* 15*, vom calcula sindromul şi - dacă acesta se află în Tabelul 6.1 vom modifica bitul* a_{14}*. Apoi se face o permutare ciclică şi se corectează* a_{13} *ş.a.m.d.*

Să presupunem că s-a recepţionat mesajul $\mathbf{a} = 011001000111011$*. Pentru decodificare, el este introdus în circuitul liniar asociat codului (de împărţire cu* $g(X)$*)*

Tabelul 6.1:

Tip eroare	*Sindrom polinomial*	*Sindrom*
X^{14}	$X^7 + X^6 + X^5 + X^3$	000010111
$1 + X^{14}$	$X^7 + X^6 + X^5 + X^3 + 1$	10010111
$X + X^{14}$	$X^7 + X^6 + X^5 + X$	01000111
$X^2 + X^{14}$	$X^7 + X^6 + X^5 + X^3 + X^2$	00110111
$X^3 + X^{14}$	$X^7 + X^6 + X^5$	00000111
$X^4 + X^{14}$	$X^7 + X^6 + X^5 + X^4 + X^3$	00011111
$X^5 + X^{14}$	$X^7 + X^6 + X^3$	00010011
$X^6 + X^{14}$	$X^7 + X^5 + X^3$	00010101
$X^7 + X^{14}$	$X^6 + X^5 + X^3$	00010110
$X^8 + X^{14}$	$X^5 + X^4 + X^3 + 1$	10011100
$X^9 + X^{14}$	$X^7 + X^4 + X^3 + X + 1$	11011001
$X^{10} + X^{14}$	$X^3 + X^2 + X$	01110000
$X^{11} + X^{14}$	$X^7 + X^6 + X^5 + X^4 + X^2 + X$	01101111
$X^{12} + X^{14}$	$X^7 + X^6 + X^4 + X$	01001011
$X^{13} + X^{14}$	$X^7 + X^4 + X^3 + X^2$	00111001



care funcţionează 15 *tacţi. La sfârşitul lor, în elementele de înmagazinare se află sindromul, cu valoarea* $\mathbf{s} = 01001100$.

Pentru următorii paşi se obţin valorile

Bit	*Sindrom*	*Bit*	*Sindrom*	*Bit*	*Sindrom*
a_{14}	01001100	a_9	10101011	a_4	00101111
a_{13}	00100110	a_8	11011110	a_3	**10011100**
a_{12}	**00010011**	a_7	01101111	a_2	01001110
a_{11}	10000010	a_6	10111100	a_1	00100111
a_{10}	01000001	a_5	01011110	a_0	10011000

Două din aceste sindromuri - cele corespunzătoare lui a_{12} *şi* a_3 *- se află şi în Tabelul 6.1; deci aceşti doi biţi trebuie corectaţi (prin complementare). Cuvântul - cod obţinut este* 011101000111111.

De remarcat că în Exemplul 6.3 decodificarea nu este completă: algoritmul Meggitt corectează în total 121 erori - tip. Privit însă ca un cod

liniar, tabela de decodificare va avea $2^{15}/2^7 = 256$ linii, ceea ce înseamnă că acest cod este capabil să corecteze mult mai multe erori. Cu algoritmul Meggitt se vor corecta numai erorile independente simple sau duble, ignorând complet posibilitatea existenţei altor tipuri de erori.

Vom încerca în continuare să îmbunătăţim această situaţie.

6.2 Pachete - standard de erori

Fie $C \subseteq A_n$ un (n, k) - cod ciclic peste Z_q, capabil să corecteze t erori.

Definiţia 6.2 *Pentru un polinom* $e(X) = X^s e_0(X) \in Z_q[X]$ *cu* $e_0(0) \neq 0$, *spunem că "lungimea pachet" a lui* $\mathbf{e}$ *este* $1 + grad(e_0(X))$.

Cuvântul $\mathbf{e} \in Z_q^n$ *este un "pachet - standard de lungime* j*" dacă gradul minim al tuturor polinoamelor* $\{X^i e(X)\}$ $(0 \leq i \leq n-1)$ *este* $j - 1$.

Dacă se trimite un cuvânt $\mathbf{a}$ şi se recepţionează $\mathbf{v}$, spunem că erorile au afectat j poziţii consecutive, dacă eroarea - tip $\mathbf{e} = \mathbf{v} - \mathbf{a}$ este un pachet de lungime j. Acesta este un *pachet - standard* de j erori.

Exemplul 6.4 *Fie* $n = 7$ *şi* $\mathbf{e} = 0101100$. *Atunci* $e(X) = X + X^3 + X^4 = X(1 + X^2 + X^3)$. *Efectuând permutări ciclice, toate polinoamele obţinute au grad mai mare decât* 3, *înafară de* $\{X^6 e(X)\} = \{1 + X^2 + X^3\}$, *care are gradul* 3. *Deci* $\mathbf{e}$ *este un pachet - standard de lungime* 4.

Dacă se ia $\mathbf{e} = 1000100$, $1 + X^4$ *are gradul* 4, *dar permutarea ciclică* $\{X^3(1 + X^4)\} = \{1 + X^3\}$ *are gradul* 3, *deci şi acesta este un pachet - tip de lungime* 4.

Reamintim că la codurile liniare se construia un tablou standard în care pe fiecare linie se aflau toate cuvintele cu acelaşi sindrom. Codul era corector de t erori independente dacă toate cuvintele din Z_q^n de pondere cel mult t erau pe linii diferite; aceste cuvinte constituiau reprezentanţii sindromurilor şi apăreau pe prima coloană, fiind considerate erori - tip pentru fiecare linie.

Astfel de tabele se pot construi şi pentru codurile ciclice; aici se iau ca reprezentanţi pe fiecare linie, pachetele - standard de erori de lungime minimă. În acest fel, un cod liniar corector de t erori (independente) este corector de pachete de j erori consecutive, dacă toate cuvintele de lungime - pachet cel mult j sunt alese ca reprezentanţi.

Observaţia 6.1

1. *Condiţia ca j erori să formeze un pachet - standard este mai restrictivă decât dacă cele j erori sunt independente. Un pachet de erori afectează cuvântul pe o porţiune fixată, de lungime j, pe când erorile independente pot apare oriunde în cuvânt.*

2. *Când spunem că sunt afectate de erori j poziţii consecutive, aceasta nu înseamnă că toate cele j poziţii sunt greşite efectiv, ci doar că aceste j poziţii formează componentele unui pachet - standard de erori (unele din ele putând fi corecte).*

Definiţia 6.3 *Un (n,k) - cod ciclic corectează pachetele de j erori consecutive dacă orice pachet - standard de lungime cel mult j este selectat ca sindrom.*

Lema 6.1 *Dacă un cod C este corector de t erori independente şi corector de pachete de j erori consecutive, atunci $t \leq j$.*

Demonstraţie: Exerciţiu. □

Exemplul 6.5 *Să considerăm toate pachetele - standard nenule de lungime cel mult 3 în Z_2^{15}. Fiecare astfel de cuvânt este de forma $\{e(X)\} = \{X^i e_0(X)\}$ cu $0 \leq i \leq 14$ şi $e_0(X) \in \{1, 1+X, 1+X^2, 1+X+X^2\}$.*

În total sunt $15 \cdot 4 = 60$ astfel de pachete de erori.

Exemplul 6.6 *Fie $g(X) = 1 + X + X^2 + X^3 + X^6$ polinomul generator al unui $(15,9)$ - cod ciclic binar. El nu este un cod corector de 3 erori independente pentru că sunt 576 tipuri de erori de pondere cel mult 3 şi numai $2^{15}/2^9 = 64$ sindromuri. În schimb sunt numai 61 pachete - standard de lungime maxim 3 (Exemplul 6.5, la care se adaugă pachetul nul), deci acest cod poate corecta orice pachet de maxim 3 erori consecutive. Faptul că acesta este un cod ciclic corector de pachete de 3 erori consecutive rezultă din calculul sindromurilor polinoamelor $\{X^i e_0(X)\}$ unde $0 \leq i \leq 14$ şi $e_0(X) \in \{1, 1+X, 1+X^2, 1+X+X^2\}$.*

6.2.1 Detectarea pachetelor de erori

Teorema 6.1 *Un (n,k) - cod ciclic detectează orice pachet de maxim $n-k$ erori.*

Demonstraţie: Dacă $\{e(X)\}$ este o eroare - tip de lungime cel mult $n-k$, atunci $e(X) = X^i e_0(X)$ cu $grad(e_0(X)) < n-k$. Fie $g(X)$ polinomul generator al codului, cu $grad(g(X)) = n-k$.

Ştim că $\{e(X)\}$ este cuvânt - cod $\iff$ $g(X)|e(X)$.

Cum $g(X)|X^n-1$, el va fi prim cu X^s; deci $\{e(X)\}$ este cuvânt - cod dacă şi numai dacă $g(X)|e_0(X)$, absurd deoarece $grad(e_0(X)) < n-k = grad(g(X))$. □

Teorema 6.2 *Proporţia pachetelor - standard de $j > n-k$ erori pe care nu le poate detecta un (n,k) - cod ciclic este*

$$\begin{cases} \dfrac{q^{-(n-k-1)}}{q-1} & \text{dacă } j = n-k+1, \\ q^{-(n-k)} & \text{dacă } j > n-k+1. \end{cases}$$

Demonstraţie: Fie $\{e(X)\} = \{X^s e_0(X)\}$ unde $grad(e_0(X)) = j-1$. Să numărăm câte asemenea pachete - standard de erori sunt posibile. Ca prim şi ultim coeficient al lui $e_0(X)$ poate fi orice element din $Z_q \setminus \{0\}$ (în număr de $q-1$), iar coeficienţii intermediari pot fi orice elemente din Z_q (în număr de q). Deci sunt $(q-1)^2 q^{j-2}$ pachete - standard de erori care încep şi se termină în aceeaşi poziţie.

Un pachet de erori $\{e(X)\}$ nu este detectat de un (n,k) - cod ciclic dacă şi numai dacă $\{e(X)\}$ este cuvânt - cod, adică $e_0(X) = g(X)h(X)$, unde $grad(h(X)) = grad(e_0(X)) - grad(g(X)) = j-1-(n-k)$. Deci polinomul $h(X)$ are $j-n+k$ coeficienţi. Apar două situaţii:

1. $grad(h(X)) = 0$, adică $j = n-k+1$. Atunci $h(X)$ se reduce la o constantă şi există $q-1$ asemenea polinoame $h(X)$. Deci, raportul dintre numărul pachetelor - standard de lungime j nedetectabile şi numărul total al pachetelor - standard de lungime j care pot fi localizate este
$$\frac{q-1}{(q-1)^2 q^{j-2}} = \frac{q^{-(n-k-1)}}{q-1}.$$

2. Dacă $grad(h(X)) > 0$, adică $j > n-k+1$, vom avea $(q-1)^2 q^{j-n+k-2}$ polinoame $h(X)$ posibile (pachete - standard de lungime j) nedetectabile. Rezultă că raportul de sus este în acest caz
$$\frac{(q-1)q^{j-n+k-2}}{(q-1)^2 q^{j-2}} = q^{-(n-k)}.$$

□

Exemplul 6.7 *Codul ciclic definit în Exemplul 6.3 detectează orice pachet de maxim 9 erori consecutive.*

Dintre pachetele - standard de 10 erori posibile care pot apare, doar $1/2^7$ nu pot fi detectate, iar dintre pachetele - standard de $11-15$ erori nu pot fi detectate $1/2^8$.

La codul ciclic definit în Exemplul 6.6, proporţia este de $1/2^5$ erori nedectabile de lungime 7 şi $1/2^6$ erori nedectabile de lungimi $8-15$ (codul detectează orice pachet - standard de 6 erori).

6.2.2 Corectarea pachetelor de erori

Algoritmul de corectare a pachetelor - standard de erori este o variantă a Algoritmului Meggitt.

Fie $C = (\{g(X)\})$ un (n,k) - cod ciclic binar corector de pachete de maxim j erori consecutive, şi $\mathbf{v} \in Z_q^n$ un cuvânt recepţionat.

1. Se calculează sindromul $\{s(X)\}$;
2. Pentru fiecare $i \geq 0$ se calculează $s_i(X) = X^i s(X) \ mod \ g(X)$, până se ajunge la un p cu $grad(s_p(X)) < j-1$.

   Atunci eroarea - tip este $\{e(X)\} = \{X^{n-p} s_p(X)\}$ şi se generează cuvântul - cod $\mathbf{v}+\mathbf{e}$.

Exemplul 6.8 *Conform Exemplului 6.6, polinomul $g(X) = 1+X+X^2+X^3+X^6$ generează un $(15,9)$ - cod ciclic binar corector de pachete - standard de 3 erori. Să folosim algoritmul Meggitt pentru a decodifica* $\mathbf{v} = 111100100001010$.

$s(X) = 1+X+X^2+X^3+X^6+X^{11}+X^{13} \ (mod \ g(X)) = 1+X^3+X^4+X^5$.

$s_1(X) = Xs(X) \ (mod \ g(X)) = 1+X^2+X^3+X^4+X^5$,

$s_2(X) = X^2 s(X) \ (mod \ g(X)) = 1+X^2+X^4+X^5$,

$s_3(X) = X^3 s(X) \ (mod \ g(X)) = 1+X^2+X^5$,

$s_4(X) = X^4 s(X) \ (mod \ g(X)) = 1+X^2$.

S-a ajuns la $grad(s_4(X)) = 2 \leq 3-1$. *Deci eroarea - tip este*

$$\{e(X)\} = \{X^{15-4} s_4(X)\} = \{X^{11}+X^{13}\}.$$

Cuvântul - cod transmis a fost

$\mathbf{a} = \mathbf{v}+\mathbf{e} = 111100100001010 + 000000000001010 = 111100100000000.$

Exemplul 6.9 *Codul ciclic binar definit în Exemplul 6.3 este de asemenea corector de pachete - standard de 3 erori. Să presupunem că s-a recepţionat cuvântul* $\mathbf{v} = 010010110011001$. *Sindromul său este* $s(X) = v(X) \ (mod \ g(X)) = 1 + X^6 + X^7$. *Căutăm o valoare a lui* i *astfel ca* $X^i s(X)$ *să fie un polinom de grad cel mult* 2. *Se obţine* $X^5 s(X) = 1 + X + X^2$. *Deci eroarea - tip este*

$$\{e(X)\} = \{X^{10}(1 + X + X^2)\} = \{X^{10} + X^{11} + X^{12}\}.$$

S-a transmis cuvântul - cod
$\mathbf{a} = \mathbf{v} + \mathbf{e} = 010010110011001 + 000000000011100 = 010010110000101.$

6.3 Exerciţii

6.1 *Demonstraţi Lema 6.1*

6.2 *Arătaţi că dacă un cod ciclic detectează o eroare - tip* $\mathbf{e}$, *atunci detectează toate erorile - tip obţinute prin permutări ciclice ale lui* $\mathbf{e}$.

6.3 *Verificaţi că pachetele - standard ciclice de erori de lungime* 3 *din* Z_2^{15} *au sindromuri diferite pentru codul din Exemplul 6.3.*

6.4 *Aceeaşi problemă pentru codul ciclic binar definit în Exemplul 6.6.*

6.5 *Arătaţi că* $g(X) = 1 + X^2 + X^4 + X^5$ *generează un* $(15, 10)$ *- cod ciclic binar corector de pachete - standard de maxim* 2 *erori. Este acesta un cod corector de* 2 *erori independente ?*

6.6 *Arătaţi că* $g(X) = 1 + X^3 + X^4 + X^5 + X^6$ *generează un* $(15, 9)$ *- cod ciclic binar corector de pachete - standard de maxim* 3 *erori. Este acesta un cod corector de* 3 *erori independente ?*

6.7 *Arătaţi că* $g(X) = 1 + X^4 + X^6 + X^7 + X^8$ *generează un* $(15, 7)$ *- cod ciclic binar corector de maxim* 2 *erori independente şi de pachete - standard de* 4 *erori.*

6.8 *Fie codul din Exemplul 6.6. Să se decodifice mesajele:*
101101110001000, 001101100010101, 100110101010011,
101101000010111, 000000111110000.

6.9 *Fie* $(15, 10)$ *- codul ciclic binar generat de* $g(X) = 1 + X^2 + X^4 + X^5$. *Ce lungime au pachetele - standard de erori pe care le poate corecta ?*
Decodificaţi mesajele:
010101000010010, 011010010010100, 001101000000100,
000100010100101, 000000011111001.

Capitolul 7

Alte definiţii ale codurilor ciclice

Înafara definirii codurilor ciclice folosind noţiunea de ideal, se pot utiliza şi alte caracteristici pentru introducerea acestor clase de coduri. Pentru aceasta vom trece în revistă la început câteva noţiuni algebrice elementare.

7.1 Elemente primitive în extensii Galois

În acest paragraf ne vom referi la un corp finit $(F, +, \cdot)$, în care s-au notat cu 0 respectiv 1 elementele unitate faţă de cele două operaţii.

Definiţia 7.1 *Un element $a \in F$ are ordin n $(n \geq 1)$ dacă $a^n = 1$ şi $a^k \neq 1$, $\forall k$, $0 < k < n$. Vom scrie $ord(a) = n$.*

Propoziţia 7.1 *Într-un corp finit $(F, +, \cdot)$:*
- *Orice element nenul are un ordin finit;*
- *Dacă $n = ord(a)$ atunci $a, a^2, \ldots, a^n$ sunt distincte;*
- *$a^k = 1$ dacă şi numai dacă $ord(a) | k$ $(ord(a)$ este divizor al lui $k)$.*

Demonstraţie: Fie $a \in F$, $a \neq 0$. Cum F este un corp finit, elementele $a, a^2, a^3, \ldots$ nu pot fi toate distincte. Vom alege cel mai mic n pentru care există un număr i cu $a^{n+i} = a^i$. Relaţia se poate simplifica (lucrăm într-un corp) cu a^i şi se obţine $a^n = 1$. Din construcţie, $a, a^2, \ldots, a^n$ sunt distincte, deci n este ordinul lui a.

Pentru $k = ni$ avem $a^k = (a^n)^i = 1^i = 1$. Reciproc, să presupunem $a^k = 1$. Teorema împărţirii cu rest dă $k = qn + r$, $0 \leq r < n$. Avem $1 = a^k = a^{qn}a^r = a^r$.

Cum $r < n$ rezultă $r = 0$. □

Definiţia 7.2 *Un element $a \in F$ este primitiv dacă $F^* = F \setminus \{0\} = \{1, a, a^2, a^3, \ldots\}$.*

Observaţia 7.1 *Dacă F are r elemente, atunci un element este primitiv dacă şi numai dacă are ordinul $r - 1$.*

Teorema 7.1 *Orice corp finit are cel puţin un element primitiv.*

Demonstraţie: Deoarece F este finit ($card(F) = r$), se poate găsi un element $a \in F$ de ordin n maxim. Evident $n \leq r - 1$. Mai trebuie arătat că $n \geq r - 1$.

Fie $b \in F^*$ un element arbitrar, de ordin s. Să-l descompunem pe s în factori primi: $s = p^i q^j \ldots$ ($p, q, \ldots$ numere prime distincte). La rândul lui $n = p^t n'$ unde n' nu este divizibil cu p (iar t poate fi eventual zero).

Să considerăm $s' = s/p^i$ (deci $s = p^i s'$) şi fie $c = a^{p^t} b^{s'} \in F^*$. Vrem să arătăm că c are ordinul $m = p^i n'$. Pentru aceasta, avem:

$$c^m = a^{p^t m} b^{s'm} = a^{p^t n' p^i} b^{p^i s' n'} = (a^n)^{p^t} (b^s)^{n'} = 1^{p^t} 1^{n'} = 1.$$

Mai rămâne de arătat că $c^{m'} = 1 \Longrightarrow m' \geq m$. Este suficient să verificăm că p^i şi n' sunt divizori ai lui m'; cum cele două numere sunt prime între ele, va rezulta că produsul $m = p^i n'$ este de asemenea divizor al lui m'.

Din $1 = (c^{m'})^{n'} = a^{p^t n' m'} b^{s' m' n'} = (a^n)^{m'} b^{s' m' n'} = b^{s' m' n'}$ rezultă (Propoziţia 7.1) că ordinul $s = p^i s'$ al lui b divide $s' m' n'$; cum $(p^i, n') = 1$, rezultă $p^i | m'$.

În plus, din $1 = (c^{m'})^{p^t} = a^{p^t p^i m'} b^{p^t s' m'} = a^{p^t p^i m'} (b^s)^{m'} = a^{p^t p^i m'}$ rezultă că ordinul $n = p^t n'$ al lui a divide $p^t p^i m'$; deci $n' | p^i m'$, adică $n' | m'$.

Cum n este cel mai mare ordin ale elementelor din F, avem $ord(c) \leq ord(a)$ sau $m \leq n$, adică $p^i n' \leq p^t n'$; deci $i \leq t$, ceea ce înseamnă că p^i divide $n = p^t n'$.

În mod similar, toţi factorii lui s sunt divizori ai lui n, deci $s | n$.

Am arătat următoarea afirmaţie:

$$\forall b \in F^* \quad \Longrightarrow ord(b) | n.$$

Deci b este o rădăcină a polinomului $X^n - 1 = 0$, de unde rezultă că polinomul $X^n - 1 = 0$ are $r - 1$ rădăcini, adică $r - 1 \leq n$. □

Fie $(F, +, \cdot)$ un corp finit şi $f(X) \in F[X]$ un polinom arbitrar cu coeficienţi în F. Spunem că $a \in F$ este *rădăcină* a lui f dacă $f(a) = 0$.

Se poate demonstra imediat afirmaţia:

Propoziţia 7.2 *Dacă un polinom f are rădăcinile distincte $a_1, a_2, \ldots, a_n$ atunci el este divizibil cu polinomul $(X - a_1)(X - a_2)\ldots(X - a_n)$.*

Exemplul 7.1 *Polinomul $X^3 + 1$ are o rădăcină 1 în Z_2. Deci $X^3 + 1$ se divide cu $X + 1$:*

$$X^3 + 1 = (X + 1)(X^2 + X + 1).$$

Aceasta este o factorizare completă: deoarece $X^2 + X + 1$ nu are rădăcini în Z_2, el nu poate fi descompus în produsul a două polinoame de gradul 1. Acelaşi polinom $X^3 + 1$ are în Z_3 pe 2 ca rădăcină triplă, deci aici putem scrie:

$$X^3 + 1 = (X + 1)^3.$$

Rezultă că factorizarea unui polinom depinde de corpul în care este definit.

Din Teorema 7.1 şi Propoziţia 7.2 rezultă că dacă $a \in F$ este un element de ordin n, atunci $X^n - 1$ se poate descompune în

$$X^n - 1 = (X - a)(X - a^2)\ldots(X - a^n).$$

Definiţia 7.3 *Un polinom $f(X) \in F[X]$ de grad n este "ireductibil" în corpul F dacă nu se poate descompune în produsul a două polinoame din $F[X]$ de grad mai mic decât n. În caz contrar, $f(X)$ este "reductibil".*

Observaţia 7.2

- *Orice polinom liniar este ireductibil. Pentru polinoame de grad cel puţin 2, Propoziţia 7.2 afirmă că* **un polinom ireductibil într-un corp nu are rădăcini în acel corp.**

  Este interesant că reciproca nu este adevărată decât pentru polinoamele de grad 2 şi 3. Astfel, polinomul $f(X) = (X^2 + X + 1)^2 \in Z_2[X]$, deşi nu are rădăcini în Z_2, este reductibil.

- *În $\mathcal{R}$ singurele polinoame ireductibile sunt de gradul 1 sau 2. Dacă un polinom are grad impar, el are sigur o rădăcină reală, iar dacă este de grad par, atunci are sau cel puţin o rădăcină reală, sau cel puţin două rădăcini complexe de forma $a \pm ib$. În acest ultim caz, el se va divide cu $(x - a)^2 + b^2$, care este un polinom de gradul 2 în $\mathcal{R}[X]$.*

- *În corpul complex C, teorema fundamentală a algebrei asigură că orice polinom ireductibil are gradul 1.*

Definiţia 7.4 *Caracteristica unui corp finit F este cel mai mic număr de termeni ai sumei $S = 1 + 1 + \ldots + 1$ cu proprietatea $S = 0$.*

Propoziţia 7.3 *Caracteristica unui corp finit este număr prim.*

Demonstraţie: Să considerăm elementele $S_i = 1 + 1 + \ldots + 1$ de câte i termeni fiecare. Deoarece nu pot fi o infinitate de valori distincte, fie p minim cu proprietatea $\exists i,\ S_i = S_{i+p}$. Rezultă $S_p = S_{i+p} - S_i = 0$; deci F are caracteristica p.

Presupunem $p = jt$ cu $1 \leq j < p$. Evident $0 = S_{jt} = S_j + S_j + \ldots + S_j$ (t termeni). Deoarece $j < p$ avem $S_j \neq 0$. Împărţind relaţia cu S_j se obţine $0 = 1 + 1 + \ldots + 1 = S_t$. Deci $t = p$. □

Corolarul 7.1 *Orice corp de caracteristică q este o extensie a lui Z_q.*

Demonstraţie: Exerciţiu.

Corolarul 7.2 *$GF(q^r)$ este un corp de caracteristică q.*

Demonstraţie: Reamintim că $GF(q^r)$ conţine toate polinoamele de grad cel mult $r-1$ din $Z_q[X]$ (Capitolul 4). În particular, polinoamele de grad 0 (constantele) formează un subcorp izomorf cu Z_q, care are caracteristica q. □

Propoziţia 7.4 *Într-un corp de caracteristică q avem $(a+b)^q = a^q + b^q$.*

Demonstraţie: Se foloseşte binomul lui Newton, în care $C_q^k = 0,\ \forall k,\ 0 < k < q$. □

7.2 Polinoame minimale

În cele ce urmează vom restrânge studiul la cazul $F = Z_q$, q număr prim.

Definiţia 7.5 *Fie β un element dintr-o extensie a lui Z_q. Se numeşte "polinom minimal" al lui β polinomul normat $g(X) \in Z_q[X]$ de grad minim, cu $g(\beta) = 0$.*

Existenţa polinoamelor minimale este asigurată de Teorema 4.1, pe baza căreia s-au definit extensiile Galois.

Exemplul 7.2 *Să considerăm extensia $GF(2^3)$ generată de rădăcina α a plinomului $1 + X + X^3$. Deoarece α este primitiv, avem $GF(2^3) = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$. Polinoamele minimale ale fiecărui element sunt date în tabelul:*

Element	*Polinom minimal*
0	X
1	$1 + X$
$\alpha, \alpha^2, \alpha^4$	$1 + X + X^3$
$\alpha^3, \alpha^5, \alpha^6$	$1 + X^2 + X^3$

Propoziţia 7.5 *Un polinom minimal $g(X) \in Z_q[X]$ este ireductibil peste Z_q.*

Demonstraţie: Dacă ar exista descompunerea $g(X) = u(X)v(X)$ cu $u(X), v(X) \in Z_q[X]$, atunci vom avea $u(\beta) = 0$ sau $v(\beta) = 0$, ceea ce contrazice definiţia polinomului minimal pentru β. □

Corolarul 7.3 *Dacă $f(X) \in Z_q[X]$ verifică $f(\beta) = 0$ iar $g(X)$ este polinomul minimal al lui β, atunci $g(X)|f(X)$.*

De aici rezultă că un polinom normat ireductibil peste Z_q, care admite pe β ca rădăcină este polinom minimal al lui β. Acest polinom este unic.

Teorema 7.2 *Fie $\beta \in GF(q^k)$. Atunci există un polinom minimal al lui β de grad cel mult k.*

Demonstraţie: După cum rezultă din definirea extensiilor Galois (Capitolul 4) şi Teorema 5.3 (Capitolul 5) $GF(q^k)$ are dimensiunea k. O bază a lui este $\{1\}, \{X\}, \ldots, \{X^{k-1}\}$. Deci cele $k+1$ elemente $1, \beta, \beta^2, \ldots, \beta^k \in GF(q^k)$ sunt liniar dependente. Relaţia lor de dependenţă conduce la construirea unui polinom de grad cel mult k pentru care β este rădăcină. □

Teorema 7.3 *Elementele din Z_q^* sunt soluţiile ecuaţiei $X^{q-1} - 1 = 0$.*

Demonstraţie: Elementele nenule din Z_q formează grup multiplicativ. Ordinul fiecărui element divide ordinul grupului, care este $q-1$ (Teorema 7.1). Fie $\beta \in Z_q \setminus \{0\}$; subgrupul ciclic generat de β este $1, \beta, \beta^2, \ldots, \beta^{t-1}$ unde $b^t = 1$ şi $t|q-1$. Deci $\beta^{q-1} = 1$.

În plus, ecuaţia $X^{q-1} - 1 = 0$ are $q-1$ rădăcini. □

Teorema 7.4 $X^m - 1 | X^n - 1 \iff m|n$.

Demonstraţie: "$\Leftarrow$": Fie $n = md$. Cum $Y - 1$ divide pe $Y^d - 1$, dacă se ia $Y = X^m$, vom avea $X^m - 1 | X^{md} - 1$.

"$\Rightarrow$": Să presupunem că $X^m - 1 | X^n - 1$ şi fie $n = md + s$, $(s < m)$. Avem $X^n - 1 = X^s(X^{md} - 1) + X^s - 1 = q(X)(X^m - 1) + r(X)$ cu $grad(r(X)) = s < m$.

Pentru a verifica ipoteza, trebuie ca $s = 0$, deci $n = md$. □

Teorema 7.5 *Fie $f(X) \in Z_q[X]$ şi β o rădăcină a sa (eventual dintr-o extensie a lui Z_q). Atunci şi β^q este rădăcină a lui $f(X)$.*

Demonstraţie: Scriem $f(X) = a_0 + a_1X + \ldots + a_nX^n$. Cum Z_q este corp de caracteristică q, este adevărată relaţia $(a + b)^q = a^q + b^q$ (Propoziţia 7.4).

De asemenea (Teorema 7.3) $\forall a \in Z_q \setminus \{0\}$ avem $a^{q-1} - 1 = 0$, deci $a^q = a$.

Atunci se poate scrie

$$(f(X))^q = a_0^q + a_1^q X^q + \ldots + a_n^q (X^n)^q = a_0 + a_1 X^q + \ldots + a_n X^{nq} = f(X^q).$$

În particular, $f(\beta^q) = (f(\beta))^q = 0$. □

Teorema 7.6 *Orice polinom normat ireductibil peste Z_q de grad m este un factor al polinomului $X^{q^m} - X$.*

Demonstraţie: Dacă $g(X) = X$, afirmaţia este banală.

Să presupunem că $g(X) \neq X$ şi fie $\beta \in GF(q^m)$ o rădăcină nenulă a sa, deci $g(\beta) = 0$. Cum $g(X)$ este ireductibil, el va fi chiar polinomul minimal al lui β. Conform Teoremei 7.3, β satisface ecuaţia $X^{q^m-1} - 1 = 0$, deci (Corolarul 7.3) $X^{q^m-1} - 1$ se divide cu polinomul minimal $g(X)$. □

Teorema 7.7 *Orice factor ireductibil $g(X) \in Z_q[X]$ al lui $X^{q^m} - X$ are gradul cel mult m.*

Demonstraţie: Fie $g(X) | X^{q^m} - X$, ireductibil peste Z_q, cu $grad(g(X)) = k$. Vom considera algebra polinoamelor modulo $g(X)$, în care luăm $\alpha = \{X\}$. Se ştie (Teorema 4.1) că $g(\alpha) = 0$. Un element oarecare din această algebră este de forma

$$\beta = a_0 + a_1\alpha + \ldots + a_{k-1}\alpha^{k-1}.$$

Atunci $\beta^{q^m} = a_0^{q^m} + a_1^{q^m}\alpha^{q^m} + \ldots + a_{k-1}^{q^m}(\alpha^{k-1})^{q^m} = a_0 + a_1\alpha^{q^m} + \ldots + a_{k-1}(\alpha^{q^m})^{k-1} = a_0 + a_1\alpha + \ldots + a_{k-1}\alpha^{k-1} = \beta$,
deoarece $GF(q^m)$ are tot caracteristica q ca şi Z_q, iar α - fiind o rădăcină

a lui $g(X)$ - este rădăcină şi a lui $X^{q^m} - X$, deci $\alpha^{q^m} = \alpha$, de unde rezultă $\alpha^{jq^m} = \alpha^j \ \forall j \geq 0$.

Am obţinut în final faptul că β este o soluţie a ecuaţiei $X^{q^m} - X = 0$. Sunt posibile q^k asemenea elemente β distincte, iar cum ecuaţia are q^m rădăcini, rezultă $q^k \leq q^m$, deci $k \leq m$. □

Teorema 7.8 *Fie $\beta \in GF(q^m)$ de polinom minimal $g(X)$, $grad(g(X)) = k$ şi $ord(\beta) = t$. Atunci*

(i) $t | q^k - 1$;

(ii) *k este minim cu proprietatea (i).*

Demonstraţie: Ştim (Teorema 7.6) că $g(X)$ este un factor al lui $X^{q^k} - X$; deci $\beta^{q^k} = \beta$, de unde rezultă $t | q^k - 1$.

Să presupunem acum că există $p < k$ cu $t | q^p - 1$. Atunci $\beta^{q^p - 1} = 1$, adică β este o rădăcină a ecuaţiei $X^{q^p} - X = 0$, deci $g(X)$ este un factor al polinomului $X^{q^p} - X$. Conform Teoremei 7.7, rezultă $k \leq p$, contradicţie. □

Teorema 7.9 *Fie polinomul $g(X) \in Z_q[X]$, $grad(g(X)) = m$ şi $\beta \in GF(q^m)$ o rădăcină a sa. Atunci $\beta, \beta^q, \beta^{q^2}, \ldots, \beta^{q^{m-1}}$ sunt toate rădăcinile lui $g(X)$.*

Demonstraţie: Deoarece $g(\beta) = 0$, avem - conform Teoremei 7.5 - că $\beta^q, \beta^{q^2}, \ldots, \beta^{q^{m-1}}$ sunt şi ele rădăcini ale lui $g(X)$. De asemenea, conform Teoremei 7.6, β este rădăcină a lui $X^{q^m} - X$, adică $\beta^{q^m} = \beta$. Mai rămâne de arătat că aceste rădăcini sunt distincte. Presupunem că există $0 \leq i < j < m$ cu $\beta^{q^i} = \beta^{q^j}$. Avem

$$\beta = \beta^{q^m} = \left(\beta^{q^j}\right)^{q^{m-j}} = \left(\beta^{q^i}\right)^{q^{m-j}} = \beta^{q^{m+i-j}}.$$

Deci β este rădăcină a polinomului $X^{q^{m+i-j}} - X$ şi - cu Teorema 7.7, $grad(g(X)) = m \leq m + i - j < m$, contradicţie. □

Din Teorema 7.9 rezultă că luând extensia $GF(q^m)[X]$, $g(X)$ se poate scrie

$$g(X) = (X - \beta)(X - \beta^q)\ldots(X - \beta^{q^{m-1}}).$$

Teorema 7.10 *Fie $g(X)$ un polinom normat ireductibil peste Z_q, $grad(g(X)) = m$, şi $\beta \in GF(q^m)$ o rădăcină a sa. Atunci toate rădăcinile lui $g(X)$ au acelaşi ordin.*

Demonstraţie: Fie $p = ord(\beta)$, $p' = ord(\beta^{q^j})$ $(1 < j < m)$ care - conform Teoremei 7.9 - este tot rădăcină a lui $g(X)$. Avem:

$$\left(\beta^{q^j}\right)^p = (\beta^p)^{q^j} = 1 \text{ şi deci } p'|p.$$

De asemenea,

$$\beta^{p'} = \left(\beta^{q^m}\right)^{p'} = \left(\left(\beta^{q^j}\right)^{q^{m-j}}\right)^{p'} = \left(\left(\beta^{q^j}\right)^{p'}\right)^{q^{m-j}} = 1 \quad \Rightarrow p|p'.$$

S-a mai folosit Teorema 7.8, conform căreia p divide $q^m - 1$ dar nu divide nici un număr de forma $q^s - 1$ cu $s < m$.

Deci $p = p'$. □

7.3 Coduri ciclice definite prin rădăcini

O altă metodă de definire a unui cod ciclic constă în listarea tuturor rădăcinilor comune (eventual într-o extensie a lui Z_q) ale cuvintelor - cod (considerate ca polinoame).

Definiţia 7.6 *Fie $\alpha_1, \alpha_2, \ldots, \alpha_r$ elemente dintr-o extensie a lui Z_q. Codul ciclic C este format din toate elementele $\{f(X)\}$ din algebra polinoamelor modulo $X^n - 1$, care admit pe α_i $(1 \leq i \leq r)$ ca rădăcini simple.*

De remarcat că, în această definiţie, n este deocamdată nedeterminat.

Conform Corolarului 7.3, $f(X)$ se va divide cu fiecare polinom minimal $m_i(X)$ corespunzător rădăcinii α_i $(1 \leq i \leq r)$. Deci polinomul generator al codului ciclic este

$$g(X) = cmmmc\{m_1(X), m_2(X), \ldots, m_r(X)\}.$$

Propoziţia 7.6 $C = (\{g(X)\})$.

Demonstraţie: Exerciţiu.

Deoarece polinomul generator $g(X)$ divide pe $X^n - 1$, rezultă că fiecare α_i este rădăcină a lui $X^n - 1$. Deci ordinul $ord(\alpha_i)$ al fiecărei rădăcini α_i $(1 \leq i \leq r)$ va divide pe n. O modalitate naturală de definire a lui n este atunci

$$n = cmmmc\{ord(\alpha_1), ord(\alpha_2), \ldots, ord(\alpha_r)\}.$$

Un caz particular important este acela în care toate rădăcinile $\alpha_1, \alpha_2, \ldots, \alpha_r$ sunt puteri ale unui anumit element, adică

$$\alpha_i = \alpha^{u_i}, \quad 1 \leq i \leq r.$$

Fie $p = ord(\alpha)$. Atunci polinomul minimal $m_i(X)$ al lui α_i va avea (Teorema 7.9) toate rădăcinile printre elementele $\alpha^{u_i}, \alpha^{qu_i}, \alpha^{q^2 u_i}, \ldots$.

Numărul factorilor lui $g(X)$ şi gradul fiecărui polinom minimal $m_i(X)$ se vor determina atunci din ordinul p şi din exponenţii $u_i, qu_i, q^2 u_i, \ldots$ $(1 \leq i \leq r)$. Într-adevăr, numărul claselor distincte de resturi modulo p din succesiunea $u_i, qu_i, \ldots$ va da gradul polinomului minimal $m_i(X)$.

Exemplul 7.3 *Fie $\alpha \in GF(2^4)$ primitiv, rădăcină a polinomului $1+X+X^4$, ireductibil peste Z_2. Să definim un cod C care să aibă ca rădăcini $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6$.*

Fiind primitiv, $ord(\alpha) = 2^4 - 1 = 15$ (vezi şi Capitolul 4, Exemplul 4.8). Fie $m_i(X)$ polinomul minimal al rădăcinii α_i $(1 \leq i \leq 6)$.

Rădăcinile lui $m_1(X)$ sunt $\alpha, \alpha^2, \alpha^4, \alpha^8$ $(\alpha^{16} = \alpha)$, deci $m_1(X) = m_2(X) = m_4(X) = (X - \alpha)(X - \alpha^2)(X - \alpha^4)(X - \alpha^8)$.

Acest polinom este cunoscut, anume $1 + X + X^4$.

Rădăcinile lui $m_3(X)$ sunt $\alpha^3, \alpha^6, \alpha^{12}, \alpha^{24} = \alpha^9$ $(\alpha^{18} = \alpha^3)$, deci $m_3(X)$ este un polinom de gradul 4, egal cu $m_6(X)$. Notăm $m_3(X) = a_0 + a_1 X + a_2 X^2 + a_3 X^3 + X^4$. Detaliind $m_3(\alpha^3) = 0$, avem

$$a_0 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + a_1 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} + a_2 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} + a_3 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

de unde se poate construi sistemul de ecuaţii

$$a_0 + 1 = 0, \quad a_3 + 1 = 0, \quad a_2 + 1 = 0, \quad a_1 + a_2 + a_3 + 1 = 0,$$

cu soluţia $a_0 = a_1 = a_2 = a_3 = 1$, deci $m_3(X) = m_6(X) = 1 + X + X^2 + X^3 + X^4$.

Rădăcinile lui $m_5(X)$ sunt α^5, α^{10} $(\alpha^{20} = \alpha^5)$, deci $m_5(X)$ este un polinom de gradul 2 : $m_5(X) = b_0 + b_1 X + X^2$. Cum $m_5(\alpha^5) = 0$, se obţine:

$$b_0 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + b_1 \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

care are soluţia $b_0 = b_1 = 1$, *deci* $m_5(X) = 1 + X + X^2$.

Polinomul generator al codului C *este*

$g(X) = cmmmc\{m_1(X), m_2(X), m_3(X), m_4(X), m_5(X), m_6(X)\} = m_1(X)m_3(X)m_5(X) = (1 + X + X^4)(1 + X + X^2 + X^3 + X^4)(1 + X + X^2) = 1 + X + X^2 + X^4 + X^5 + X^8 + X^{10}$.

De remarcat că era suficient să cerem doar ca $\alpha, \alpha^3, \alpha^5$ *să fie rădăcini ale codului.*

Fie polinomul $f(X) = a_0 + a_1X + \ldots + a_{n-1}X^{n-1} \in Z_q[X]$ şi α o rădăcină a sa. Atunci $0 = f(\alpha) = a_0 + a_1\alpha + a_2\alpha^2 + \ldots + a_{n-1}\alpha^{n-1}$, sau - altfel scris:

$$(a_0\ a_1\ a_2\ \ldots\ a_{n-1}) \begin{pmatrix} \mathbf{1} \\ \alpha \\ \vdots \\ \alpha^{n-1} \end{pmatrix} = 0.$$

Deci cuvântul - cod corespunzător polinomului $f(X)$ este în spaţiul nul al matricii

$$(\mathbf{1}\ \alpha\ \ldots\ \alpha^{n-1}) \qquad (1).$$

Aceasta este şi condiţia de divizibilitate a polinomului $f(X)$ cu polinomul minimal $m(X)$ al lui α. Mulţimea polinoamelor care o satisfac formează idealul generat de $m(X)$. Dimensiunea acestui ideal este (Capitolul 5) $n - k$ unde $k = grad(m(X))$. Deci spaţiul liniar generat de matricea (1) are dimensiunea k.

Prin urmare, a cere ca $f(X)$ să admită pe $\alpha_1, \alpha_2, \ldots, \alpha_r \in GF(q^m)$ ca rădăcini este echivalent cu a cere ca vectorul corespunzător să aparţină spaţiului nul al matricii

$$H_{mr,n} = \begin{pmatrix} \mathbf{1} & \alpha_1 & \alpha_1^2 & & \alpha_1^{n-1} \\ \mathbf{1} & \alpha_2 & \alpha_2^2 & & \alpha_2^{n-1} \\ & & \vdots & & \\ \mathbf{1} & \alpha_r & \alpha_r^2 & & \alpha_r^{n-1} \end{pmatrix}$$

Exemplul 7.4 *Să revenim la Exemplul 7.3. Polinomul* $\{f(X)\}$ *aparţine codului respectiv dacă şi numai dacă vectorul corespunzător este în spaţiul nul al matricii*

$$H = \begin{pmatrix} \mathbf{1} & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & & \alpha^{14} \\ \mathbf{1} & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & \mathbf{1} & \alpha^3 & & \alpha^{12} \\ \mathbf{1} & \alpha^5 & \alpha^{10} & \mathbf{1} & \alpha^5 & \alpha^{10} & \mathbf{1} & & \alpha^{10} \end{pmatrix} =$$

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

S-a obţinut o matrice 12×15, *deşi - conform rezultatelor teoretice,* $n = 15$, $n - k = 10$, $k = 5$ *şi deci matricea de control a codului ar trebui să aibă dimensiunile* $(n-k) \times n = 10 \times 15$. *Se observă însă că de fapt ultima linie este nulă, iar următoarele două linii sunt identice. Deci ultimele două linii se pot elimina, ele neoferind nici o informaţie asupra cuvintelor - cod.*

7.4 Coduri ciclice ireductibile

Vom mai prezenta şi o a treia construcţie a codurilor ciclice, realizată de Edwin Berlekamp în 1968 ([10]).

Definiţia 7.7 *Fie polinomul* $Tr(X) = X + X^q + X^{q^2} + \ldots + X^{q^{r-1}} \in GF(q^r)[X]$. *Se numeşte "urma" lui* $\alpha \in GF(q^r)$, *expresia* $Tr(\alpha)$.

Propoziţia 7.7 Tr *este o aplicaţie* : $GF(q^r) \to Z_q$.

Demonstraţie: Trebuie arătat că $\forall \alpha \in GF(q^r)$ avem $Tr(\alpha) \in Z_q$. Pentru aceasta este suficient să arătăm că $Tr(\alpha)$ verifică ecuaţia $X^q - X = 0$. Folosind faptul că că $GF(q^r)$ este corp de caracteristică q şi că $\forall \alpha \in GF(q^r)$ avem $\alpha^{q^r} = \alpha$, se verifică imediat relaţia $[TR(\alpha)]^q = TR(\alpha)$. □

Propoziţia 7.8 Tr *este aplicaţie liniară.*

Demonstraţie: Relaţia $Tr(\alpha + \beta) = Tr(\alpha) + Tr(\beta)$ este uşor de verificat, deoarece se lucrează în corpuri de caracteristică q. □

Propoziţia 7.9 *Pentru orice $a \in Z_q$ există q^{r-1} valori $\alpha \in GF(q^r)$ cu $Tr(\alpha) = a$.*

Demonstraţie: Fiecare ecuaţie $Tr(X) = a$ admite maxim q^{r-1} rădăcini, iar numărul de elemente a posibile este q. Mai trebuie arătat că toate cele q ecuaţii au rădăcini distincte. Aceasta se poate deduce foarte simplu folosind derivata formală într-un corp de caracteristică q: toate ecuaţiile au aceeaşi derivată: 1. □

Teorema 7.11 *Polinomul $Tr(X)$ se poate descompune în $GF(q^r)$ în produs de polinoame minimale.*

Demonstraţie: Demonstraţia se bazează pe următorul algoritm:

(1) Fie polinomul $g(X) = Tr(X)$;

(2) Se consideră $\alpha \in GF(q^r)$ cu $g(\alpha) = 0$. Deci $g(X)$ este divizibil cu polinomul minimal $m(X)$ al lui α.

(3) $g(X) := g(X)/m(X)$. Dacă $g(X) = 1$, STOP, altfel se reia pasul (2).

Existenţa rădăcinilor pentru $g(X)$ (pasul (2)) este asigurată de faptul că în $GF(q^r)$ orice polinom ireductibil este polinom minimal.

Exemplul 7.5 *Să considerăm $GF(2^4)$, deci $q = 2$, $r = 4$. Aici se poate verifica descompunerea*

$$Tr(X) = X + X^2 + X^4 + X^8 = X(1+X)(1+X+X^2)(1+X+X^4),$$

care sunt toate polinoame minimale.

Fie $\alpha \in GF(2^4)$ rădăcină (primitivă) a ecuaţiei $1 + X + X^4 = 0$. Deci toate elementele nenule din $GF(2^4)$ se pot scrie ca puteri ale lui α. Vom folosi aceasta pentru a lista valorile funcţiei $Tr : GF(2^4) \to Z_2$:

X	$Tr(X)$	X	$Tr(X)$	X	$Tr(X)$	X	$Tr(X)$
0	0	α^3	1	α^7	1	α^{11}	1
1	0	α^4	0	α^8	0	α^{12}	1
α	0	α^5	0	α^9	1	α^{13}	1
α^2	0	α^6	1	α^{10}	0	α^{14}	1

După cum se observă, sunt $8 = 2^3$ valori 0 şi 8 valori 1.

Putem da acum teorema principală care defineşte codurile ciclice folosind operatorul Tr:

Teorema 7.12 *Fie n număr natural, q număr prim, k ordinul lui q modulo n ($q^k \equiv 1 \ mod \ n$) şi $\beta \in GF(q^k)$ element primitiv de ordin n. Atunci*

$$C = \{\mathbf{c}_\alpha = (Tr(\alpha), Tr(\alpha\beta), Tr(\alpha\beta^2), \ldots, Tr(\alpha\beta^{n-1})) | \alpha \in GF(q^k)\}$$

este un (n,k) - cod ciclic peste Z_q.

Demonstraţie: Din Propoziţia 7.8 rezultă că C este cod liniar.

Se verifică apoi că $\mathbf{c}_{\alpha\beta^{-1}} \in C$ este o permutare ciclică a lui $\mathbf{c}_\alpha$; deci C este cod ciclic. Deoarece β este primitiv, înseamnă că polinomul său minimal $m(X) = h_0 + h_1X + \ldots + h_kX^k$ are gradul k. Dacă $\mathbf{c}_\alpha = (c_0, c_1, \ldots, c_{n-1})$, avem

$$\sum_{i=0}^{k} c_i h_i = Tr(\alpha \cdot m(\beta)) = Tr(\mathbf{0}) = 0,$$

care constituie una din cele $n-k$ ecuaţii de control pentru C. Deoarece $m(X)$ este minimal (deci ireductibil), $h(X) = X^k m(X^{-1})$ este polinomul de control pentru C. Cum gradul lui este k, avem un (n,k) - cod ciclic peste Z_q. □

De remarcat că această teoremă defineşte numai codurile ciclice "ireductibile" (care nu conţin subcoduri ciclice netriviale proprii), deci clasa lor este inclusă strict în clasa codurilor ciclice dată de cele două definiţii anterioare (prin ideale şi prin rădăcinile polinomului generator).

Exemplul 7.6 *Să considerăm $n = 15$, $q = 2$, deci $k = 4$. În $GF(2^4)$ vom lua α, rădăcină a polinomului $1 + X + X^4$. Se ştie că el este un element primitiv. Aplicaţia Tr este $Tr(X) = X + X^2 + X^4 + X^8$ iar codul C este format din 16 cuvinte de forma*

$$(Tr(\beta), Tr(\beta\alpha), \ldots, Tr(\beta\alpha^{n-1})), \ \forall \beta \in GF(2^4).$$

Pentru $\beta = 0$, $00\ldots0 \in C$;

Pentru $\beta = 1$, $(Tr(1), Tr(\alpha), \ldots, Tr(\alpha^{n-1})) = 000100110101111$.

Celelalte 14 cuvinte sunt permutările circulare ale acestuia.

Polinomul de control al codului este $h(X) = X^4\left(1 + \frac{1}{X} + \frac{1}{X^4}\right) = 1 + X^3 + X^4$, iar polinomul generator

$$g(X) = \frac{X^{16} - X}{X^4 + X^3 + 1} = X + X^4 + X^5 + X^7 + X^9 + X^{10} + X^{11} + X^{12},$$

al cărui vector corespunzător se obţine din construcţia de sus pentru $\beta = \alpha^2$. Acesta este un $(15,4)$ - cod ciclic binar ireductibil.

De remarcat că pentru $n = 15$ sunt numai două astfel de coduri, celălalt fiind codul dual (rădăcina polinomului $1+X^3+X^4$ este de asemenea primitivă).

Exemplul 7.7 *Fie* $q = 2$, $n = 7$ *(va rezulta* $k = 3$*). Aici* $Tr(X) = X + X^2 + X^4$ *şi - deoarece* $X^8 - X = X(X+1)(X^3+X+1)(X^3+X^2+1)$ *- putem construi numai două* $(7,3)$ *- coduri ciclice binare ireductibile de lungime* 7*:*

(1)*: Dacă alegem ca generator pentru* $GF(2^3)$ *pe* α *ca rădăcină a ecuaţiei* $X^3 + X + 1 = 0$*, vom avea*

X	$Tr(X)$	X	$Tr(X)$
0	0	α^3	1
1	1	α^4	0
α	0	α^5	1
α^2	0	α^6	1

Codul ciclic are 8 *cuvinte de forma*

$$Tr(\beta)Tr(\beta\alpha)Tr(\beta\alpha^2)Tr(\beta\alpha^3)Tr(\beta\alpha^4)Tr(\beta\alpha^5)Tr(\beta\alpha^6)$$

unde β *ia toate valorile posibile din* $GF(2^3)$*:*

$\{0000000, 1001011, 0010111, 0101110, 1011100, 0111001, 1110010, 1100101\}$

Se observă că singurul subcod ciclic este cel trivial $\{0000000\}$*, iar ultimele* 7 *cuvinte - cod se obţin unul din altul prin permutări ciclice.*

Polinomul de control este $h(X) = X^3\left(1 + \frac{1}{X} + \frac{1}{X^3}\right) = X^3 + X^2 + 1$*, iar polinomul generator* $g(X) = X(1+X)(1+X+X^3) = X + X^3 + X^4 + X^5$*.*

(2)*: A doua posibilitate este să luăm* α *ca rădăcină a ecuaţiei* $X^3 + X^2 + 1 = 0$*. Atunci*

X	$Tr(X)$	X	$Tr(X)$
0	0	α^3	0
1	1	α^4	1
α	1	α^5	0
α^2	1	α^6	0

iar cele opt cuvinte - cod sunt

$\{0000000, 1110100, 1101001, 1010011, 0100111, 1001110, 0011101, 0111010\}$.

Aici polinomul de control este $h(X) = 1 + X + X^3$ *iar polinomul generator* $g(X) = X + X^2 + X^3 + X^5$*.*

Ceilalţi factori ireductibili ai lui $X^8 - X$ *nu au rădăcini primitive.*

7.5 Exerciţii

7.1 *Demonstraţi Corolarul 7.1.*

7.2 *Demonstraţi Propoziţia 7.6*

7.3 *Construiţi următoarele corpuri:*
(a) $GF(2^2)$;
(b) $GF(2^3)$ *folosind polinomul* $1 + X^2 + X^3$;
(c) $GF(2^4)$ *folosind polinomul* $1 + X^3 + X^4$;
(d) $GF(2^5)$ *folosind polinomul* $1 + X^2 + X^5$.

7.4 *Găsiţi o extensie a lui* Z_2 *în care* $X^9 - 1$ *să se descompună în factori liniari.*

7.5 *Găsiţi toate elementele primitive din* $GF(2^3)$ *şi* $GF(2^4)$.

7.6 *Găsiţi toate elementele primitive din* $GF(3^2)$ *şi* $GF(5^2)$.

7.7 *Construiţi* $GF(2^4)$ *ca o extensie a lui* $GF(2^2)$.

7.8 *Fie* $\alpha \in GF(2^4)$, *rădăcină a polinomului* $1 + X + X^4$.
Găsiţi polinoamele minimale al lui $\beta = \alpha^7$ *şi* $\beta = \alpha^{10}$.

7.9 *Găsiţi polinoamele minimale al tuturor elementelor din:*
- $GF(2^3)$, *generat cu* $1 + X + X^3$;
- $GF(2^5)$, *generat cu* $1 + X^2 + X^5$;
- $GF(3^2)$, *generat cu* $2 + X + X^2$;
- $GF(5^2)$, *generat cu* $3 + 2X + X^2$.

7.10 *Orice element nenul din* $GF(q^m)$ *are un ordin prim cu* q.

7.11 *Găsiţi polinomul generator al unui cod ciclic binar de lungime* 15, *de rădăcini* $1, \alpha^5, \alpha^7$, $\alpha \in GF(2^4)$ *fiind rădăcină a polinomului* $1 + X + X^4$.

Construiţi matricea de control a codului.

Arătaţi că $v(X)$ *este polinom - cod dacă şi numai dacă ponderea* $w(\mathbf{v})$ *este pară.*

7.12 *Găsiţi polinomul generator al unui cod ciclic binar de rădăcini* $\alpha^2, \alpha^3, \alpha^6$ *unde* $\alpha \in GF(2^3)$ *este rădăcină a polinomului* $1 + X + X^3$.

7.13 *Să se descompună* $Tr(X)$ *în produs de polinoame minimale în corpurile:*

(a) $GF(2^5)$; (b) $GF(3^2)$; (c) $GF(3^3)$; (d) $GF(5^2)$.

7.14 *Folosind Teorema 7.12 să se construiască toate codurile ireductibile pentru* $GF(2^3)$, $GF(2^5)$, $GF(3^2)$.

7.15 *Verificaţi că polinomul* $h(X)$ *construit în demonstraţia Teoremei 7.12 este polinomul generator al codului dual.*

Capitolul 8

Coduri BCH

Codurile BCH constituie cea mai utilizată clasă de coduri ciclice. Ele au fost definite în mod independent de Bose şi Chaudhuri pe de-o o parte, Hocquenheim pe de-altă parte. Numele de BCH (**B**ose - **C**haudhuri - **H**ocquenheim) a fost dat de Peterson, care s-a ocupat de ele în mod special, construind şi un algoritm de decodificare.

8.1 Definirea codurilor BCH

Definiţia 8.1 *Fie q număr prim, m_0, m, d numere naturale şi $\alpha \in GF(q^m)$. Un cod ciclic este cod BCH dacă este definit de rădăcinile polinomului generator*

$$\alpha^{m_0}, \alpha^{m_0+1}, \ldots, \alpha^{m_0+d-2}.$$

În majoritatea cazurilor studiate se consideră $m_0 = 0$ sau $m_0 = 1$.

Teorema 8.1 *Într-un cod BCH, lungimea n a cuvintelor - cod este egală cu ordinul lui α.*

Demonstraţie: Fie $t = ord(\alpha)$. Conform construcţiei codurilor ciclice bazată pe rădăcinile polinomului generator (Capitolul 7, Secţiunea 7.3), lungimea n a cuvintelor - cod este cel mai mic multiplu comun al ordinelor rădăcinilor. În cazul codurilor BCH avem

$$(\alpha^{m_0})^n = \alpha^{m_0 n} = 1 \text{ şi } 1 = (\alpha^{m_0+1})^n = \alpha^{m_0+n} = \alpha^{m_0}\alpha^n,$$

deci $\alpha^n = 1$. Prin urmare $t|n$ şi $n \geq t$.

De asemenea, $(\alpha^j)^t = (\alpha^t)^j = 1 \ \forall j = m_0, m_0+1, \ldots, m_0+d-2$, adică $n \leq t$.

Rezultă $n = t$. □

De remarcat că, dacă α este primitiv, atunci $t = q^m - 1$.

Teorema 8.2 *Într-un cod BCH definit de rădăcinile* $\alpha^{m_0}, \alpha^{m_0+1}, \ldots, \alpha^{m_0+d-2}$ $(\alpha \in GF(q^m))$, *distanţa minimă a codului este cel puţin* d *(şi deci codul poate corecta orice combinaţie de* $t \leq \left[\frac{d-1}{2}\right]$ *erori).*

Demonstraţie: Un cod BCH este spaţiul nul al matricii

$$H = \begin{pmatrix} 1 & \alpha^{m_0} & (\alpha^{m_0})^2 & \ldots & (\alpha^{m_0})^{n-1} \\ 1 & \alpha^{m_0+1} & (\alpha^{m_0+1})^2 & \ldots & (\alpha^{m_0+1})^{n-1} \\ & & \vdots & & \\ 1 & \alpha^{m_0+d-2} & (\alpha^{m_0+d-2})^2 & \ldots & (\alpha^{m_0+d-2})^{n-1} \end{pmatrix}$$

Să considerăm următorul determinant, obţinut prin alegerea arbitrară a $d-1$ coloane din H:

$$D = \begin{vmatrix} (\alpha^{m_0})^{j_1} & (\alpha^{m_0})^{j_2} & \ldots & (\alpha^{m_0})^{j_{d-1}} \\ (\alpha^{m_0+1})^{j_1} & (\alpha^{m_0+1})^{j_2} & \ldots & (\alpha^{m_0+1})^{j_{d-1}} \\ & & & \\ (\alpha^{m_0+d-2})^{j_1} & (\alpha^{m_0+d-2})^{j_2} & \ldots & (\alpha^{m_0+d-2})^{j_{d-1}} \end{vmatrix} =$$

$$= \alpha^{m_0(j_1+j_2+\ldots+j_{d-1})} \begin{vmatrix} 1 & 1 & \ldots & 1 \\ \alpha^{j_1} & \alpha^{j_2} & \ldots & \alpha^{j_{d-1}} \\ & & & \\ (\alpha^{j_1})^{d-2} & (\alpha^{j_2})^{d-2} & \ldots & (\alpha^{j_{d-1}})^{d-2} \end{vmatrix} =$$

$$= \alpha^{m_0(j_1+j_2+\ldots+j_{d-1})} \prod_{i>k} (\alpha^{j_i} - \alpha^{j_k}) \neq 0$$

pentru că $j_i \neq j_k$ dacă $i \neq k$.

Deci orice submulţime formată cu maxim $d-1$ coloane din H este liniar independentă de unde – conform Teoremei 2.4 (Capitolul 2) distanţa minimă a codului este cel puţin d. □

Exemplul 8.1 *Dacă se lucrează în binar, puterile pare ale rădăcinilor se pot omite din listă(dacă* α^i *este rădăcină, atunci şi* α^{2i} *este rădăcină – conform Teoremei* 7.9, *Capitolul* 7*). Atunci un cod BCH binar corector de* 3 *erori este definit de rădăcinile* β, β^3, β^5, *unde* β *este un element primitiv dintr-o extensie a lui* Z_2. *De exemplu, dacă* $\beta = \alpha$, *rădăcină a polinomului* $1 + X + X^4$, *se obţine codul din Exemplul* 7.3 *(Capitolul* 7*).*

Exemplul 8.2 *Fie $GF(3^2)$ generat de rădăcina α a polinomului ireductibil $g(X) = 2 + X + X^2$. Deoarece $ord(\alpha) = 8$, orice cod generat va avea lungimea $n = 8$. Să construim codul BCH de rădăcini $\alpha^2, \alpha^3, \alpha^4$. Polinoamele minimale sunt:*

$m_2(X) = 1 + X^2$ cu rădăcini α^2, α^6;

$m_3(X) = 2 + X + X^2$ cu rădăcini α^3, α;

$m_4(X) = 1 + X$, numai cu rădăcina α^4.

Deci polinomul generator este

$$g(X) = (1 + X^2)(2 + X + X^2)(1 + X) = 2 + X^2 + X^3 + 2X^4 + X^5.$$

Codul este un $(8, 3)$ - cod ciclic. Deoarece printre rădăcinile sale este şi α, se poate considera $m_0 = 1$, $d = 5$, deci codul poate corecta cel puţin două erori.

De remarcat că dacă folosim acest cod drept cod liniar, cum ponderea minimă a cuvintelor sale este 4 (de exemplu 02010201 este cuvânt - cod), distanţa minimă este 4, deci codul nu poate corecta decât o singură eroare. În general, tratarea codurilor ciclice sub formă de coduri liniare scade de obicei puterea de corecţie a erorilor independente.

Numerele m, m_0, d, q formează parametrii codului BCH. Ideea generală este de a-l mări pe d, pentru a putea corecta câte mai multe combinaţii de erori independente. Acest lucru este adesea posibil, m şi q fiind alese arbitrar.

Să observăm că pentru $\alpha \in GF(q^m)$ primitiv, toate puterile lui α până la $q^m - 1$ sunt distincte. În cazul binar, avem următorul rezultat:

Teorema 8.3 *Fie m un număr natural oarecare şi $\alpha \in GF(2^m)$ primitiv. Atunci există un cod BCH de lungime $n = 2^m - 1$ care corectează orice combinaţie de $t \leq 2^{m-1} - 1$ erori independente, folosind cel mult mt poziţii de control.*

Demonstraţie: Fie codul binar definit de rădăcinile $\alpha, \alpha^2, \ldots, \alpha^{2t}$. Acesta este un cod BCH pentru $q = 2$, $m_0 = 1$, $d = 2t + 1$, $n = 2^m - 1$, cu distanţa minimă cel puţin $2t + 1$. Pentru ca rădăcinile menţionate mai sus să fie distincte, trebuie ca $2t \leq n - 1 = 2^m - 2$.

Să observăm că orice putere pară a lui α este rădăcină a unui polinom minimal corespunzător unei puteri impare anterioare a lui α. Deci este suficient să dăm la început doar rădăcinile $\alpha, \alpha^3, \ldots, \alpha^{2t-1}$ ale cuvintelor codului binar. Avem - după definiţie (Capitolul 7):

$$g(X) = cmmmc\{m_1(X), m_3(X), \ldots, m_{2t-1}(X)\}.$$

Pe de - altă parte, $m_i(X) | X^n - 1 = X^{2^m - 1} - 1$ şi deci - conform Teoremei 7.7 (Capitolul 7), $grad(m_i(X)) \leq m$. Din cele două relaţii

rezultă $grad(g(X)) \leq mt$. Cum $grad(g(X))$ dă numărul de simboluri de control, teorema este demonstrată. □

Exemplul 8.3 *Să considerăm codul ciclic definit în Exemplul* 7.3. *Construcţia sa este aceea a unui cod BCH binar cu* $n = 15$ *şi* $m = 4$. *El are* $2t = 6$ *adică* $t = 3$, *deci poate corecta orice combinaţie de maxim* 3 *erori independente, folosind* $m \cdot t = 12$ *poziţii de control.*

De fapt, observaţiile de la Exemplul 7.3 *arată că numărul acestora poate fi redus la* 10, *ceea ce conduce la un* $(15, 5)$ *- cod BCH binar.*

8.2 Algoritmul Peterson de decodificare

Fie codul BCH definit de rădăcinile $\alpha^{m_0}, \alpha^{m_0+1}, \ldots, \alpha^{m_0+2t-1}$ $(d = 2t + 1)$, cu $\alpha \in GF(q^m)$.

Un asemenea cod corectează orice combinaţie de maxim t erori independente. Fie $\{f(X)\}$ un cuvânt - cod transmis şi

$$\{r(X)\} = \{f(X) + e(X)\} = \{f(X)\} + \{e(X)\}$$

secvenţa recepţionată, unde $\{e(X)\}$ este vectorul - eroare.

Avem, pentru $j = m_0, m_0 + 1, \ldots, m_0 + 2t - 1$:

$$S_j = r(\alpha^j) = f(\alpha^j) + e(\alpha^j) = e(\alpha^j).$$

S_j se numesc *componentele sindromului.*

La decodificare, eroarea - tip **e** este determinată complet dacă se ştiu:

- Valorile componentelor sale nenule $Y_i \in Z_q$;
- Localizările acestora $X_i \in GF(q^m)$ $(0 \leq i \leq w(\mathbf{e}))$.

Un cuvânt din $GF(q^m)$ este un vector cu q^m componente; vom nota poziţiile acestor componente folosind puterile lui α: pe poziţia i se află coeficientul lui α^{i-1} $(1 \leq i \leq q^m)$, deci α^{i-1} va localiza a i - a componentă dintr-un cuvânt.

Deci, dacă au intervenit t erori, eroarea - tip este complet caracterizată de cunoaşterea celor t perechi nenule (Y_i, X_i) $1 \leq i \leq t$.

Putem conveni (evident) că

$$Y_i = 0 \qquad \Longleftrightarrow \qquad X_i = 0.$$

De remarcat că X_i nu reprezintă α^i ci un α^j arbitrar, care va trebui aflat.

Deci, a găsi pe $\{e(X)\}$ revine la a determina $2t$ elemente X_i, Y_i $(1 \leq i \leq t)$, grupate în t perechi. Vom considera $Y_i = 0$ $\forall i$ $(t < i \leq q^m - 1)$. Avem - conform notaţiei polinomiale

$$S_j = e(\alpha^j) = \sum_{i=1}^{t} Y_i X_i^j, \qquad m_0 \leq j \leq m_0 + 2t - 1.$$

Atunci

$$(S_j)^q = \left(\sum_{i=1}^{t} Y_i X_i^j\right)^q = \sum_{i=1}^{t} Y_i^q X_i^{jq} = \sum_{i=1}^{t} Y_i X_i^{jq} = S_{jq}.$$

Deci, determinarea polinomului - eroare $\{e(X)\}$ se reduce la aflarea soluţiilor (Y_i, X_i), $1 \leq i \leq t$ ale sistemului de ecuaţii

$$\sum_{i=1}^{t} Y_i X_i^j = S_j,\ m_0 \leq j \leq m_0 + 2t - 1. \tag{1}$$

Dacă au intervenit k $(k < t)$ erori, se completează cele k perechi nenule (Y_i, X_i) cu alte $t - k$ perechi nule $(0, 0)$. De fapt, acest număr k nu este cunoscut apriori; de aceea, se pleacă de la t erori potenţiale (maximul pe care codul BCH îl poate corecta în mod cert), caracterizate de t perechi (Y_i, X_i). Problema care se pune este deci de a-l determina pe k şi cele $2k$ necunoscute Y_i, X_i $(1 \leq i \leq k)$.

Orice metodă de corectare a erorilor revine la rezolvarea sistemului de ecuaţii (1), care este un sistem neliniar de $2t$ ecuaţii cu $2t$ necunoscute, problemă NP - completă. Să încercăm o abordare mai simplă a acestei probleme.

Fie polinomul

$(X_1 - X)(X_2 - X)\ldots(X_t - X) = \sigma_t + \sigma_{t-1}(-X) + \ldots + \sigma_1(-X)^{t-1} + (-X)^t$,

unde σ_j sunt funcţiile simetrice date de relaţiile lui *Viète*. Dacă în această relaţie se ia $X = X_i$ $(i = 1, \ldots, t)$, obţinem

$X_i^t + (-1)\sigma_1 X_i^{t-1} + \ldots + (-1)^{t-1}\sigma_{t-1} X_i + (-1)^t \sigma_t = 0$ $(1 \leq i \leq t)$.

Înmulţim cu $Y_i X_i^j$ şi sumăm după i. Se obţine:

$$\sum_{i=1}^{t} Y_i X_i^{j+t} + (-1)\sigma_1 \sum_{i=1}^{t} Y_i X_i^{j+t-1} + \ldots + (-1)^{t-1}\sigma_{t-1} \sum_{i=1}^{t} Y_i X_i^{j+1} +$$

$$(-1)^t \sigma_t \sum_{i=1}^{t} Y_i X_i^j = 0, \quad j = m_0, m_0 + 1, \ldots, m_0 + t - 1,$$

relaţie care se poate scrie

$$S_j(-1)^t\sigma_t + S_{j+1}(-1)^{t-1}\sigma_{t-1} + \ldots + S_{j+t-1}(-1)\sigma_1 + S_{j+t} = 0,$$
$$m_0 \leq j \leq m_0 + t - 1. \tag{2}$$

Algoritmul lui Peterson de corectare a k ($k \leq t$) erori într-un cod BCH este:

1. Se rezolvă sistemul (2) în care necunoscutele sunt $(-1)^p\sigma_p$, $1 \leq p \leq t$;
2. Se înlocuiesc valorile aflate în ecuaţia
   $X^t + (-1)\sigma_1 X^{t-1} + \ldots + (-1)^{t-1}\sigma_{t-1}X + (-1)^t\sigma_t = 0$
   şi se află cele t rădăcini $X_1, X_2, \ldots, X_t \in GF(q^m)$ ale sale;
3. Cu aceste valori introduse în sistemul (1), se determină din primele t ecuaţii valorile $Y_1, Y_2, \ldots, Y_t \in Z_q$;
4. Dacă (Y_i, α^{j_i}) $(1 \leq i \leq k)$ sunt cele $k \leq t$ valori nenule ale soluţiei obţinute, atunci:
   - Pentru $k = 0$, cuvântul recepţionat este $\{r(X)\}$ (fără erori);
   - Pentru $0 < k \leq t$, fie $e(X) = \sum_{i=1}^{k} Y_i X^{j_i}$.

   Cuvântul - cod decodificat este $\{f(X)\} = \{r(X)\} - \{e(X)\}$.

De remarcat că în cazul binar, pasul (3) nu mai este necesar, deoarece

$$Y_i = 1, \quad \forall i \leq k.$$

Pentru corectitudinea algoritmului, trebuie studiate compatibilităţile sistemelor de ecuaţii (1) şi (2).

Teorema 8.4 *Sistemul* $\sum_{i=1}^{t} Y_i X_i^j = S_j$ $(m_0 \leq j \leq m_0 + t - 1)$, *de necunoscute* Y_i, *este compatibil dacă şi numai dacă au intervenit* t *erori.*

Demonstraţie: Considerat ca sistem în Y, sistemul este liniar; determinantul său principal este

$$\begin{vmatrix} X_1^{m_0} & X_2^{m_0} & & X_t^{m_0} \\ X_1^{m_0+1} & X_2^{m_0+1} & & X_t^{m_0+1} \\ X_1^{m_0+2} & X_2^{m_0+2} & & X_t^{m_0+2} \\ & & & \\ X_1^{m_0+t-1} & X_2^{m_0+t-1} & & X_t^{m_0+t-1} \end{vmatrix} =$$

$$X_1^{m_0} X_2^{m_0} \dots X_t^{m_0} \begin{vmatrix} 1 & 1 & & 1 \\ X_1 & X_2 & & X_t \\ X_1^2 & X_2^2 & & X_t^2 \\ & & & \\ X_1^{t-1} & X_2^{t-1} & \dots & X_t^{t-1} \end{vmatrix}$$

$$= X_1^{m_0} X_2^{m_0} \dots X_t^{m_0} \prod_{1 \le j < i \le t} (X_i - X_j).$$

Acesta este nenul numai dacă au intervenit t erori, pentru că atunci toţi X_i sunt distincţi şi nenuli. □

Teorema 8.5 *Matricea* $M = \begin{pmatrix} S_{m_0} & S_{m_0+1} & & S_{m_0+t-1} \\ S_{m_0+1} & S_{m_0+2} & \dots & S_{m_0+t} \\ & & \vdots & \\ S_{m_0+t-1} & S_{m_0+t} & \dots & S_{m_0+2t-1} \end{pmatrix}$

este nesingulară dacă au intervenit t erori şi este singulară dacă au intervenit mai puţin de t erori.

Demonstraţie: Ţinând cont de forma sistemului (1), este posibilă descompunerea matricii M în produs de 3 matrici $M = ABA^T$ unde:

$$A = \begin{pmatrix} 1 & 1 & & 1 \\ X_1 & X_2 & & X_t \\ X_1^2 & X_2^2 & & X_t^2 \\ & & & \\ X_1^{t-1} & X_2^{t-1} & & X_t^{t-1} \end{pmatrix} \quad B = \begin{pmatrix} Y_1 X_1^m & 0 & & 0 \\ 0 & Y_2 X_2^m & & 0 \\ & & & \\ 0 & 0 & & Y_t X_t^m \end{pmatrix}$$

M este nesingulară dacă cele două matrici din descompunere sunt nesingulare. Matricea A este nesingulară dacă şi numai dacă valorile X_i $(1 \le i \le t)$ sunt distincte şi nenule, deci dacă au intervenit t erori. Matricea B este nesingulară dacă toate elementele de pe diagonala sa principală sunt nenule, deci dacă toate perechile (Y_i, X_i) $1 \le i \le t$ sunt nenule – adică au apărut t erori. □

Aplicarea algoritmului Peterson se face în felul următor:

1. Folosind polinomul recepţionat $\{r(X)\}$, se determină
$$S_j = r(\alpha^j), \quad j = m_0, m_0+1, \ldots, m_0+2t-1.$$

2. Se studiază compatibilitatea sistemului
$$S_j(-1)^t\sigma_t + S_{j+1}(-1)^{t-1}\sigma_{t-1} + \ldots + S_{j+t-1}(-1)\sigma_1 + S_{j+t} = 0,$$
$$m_0 \leq j \leq m_0 + t - 1.$$
Dacă matricea M a acestui sistem este nesingulară, înseamnă că au apărut t erori. Dacă M este singulară, atunci se observă că $\sigma_t = 0$ (deoarece $\sigma_t = X_1X_2\ldots X_t$) şi - după ce se înlocuieşte $\sigma_t = 0$ - se obţine un nou sistem liniar format din primele $t-1$ ecuaţii, cu primele $t-1$ necunoscute.

Se studiază matricea acestui nou sistem. Dacă ea este nesingulară, înseamnă că au apărut $t-1$ erori. Dacă şi aceasta este singulară, cum $\sigma_{t-1} = 0$, se repetă procedeul.

Primul sistem compatibil (cu matricea nesingulară) care se obţine va da numărul $k \leq t$ de erori care au intervenit în transmisie, precum şi valorile
$$(-1)\sigma_1,\ (-1)^2\sigma_2, \ldots, (-1)^k\sigma_k.$$

3. Se introduc valorile determinate mai sus în polinomul
$P(X) = X^k + (-1)\sigma_1X^{k-1} + \ldots + (-1)^{k-1}\sigma_{k-1}X + (-1)^k\sigma_k$
şi se află cele k rădăcini $X_1, X_2, \ldots, X_k \in GF(q^m)$ ale ecuaţiei $P(X) = 0$ (prin diverse metode, eventual chiar prin simpla verificare a ecuaţiei cu toate elementele din $GF(q^m)$). Aceste rădăcini vor da localizările erorilor.

De remarcat că ordinea de notare a rădăcinilor $X_1, X_2, \ldots, X_k$ este neesenţială.

4. Se introduc aceste soluţii în ecuaţiile sistemului
$$\sum_{i=1}^{k} Y_iX_i^j = S_j, \quad j = m_0, m_0+1, \ldots, m_0+k-1$$
şi se determină valorile erorilor $Y_1, Y_2, \ldots, Y_k$.

5. Se calculează astfel polinomul eroare $e(X) = \sum_{i=1}^{k} Y_iX^{j_i}$, unde $X_i = \alpha^{j_i}$ $(i = 1, \ldots, k)$. Acesta se scade din polinomul recepţionat şi se obţine cuvântul - cod transmis:
$$\{r(X)\} - \{e(X)\} = \{r(X) - e(X)\} = \{f(X)\}.$$

Exemplul 8.4 *Fie codul BCH definit de rădăcinile* $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5\alpha^6$ *unde* $\alpha \in GF(2^4)$ *este element primitiv, rădăcină a polinomului* $1 + X + X^4$ *(Exemplul 7.3, Capitolul 7). Am arătat (Exemplul 8.3) că acest cod este capabil să corecteze maxim 3 erori independente.*

Să presupunem că au apărut două erori, pe poziţiile α^3 *şi* α^{10} *(adică pe poziţiile 4 şi 11) ale cuvântului transmis. Ele sunt reprezentate prin polinomul eroare* $e(X) = X^3 + X^{10}$.

Să calculăm sindromul (pentru operaţile folosite, a se vedea Exemplele 4.8 şi 4.9, Capitolul 4):

$$\begin{aligned}
&S_1 = r(\alpha) = e(\alpha) = \alpha^3 + \alpha^{10} = \alpha^{12};\\
&S_2 = (S_1)^2 = \alpha^{24} = \alpha^9;\\
&S_3 = r(\alpha^3) = e(\alpha^3) = \alpha^9 + \alpha^{30} = \alpha^9 + \alpha^0 = \alpha^7;\\
&S_4 = (S_2)^2 = \alpha^{18} = \alpha^3;\\
&S_5 = r(\alpha^5) = e(\alpha^5) = \alpha^{15} + \alpha^{50} = \alpha^0 + \alpha^5 = \alpha^{10};\\
&S_6 = (S_3)^2 = \alpha^{14}.
\end{aligned}$$

Observăm că într-o situaţie reală, nu se va şti unde sunt localizate aceste erori şi deci - în cazul în care au intervenit două erori pe poziţiile α^3 *şi* α^{10} *(care nu se cunosc încă) - din vectorul recepţionat se obţine direct sindromul*

$$S = (S_1, S_2, S_3, S_4, S_5, S_6) = (\alpha^{12}, \alpha^9, \alpha^7, \alpha^3, \alpha^{10}, \alpha^{14}),$$

fără să ştim din combinaţiile căror localizări ale erorilor provin aceste componente ale sindromului.

Să considerăm sistemul de ecuaţii

$$\begin{cases}
S_1\sigma_3 + S_2\sigma_2 + S_3\sigma_1 &= S_4\\
S_2\sigma_3 + S_3\sigma_2 + S_4\sigma_1 &= S_5\\
S_2\sigma_3 + S_4\sigma_2 + S_5\sigma_1 &= S_6
\end{cases}$$

adică

$$\begin{cases}
\alpha^{12}\sigma_3 + \alpha^9\sigma_2 + \alpha^7\sigma_1 &= \alpha^3\\
\alpha^9\sigma_3 + \alpha^7\sigma_2 + \alpha^7\sigma_1 &= \alpha^{10}\\
\alpha^7\sigma_3 + \alpha^3\sigma_2 + \alpha^{10}\sigma_1 &= \alpha^{14}
\end{cases}$$

Înmulţim cele trei ecuaţii ale sistemului respectiv cu

$$\alpha^{-7} = \alpha^8,\ \alpha^{-3} = \alpha^{12},\ \alpha^{-10} = \alpha^5$$

şi obţinem:

$$\begin{cases}
\alpha^5\sigma_3 + \alpha^2\sigma_2 + \sigma_1 &= \alpha^{11}\\
\alpha^6\sigma_3 + \alpha^4\sigma_2 + \sigma_1 &= \alpha^7\\
\alpha^{12}\sigma_3 + \alpha^8\sigma_2 + \sigma_1 &= \alpha^4
\end{cases}$$

Adunăm prima ecuaţie la ultimele două şi atunci

$$\begin{cases} \alpha^9\sigma_3 + \alpha^{10}\sigma_2 &= \alpha^8 \\ \alpha^{14}\sigma_3 + \alpha^0\sigma_2 &= \alpha^{13} \end{cases}$$

Deoarece a doua ecuaţie se obţine din prima prin înmulţire cu α^5, *ele nu sunt independente; deci au apărut mai puţin de* 3 *erori. Facem* $\sigma_3 = 0$ *şi a doua ecuaţie va da* $\sigma_2 = \alpha^{13}$. *Din prima ecuaţie a sistemului se ajunge la* $\alpha^0 + \sigma_1 = \alpha^{11}$, *adică* $\sigma_1 = \alpha^{12}$.

Deci, în acest caz $k = 2$, $\sigma_1 = \alpha^{12}$, $\sigma_2 = \alpha^{13}$. *Se ajunge la ecuaţia* $X^2 + \sigma_1 X + \sigma_2 = X^2 + \alpha^{12}X + \alpha^{13} = 0$, *care are ca rădăcini* $X_1 = \alpha^3$, $X_2 = \alpha^{10}$.

Am redescoperit astfel cele două localizări ale erorilor. Codul fiind binar, corectarea se face imediat: se modifică (din 1 *în* 0 *sau din* 0 *în* 1*) biţii de pe poziţiile* 4 *respectiv* 11 *din cuvântul recepţionat.*

Exemplul 8.5 *Fie codul BCH definit de rădăcinile* $\alpha, \alpha^2, \alpha^3, \alpha^4$ *unde* $\alpha \in GF(2^4)$ *este rădăcina polinomului* $1 + X + X^4$. *Acest cod are* $n = 15$, $m_0 = 1$, $m_0 + 2t - 1 = 4$, *deci* $t = 2$; *codul poate corecta în mod sigur cel puţin două erori independente. Să presupunem că s-a recepţionat cuvântul*

$$\mathbf{r} = 100100110000100.$$

Să determinăm cuvântul - cod transmis.

Polinomul corespunzător este $r(X) = 1 + X^3 + X^6 + X^7 + X^{12}$. *Componentele sindromului sunt*

$$S_1 = r(\alpha) = \alpha^0, \qquad S_2 = r(\alpha^2) = (S_1)^2 = \alpha^0,$$
$$S_3 = r(\alpha^3) = \alpha^4, \qquad S_4 = r(\alpha^4) = (S_2)^2 = \alpha^0.$$

Trebuie rezolvat sistemul

$$\begin{cases} S_1\sigma_2 + S_2\sigma_1 &= S_3 \\ S_2\sigma_2 + S_3\sigma_1 &= S_4 \end{cases} \quad \text{adică} \quad \begin{cases} \alpha^0\sigma_2 + \alpha^0\sigma_1 &= \alpha^4 \\ \alpha^0\sigma_2 + \alpha^4\sigma_1 &= \alpha^0 \end{cases},$$

de unde se obţine $\sigma_1 = \alpha^0$, $\sigma_2 = \alpha$.

Polinomul de localizare a erorilor este $X^2 + \sigma_1 X + \sigma_2 = X^2 + X + \alpha$, *cu rădăcinile* $X_1 = \alpha^7$, $X_2 = \alpha^9$. *Au apărut deci două erori, pe poziţiile* 8 *şi* 10. *Pentru că suntem în cazul binar,* $Y_1 = Y_2 = 1$; *vectorul - eroare obţinut este* $e(X) = X^7 + X^9$, *deci cuvântul - cod transmis este*

$$\{f(X)\} = \{r(X)\} - \{e(X)\} = \{r(X) + e(X)\} = 1 + X^3 + X^6 + X^9 + X^{12},$$

adică $\mathbf{f} = 100100100100100$.

8.3 Decodificarea codurilor BCH binare

După cum s-a observat, în cazul codurilor BCH binare este suficient să se localizeze eroarea; corectarea se face imediat prin complementarea biţilor aflaţi pe poziţiile respective. După determinarea valorilor $\sigma_1, \sigma_2, \ldots, \sigma_t$, algoritmul Peterson nu explicitează modul de rezolvare a ecuaţiei de localizare al erorii, care uneori poate fi dificil de prelucrat.

Un procedeu eficient de determinare a poziţiilor erorilor pentru codurile BCH binare a fost realizat de Chien.

Fie polinomul (de localizare a erorii):

$$S(X) = (1-X_1X)(1-X_2X)\ldots(1-X_tX) = \sigma_0+\sigma_1X+\sigma_2X^2+\ldots+\sigma_tX^t$$

unde

$$\begin{aligned}\sigma_0 &= 1\\ \sigma_1 &= X_1 + X_2 + \ldots + X_t\\ &\vdots\\ \sigma_t &= X_1X_2\ldots X_t\end{aligned}$$

(operaţiile fiind efectuate în binar, semnul $-$ este asimilat cu $+$).

Rădăcinile acestui polinom sunt inversele localizărilor erorilor X_i^{-1} $(1 \leq i \leq t)$.

Procedeul lui Chien de corectare a erorilor pentru codurile BCH binare, este următorul:

Fie $\mathbf{r} = r_0r_1\ldots r_{n-1}$ cuvântul recepţionat; componentele lui se vor decodifica pas - cu - pas (pe măsură ce sunt recepţionate).

Să considerăm primul bit primit r_{n-1} şi să cercetăm dacă el este corect sau nu. Aceasta este echivalent cu a verifica dacă α^{n-1} este o localizare a erorii, sau dacă $\alpha^{-(n-1)} = \alpha$ este o rădăcină a polinomului $S(X)$ de localizare a erorii.

Dacă $\sigma_1\alpha + \sigma_2\alpha^2 + \ldots + \sigma_t\alpha^t = \mathbf{1}$ adică $S(\alpha) = \mathbf{0}$, atunci a intervenit o eroare pe poziţia α^{n-1} şi deci componenta r_{n-1} trebuie corectată.

Dacă $\sigma_1\alpha + \sigma_2\alpha^2 + \ldots + \sigma_t\alpha^t \neq \mathbf{1}$ adică $S(\alpha) \neq \mathbf{0}$, atunci r_{n-1} s-a recepţionat corect.

Procedeul se repetă: în general componenta recepţionată r_{n-s} este corectă dacă $\alpha^{-(n-s)} = \alpha^s$ nu este rădăcină a polinomului $S(X)$, adică

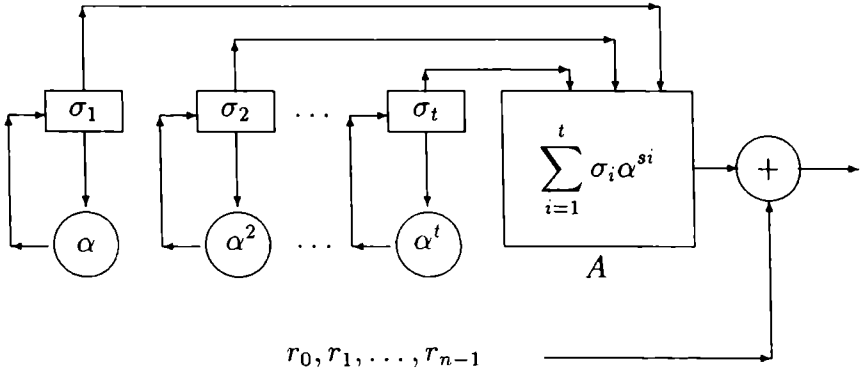
$$\sigma_1\alpha^s + \sigma_2\alpha^{2s} + \ldots + \sigma_t\alpha^{ts} \neq \mathbf{1}.$$

În caz contrar, r_{n-s} va fi corectată în $r_{n-s} + 1 \ (mod\ 2)$.

Algoritmul lui Chien este deci:

1. $s := 0$;
2. **while** $s \leq n$ **do**
   (a) $s := s + 1$;
   (b) Se calculează $P := \sigma_1\alpha^s + \sigma_2\alpha^{2s} + \ldots + \sigma_t\alpha^{ts}$;
   (c) Se decodifică $a_{n-s} := r_{n-s} + v$ unde
$$v = \begin{cases} 0 & \text{dacă } P \neq \mathbf{1} \\ 1 & \text{dacă } P = \mathbf{1} \end{cases}$$

Circuitul liniar care implementează acest algoritm este:

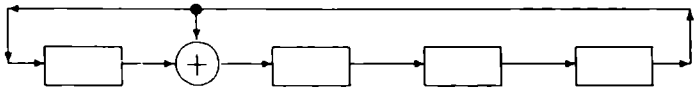


Cele t circuite din stânga realizează la fiecare tact multiplicarea cu α^i.

La momentul iniţial blocul A calculează suma $\sigma_1\alpha + \sigma_2\alpha^2 + \ldots + \sigma_t\alpha^t$ dând la ieşire valoarea 1 (scalar) dacă această sumă este egală cu $\mathbf{1} = (1, 0, \ldots, 0)$ şi 0 (scalar) dacă suma anterioară dă orice element din $GF(2^m)$ diferit de $\mathbf{1}$. În funcţie de acest rezultat, componenta r_{n-1} este schimbată, respectiv păstrată.

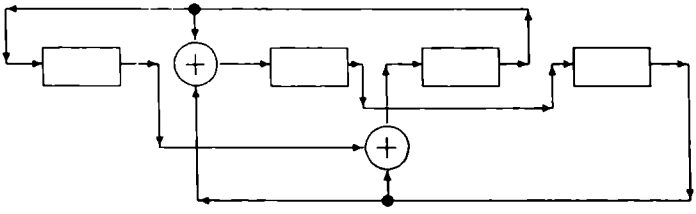
La momentul următor, blocul A calculează suma $\sigma_1\alpha^2 + \sigma_2\alpha^4 + \ldots + \sigma_t\alpha^{2t}$. Dacă această sumă este $\mathbf{1}$, α^2 este rădăcină a polinomului de localizare a erorii $S(X)$, deci $\alpha^{-2} = \alpha^{n-2}$ este o localizare a erorii. În caz contrar, suma este diferită de $\mathbf{1}$. La ieşirea din blocul A se obţine scalarul 1 respectiv 0 – componenta r_{n-2} fiind modificată respectiv păstrată.

Exemplul 8.6 *Fie $\alpha \in GF(2^4)$ primitiv, rădăcina polinomului $1 + X + X^4$. Să presupunem că se ştiu σ_1 şi σ_2. Circuitul care multiplică cu α este (Capitolul 4, Exemplul 4.8) circuitul de împărţire cu $m(X) = 1 + X + X^4$:*



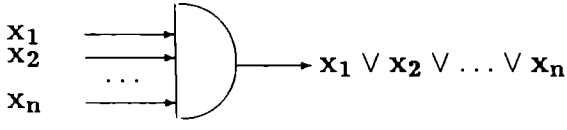
Circuitul de înmulţire cu α^2 *este - conform egalităţii*

$\alpha^2(a_1 + a_2\alpha + a_3\alpha^2 + a_4\alpha^3) = a_1\alpha^2 + a_2\alpha^3 + a_3\alpha^4 + a_4\alpha^5 = a_3 + (a_3 + a_4)\alpha + (a_1 + a_4)\alpha^2 + a_2\alpha^3$ *următorul (vezi şi Exemplul* 4.10, *Capitolul* 4*):*



Iniţial în elementele de înmagazinare ale celor două circuite se află σ_1 *respectiv* σ_2.

Pentru cazul binar vom nota cu



dispozitivul "sau" care dă valoarea (scalar) 0 *dacă şi numai dacă*

$$\mathbf{x_1} = \mathbf{x_2} = \ldots = \mathbf{x_n} = \mathbf{0},$$

şi cu

dispozitivul "not".

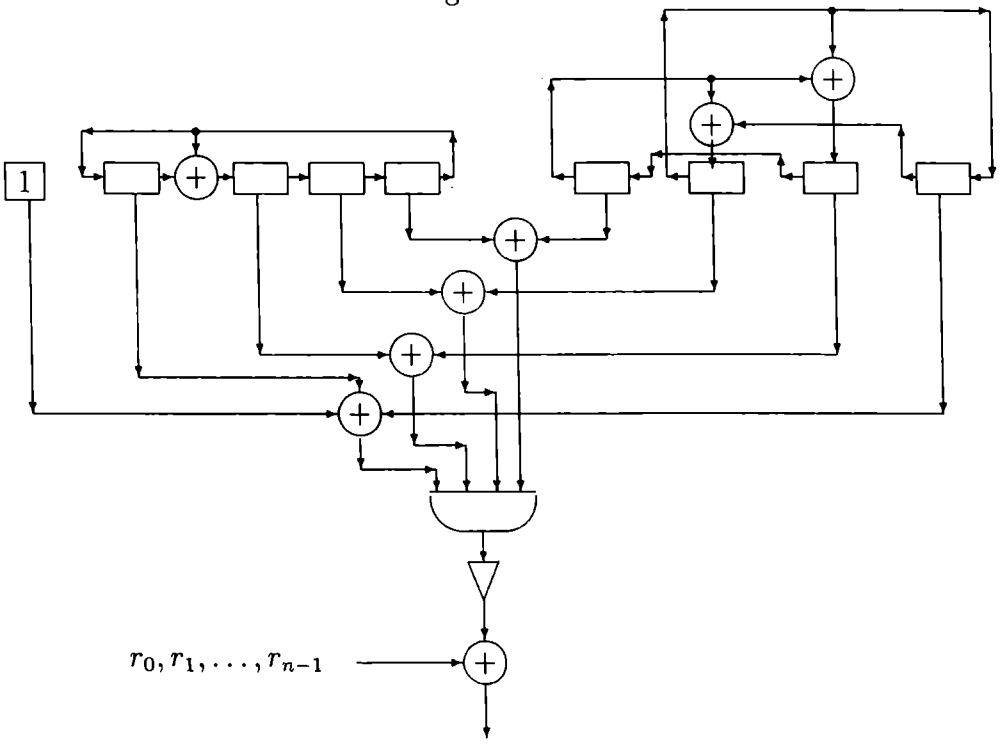
Atunci, circuitul de implementare a procedeului Chien de localizare a erorii este dat în Figura 8.1:

În acest circuit, cele patru elemente de înmagazinare din stânga formează circuitul de înmulţire cu α *(unde iniţial se introduce* σ_1*), iar cele patru elemente din dreapta formează circuitul de înmulţire cu* α^2*, aranjat în ordine inversă (unde* σ_2 *se introduce iniţial de la dreapta spre stânga).*

De asemenea, există un element de înmagazinare care conţine numai valoarea binară 1*, pe care o furnizează la fiecare tact.*

Iniţial, circuitul funcţionează odată la momentul 0 *fără a introduce nimic de pe canal (pentru "amorsarea" în elementele de înmagazinare a coeficienţilor pentru* $\sigma_1\alpha$ *respectiv* $\sigma_2\alpha^2$*).*

Figura 8.1:



La momentul următor se începe procesul efectiv de decodificare. Dacă r_{n-1} *a fost perturbat, atunci* α^{n-1} *este localizarea erorii, adică* α *este rădăcină a polinomului de localizare a erorii şi avem* $\mathbf{1}+\sigma_1\alpha+\sigma_2\alpha^2=\mathbf{0}$. *În acest caz, la intrarea în sumatorul "sau" este vectorul* $\mathbf{0}=(0,0,0,0)$, *iar la ieşire scalarul* 0. *După complementare se obţine scalarul* 1, *care se adună (pentru corecţie) la* r_{n-1}.

Dacă r_{n-1} *este corect, la intrarea în sumator se obţine o valoare nenulă, deci ieşirea va fi* 1 - *care prin complementare este* 0 *şi* r_{n-1} *rămâne nemodificat.*

8.4 Exerciţii

8.1 *Să se verifice valabilitatea descompunerii* $M=ABA^T$ *din demonstraţia Teoremei 8.5*

8.2 *Fie* $\alpha\in GF(2^m)$ $(m>2)$ *primitiv. Să se arate că polinoamele minimale* $m_1(X)$ *şi* $m_3(X)$ *au acelaşi grad.*

8.3 *Construiţi un cod BCH binar corector de* 2 *erori, de lungime* $n=11$.

8.4 *Construiţi un cod BCH ternar de lungime* $n = 26$ *şi* $d = 5$.

8.5 *Să se determine polinomul generator şi cel de control pentru codul BCH binar corector de* 3 *erori de lungime*

$(a)\ n = 15 \qquad (b)\ n = 31.$

Codificaţi mesajul de informaţie $11\ldots1$.

8.6 *Caracterizaţi codurile BCH care sunt conţin toate elementele extensiei Galois* $GF(q^m)$.

8.7 *Dacă* $g(X) = 1 + X^4 + X^6 + X^7 + X^8$ *este polinomul generator al unui cod BCH binar de lungime* 15, *să se cerceteze dacă următoarele secvenţe sunt cuvinte - cod:*

(a) 011001011000010; (b) 000111010000110;
(c) 011100000010001; (d) 111111111111111.

8.8 *Aflaţi rădăcinile polinoamelor cu coeficienţi în* $GF(2^4)$ *(*α *este rădăcina polinomului* $1 + X + X^4$*):*

$X^2 + \alpha^4 X + \alpha^{13}$, $X^2 + \alpha^7 X + \alpha^2$, $X^2\alpha^2 + X + \alpha^5$
$X^2 + \alpha^6$, $X^2 + \alpha^2 X$, $X^2 + X + \alpha^8$.

8.9 *Definim codul BCH de polinom generator* $g(X) = m_1(X)m_3(X)$ *peste* $GF(2^4)$ *folosind elementul primitiv* α, *rădăcină a polinomului* $1 + X + X^4$. *Se recepţionează cuvântul* $\mathbf{r}$ *şi se calculează sindromul* $\mathbf{r}H$. *Ştiind acest sindrom, localizaţi - dacă este posibil - erorile în fiecare din următoarele cazuri:*

(a) 01000101; (b) 11101000; (c) 11001101; (d) 01000000;
(e) 00000100; (f) 10100100; (g) 00111101; (h) 00000000.

8.10 *Pentru codul BCH definit în exerciţiul anterior, decodificaţi mesajele:*

(a) 110000000000000; (b) 000010000100001; (c) 010001010100000;
(d) 110011100111000; (e) 110011100100000; (f) 111000000000001;
(g) 101110000000000; (h) 101010010110001; (i) 010000100000000;
(j) 010101001011000; (k) 110111011101100; (l) 101110000001000;
(m) 111001011000000; (n) 000111010000110.

8.11 *În codul BCH binar de lungime* $n = 15$, *corector de* 3 *erori (vezi Exemplele 8.3 şi 8.4). să se decodifice mesajele:*

(a) 001000100000000; (b) 101011001000000.

8.12 *În codul BCH peste Z_3 de lungime $n = 8$, corector de 2 erori, să se decodifice mesajele*

(a) 00100000; (b) 12121212; (c) 11211122;
(d) 02020202; (e) 10201020; (f) 22110011.

8.13 *Fie $\alpha \in GF(2^5)$ primitiv, rădăcină a polinomului $1 + X^2 + X^5$. Se definește un cod BCH de lungime $n = 31$ și $d = 5$.*

1. *Să se construiască polinomul generator al codului:* $g(X) = m_1(X)m_3(X)$.

2. *Să se decodifice mesajul* 10010110111100001101010101111111.

Capitolul 9

Coduri Reed - Solomon

9.1 Definirea codurilor RS

O clasă foarte interesantă de coduri ciclice a fost definită în 1960 de Reed şi Solomon. Numite în articolul iniţial *coduri polinomiale*, Peterson arată un an mai târziu că ele sunt de fapt un caz particular de coduri BCH. Construcţia dată de autori a fost următoarea:

Fie $m \geq 2$ şi $\alpha \in GF(q^m)$ primitiv (deci $ord(\alpha) = n = q^m - 1$).

Dacă $\mathbf{a_0 a_1} \ldots \mathbf{a_{k-1}} \in GF(q^m)^k$ este secvenţa de informaţie, se defineşte polinomul $h(X) = \mathbf{a_0} + \mathbf{a_1}X + \ldots + \mathbf{a_{k-1}}X^{k-1} \in GF(q^m)[X]$. Codul polinomial va fi format de toate cuvintele de forma

$$h(\mathbf{1})h(\alpha)\ldots h(\alpha^{n-1}).$$

Teorema 9.1 *Un cod polinomial este un cod BCH peste $GF(q^m)$, definit de rădăcinile $\alpha, \alpha^2, \ldots, \alpha^{n-k}$.*

Demonstraţie: În algebra polinoamelor modulo $X^n - 1$ ($n = q^m - 1$), să considerăm clasa $\{f(X)\}$ asociată cuvântului - cod $h(\mathbf{1})h(\alpha)\ldots h(\alpha^{n-1})$: $f(X) = h(\mathbf{1}) + h(\alpha)X + \ldots + h(\alpha^{n-1})X^{n-1} = \mathbf{a_0}(1 + X + X^2 + \ldots + X^{n-1}) + \mathbf{a_1}(1 + \alpha X + \alpha^2 X^2 + \ldots + \alpha^{n-1}X^{n-1}) + \mathbf{a_2}(1 + \alpha^2 X + \alpha^4 X^2 + \ldots + (\alpha^2 X)^{n-1}) + \ldots + \mathbf{a_{k-1}}(1 + \alpha^{k-1}X + \ldots + (\alpha^{k-1}X)^{n-1})$.

Se ştie că toate elementele nenule din $GF(q^m)$ sunt rădăcinile polinomului $X^{q^m-1} - 1 = X^n - 1 = (X - 1)(1 + X + X^2 + \ldots + X^{n-1})$, deci toate elementele din $GF(q^m) \setminus \{\mathbf{0}, \mathbf{1}\}$ sunt rădăcinile polinomului $1 + X + X^2 + \ldots + X^{n-1}$. Reamintim, $\mathbf{0} = 00\ldots0$, $\mathbf{1} = 100\ldots0$.

Prin calcul se obţine $f(\mathbf{1}) = \mathbf{a_0}$, $f(\alpha^{-1}) = -\mathbf{a_1}$, $f(\alpha^{-2}) = -\mathbf{a_2}, \ldots$, $f(\alpha^{-(k-1)}) = -\mathbf{a_{k-1}}$ şi $f(\alpha^{-j}) = 0$ dacă $n - 1 \geq j \geq k$.

Dar cum $\alpha^n = 1$, avem $\alpha^{-j} = \alpha^{n-j}$ şi deci $f(\alpha^i) = 0$, $1 \leq i \leq n-k$. □

În continuare vom numi aceste coduri *Reed - Solomon*, pentru care vom folosi următoarea definiţie:

Definiţia 9.1 *Fie $\alpha \in GF(q^m)$ primitiv. Un cod cod Reed - Solomon (RS) este un cod BCH de polinom generator $g(X) \in GF(q^m)[X]$, definit prin*

$$g(X) = (X - \alpha^{m_0})(X - \alpha^{m_0+1}) \dots (X - \alpha^{m_0+d-2}).$$

Deci, diferenţa faţă de codurile BCH generale este aceea că un cod RS este definit direct pe extensie, nu pe corpul de bază. Astfel, la un cod BCH cuvintele - cod sunt din Z_q^n, pe când la un cod RS, ele sunt din $GF(q^m)^n$. Cum şi $GF(q^m)$ este corp, toate proprietăţile codurilor BCH se păstrează şi în cazul codurilor RS.

Deoarece au fost studiate (şi utilizate) numai codurile Reed - Solomon de caracteristică 2, vom considera în continuare cazul $q = 2$.

Exemplul 9.1 *Fie $GF(2^3)$ generat de rădăcina α a polinomului $1+X+X^3$, şi să considerăm codul RS definit de polinomul $g(X) = (\alpha+X)(\alpha^2+X) = \alpha^3+\alpha^4X+X^2$. Acesta este un cod ciclic de lungime $n = 7$, $k = 5$ şi $d = 3$. Folosind definiţia din Capitolul 5, secţiunea 5.3, se poate construi matricea generatoare a codului:*

$$G = \begin{pmatrix} \alpha^3 & \alpha^4 & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \alpha^3 & \alpha^4 & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \alpha^3 & \alpha^4 & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \alpha^3 & \alpha^4 & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \alpha^3 & \alpha^4 & \mathbf{1} \end{pmatrix}$$

Codul are 8^5 cuvinte, obţinute prin înmulţirea cu $g(X)$ a tuturor polinoamelor din $GF(2^3)[X]$ de grad cel mult 4 sau - echivalent - din înmulţirea cu matricea G a tuturor cuvintelor din $GF(2^3)^5$.

De exemplu, codificarea mesajului de informaţie $\mathbf{a} = \mathbf{1}\alpha\mathbf{0}\mathbf{0}\alpha^3$, sau $a(X) = \mathbf{1} + \alpha X + \alpha^3 X^4$ este $a(X)g(X) = \mathbf{1} + \alpha + (\alpha + \alpha^2)X^2 + \alpha X^3 + (\mathbf{1} + \alpha^2)X^4 + X^5 + (1 + \alpha)X^6 = \alpha^3 + \alpha^4 X^2 + \alpha X^3 + \alpha^6 X^4 + X^5 + \alpha^3 X^6$ (respectiv $\alpha^3\mathbf{0}\alpha^4\alpha\alpha^6\mathbf{1}\alpha^3$).

O matrice de control foarte simplă se obţine folosind (conform cu 7.3, Capitolul 7) rădăcinile α şi α^2 ale cuvintelor - cod (reamintim, $\alpha^7 = 1$):

$$H = \begin{pmatrix} \mathbf{1} & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ \mathbf{1} & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 \end{pmatrix}.$$

Teorema 9.2 *Distanţa minimă a unui cod RS este* $d = n - k + 1$.

Demonstraţie: Să notăm - temporar - cu d_H distanţa minimă a unui cod. Conform Teoremei 2.9 (Capitolul 2), $d_H \leq n - k + 1$. Cum într-un cod RS, $grad(g(X)) = n - k = d - 1$, vom avea $d = n - k + 1$. Din cele două afirmaţii rezultă $d_H \leq d$. Pe de-altă parte, din Teorema 8.2, cum codul RS este un caz particular de cod BCH, rezultă $d_H \geq d$. □

Din această teoremă rezultă că un cod RS are *distanţa maximă separabilă* (se mai spune că este un cod DMS).

9.2 Coduri RS scurte

Deoarece $\alpha \in GF(2^m)$ este primitiv, toate cuvintele unui cod RS au lungimea $n = 2^m - 1$ şi $k = n - d + 1 = 2^m - d$ simboluri de informaţie. Uneori însă sunt necesare coduri RS ale căror caracteristici să nu fie de această formă. Ele se obţin astfel:

Fie C un (n, k) - cod RS de distanţă minimă d şi s $(1 \leq s < 2^m - d)$ un număr întreg. Se construieşte codul RS *scurt* $C(s)$ format din toate cuvintele lui C care au $\mathbf{0}$ pe ultimele s poziţii, după care aceste s poziţii se ignoră.

Exemplul 9.2 *Fie codul* C *dat de* $g(X) = \alpha + X$, *unde* α *este rădăcina ecuaţiei* $1 + X + X^2 = 0$ *care generează* $GF(2^2)$. C *are deci* $n = 3$, $k = 2$, $d = 2$ *şi* 16 *cuvinte - cod. Codul* $C(1)$ *se obţine luând toate cuvintele lui* C *care au* $\mathbf{0}$ *pe ultima poziţie (acestea sunt* $\mathbf{000}, \alpha\mathbf{10}, \alpha^2\alpha\mathbf{0}, \mathbf{1}\alpha\mathbf{0}$*), din care se elimină ultimul caracter.*

Deci $C(1) = \{\mathbf{00}, \alpha\mathbf{1}, \alpha^2\alpha, \mathbf{1}\alpha\}$.

Atenţie: Ultima poziţie nu înseamnă ultimul bit. Fiecare element din cuvânt are aici două poziţii binare. Astfel, cele patru cuvinte - cod alese din C *au reprezentarea binară*

$$000000,\ 011000,\ 110100,\ 100100$$

iar codul scurt, reprezentat în binar este $\hat{C}(1) = \{0000, 0110, 1101, 1001\}$.

Din definiţie,

$$C(s) = \{\mathbf{b} \in C | grad(b(X)) < n - s\}.$$

Dacă $g(X)$ este polinomul generator al codului Reed - Solomon C, atunci $C(s)$ conţine toate polinoamele de forma $b(X)$ unde $b(X) = a(X)g(X)$, $grad(a(X)) < k - s = 2^m - d - s$ (deoarece $grad(g(X)) = d - 1 = n - k$).

De aici rezultă că matricea generatoare a unui cod RS scurt $C(s)$ este

$$G = \begin{pmatrix} g(X) \\ Xg(X) \\ \vdots \\ X^{k-s-1}g(X) \end{pmatrix}$$

obţinută din matricea generatoare a codului C prin eliminarea ultimelor s linii. Numărul de simboluri de informaţie este egal cu numărul de linii din G, deci $k_s = k - s = 2^m - d - s$.

Să notăm acum cu d respectiv d_s distanţele minime ale celor două coduri. Se observă că $\forall \mathbf{c_1}, \mathbf{c_2} \in C(s)$, $\mathbf{c_1'} = \mathbf{c_1 00 \ldots 0}, \mathbf{c_2'} = \mathbf{c_2 00 \ldots 0} \in C$ şi deci $d(\mathbf{c_1}, \mathbf{c_2}) = d(\mathbf{c_1'}, \mathbf{c_2'})$, adică $d_s \geq d$.

Pe de-altă parte (Teorema 2.9, Capitolul 2), $d_s \leq n_s - k_s + 1 = 2^m - 1 - s - (2^m - d - s) + 1 = d$, deci $d_s = d$.

Din cele arătate mai sus putem enunţa teorema:

Teorema 9.3 *Fie C un (n, k) - cod RS cu distanţa minimă d şi $C(s)$ codul său scurt, cu parametrii n_s, k_s, d_s. Atunci*

$$n_s = 2^m - 1 - s, \quad k_s = 2^m - d - s, \quad d_s = d$$

şi $C(s)$ este un cod DMS.

Observaţie: Alte coduri scurte se pot obţine eliminând orice combinaţie fixă de s elemente din cuvintele unui cod RS. Se poate vedea imediat că proprietăţile arătate mai sus nu se schimbă.

Exemplul 9.3 *Plecând de la codul RS din Exemplul 9.1, codul scurt $C(2)$ va avea ca matrice generatoare matricea G fără ultimele două linii şi coloane,*

$$G = \begin{pmatrix} \alpha^3 & \alpha^4 & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \alpha^3 & \alpha^4 & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \alpha^3 & \alpha^4 & \mathbf{1} \end{pmatrix}$$

iar ca parametri $n_2 = 5,\ k_2 = 3,\ d_2 = 3$.

9.3 Decodificarea codurilor RS

Fiind cazuri particulare de coduri BCH, codurile RS se pot decodifica folosind direct algoritmul Peterson (Capitolul 8, Secţiunea 8.2). Să reluăm detaliat pe un exemplu aplicarea acestui algoritm.

Exemplul 9.4 *Fie* $g(X) = (1+X)(\alpha+X)(\alpha^2+X)(\alpha^3+X)$ *unde* $\alpha \in GF(2^3)$ *este rădăcina (primitivă) a polinomului* $1+X+X^3$. *Codul are distanţa minimă* $d = 5$, *deci poate corecta cel mult* 2 *erori independente. Mai ştim că:*

$$\alpha^3 = 1+\alpha, \quad \alpha^4 = \alpha+\alpha^2, \quad \alpha^5 = 1+\alpha+\alpha^2, \quad \alpha^6 = 1+\alpha^2.$$

Să presupunem că s-a recepţionat cuvântul $\mathbf{r} = \alpha^6\alpha\alpha^5\alpha^2\mathbf{10}\alpha^2$. *Pentru decodificare se aplică următorii paşi:*

(1)*: Se calculează cele patru sindromuri:*

$S_0 = v(\alpha^0) = \alpha^6+\alpha+\alpha^5+\alpha^2+\mathbf{1}+\mathbf{0}+\alpha^2 = \mathbf{1}+\alpha^2+\alpha+\mathbf{1}+\alpha+\alpha^2+\alpha^2+\mathbf{1}+\alpha^2 = \mathbf{1}$;

$S_1 = v(\alpha) = \alpha^6+\alpha^2+\alpha^7+\alpha^5+\alpha^4+\mathbf{0}+\alpha^8 = \alpha^3$;

$S_2 = v(\alpha^2) = \alpha^6+\alpha^3+\alpha^9+\alpha^8+\alpha^8+\mathbf{0}+\alpha^{14} = \alpha^3$;

$S_3 = v(\alpha^3) = \alpha^6+\alpha^4+\alpha^{11}+\alpha^{11}\alpha^{12}+\mathbf{0}+\alpha^{20} = \mathbf{1}$.

(2) *Deoarece rangul matricii* $\begin{pmatrix} S_0 & S_1 \\ S_1 & S_2 \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \alpha^3 \\ \alpha^3 & \alpha^3 \end{pmatrix}$ *este* 2, *cuvântul* $\mathbf{r}$ *are două erori; cum distanţa minimă este* 5, *ele pot fi corectate.*

(3) *Se rezolvă sistemul liniar*

$$\begin{cases} S_0\sigma_2 + S_1\sigma_1 = S_2 \\ S_1\sigma_2 + S_2\sigma_1 = S_3 \end{cases} \text{ adică } \begin{cases} \sigma_2 + \alpha^3\sigma_1 = \alpha^3 \\ \alpha^3\sigma_2 + \alpha^3\sigma_1 = \mathbf{1} \end{cases}$$

Adunând cele două ecuaţii, se obţine $(\mathbf{1}+\alpha^3)\sigma_2 = \mathbf{1}+\alpha^3$, *deci* $\sigma_2 = \mathbf{1}$.

Înlocuind în prima ecuaţie, avem $\alpha^3\sigma_1 = \alpha^3+\mathbf{1} = \alpha$, *de unde, prin înmulţire cu* α^4 *se ajunge la* $\sigma_1 = \alpha^5$.

(4) *Polinomul de localizare a erorii este* $\sigma(X) = X^2+\sigma_1X+\sigma_2 = X^2+\alpha^5X+\mathbf{1}$. *Prin încercări se obţine* $\sigma(\alpha) = \sigma(\alpha^6) = \mathbf{0}$, *deci cele două erori sunt pe poziţiile* $X_1 = \alpha$, $X_2 = \alpha^6$.

(5) *Se rezolvă sistemul liniar* $\begin{pmatrix} X_1^0 & X_2^0 \\ X_1 & X_2 \end{pmatrix} \cdot \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} = \begin{pmatrix} S_0 \\ S_1 \end{pmatrix}$; *detaliind, acesta se are forma:*

$$\begin{cases} Y_1 + Y_2 = \mathbf{1} \\ \alpha Y_1 + \alpha^6 Y_2 = \alpha^3 \end{cases}$$

Înmulţind a doua ecuaţie cu α *şi adunând-o la prima, se obţine* $\alpha^6 Y_1 = \alpha^5$, *deci* $Y_1 = \alpha^6$. *După înlocuire în prima ecuaţie, se determină şi* $Y_2 = \alpha^2$.

(6) *Rezumând, au apărut erorile* α^6 *pe poziţia* 1 *şi* α^2 *pe poziţia* 6. *Deci eroarea este* $\mathbf{e} = \mathbf{0}\alpha^6\mathbf{0000}\alpha^2$, *iar cuvântul - cod trimis a fost*

$$\mathbf{c} = \mathbf{r}+\mathbf{e} = \alpha^6\alpha^5\alpha^5\alpha^2\mathbf{100}.$$

9.4 Altă construcţie a codurilor RS

Plecând de la construcţia iniţială, se poate realiza şi o altă definiţie pentru codurile RS. Ea se bazează pe o modalitate diferită de reprezentare a cuvintelor din Z_q^n, anume considerând polinoamele ca funcţii (polinomiale).

Definiţia 9.2 *Fie $S \subseteq GF(2^m)$. Două polinoame $p(X), q(X)$ reprezintă aceeaşi funcţie polinomială $: S \to GF(2^m)$ dacă şi numai dacă $\forall \beta \in S$, $p(\beta) = q(\beta)$.*

Exemplul 9.5 *Fie $S = GF(2^3)$, construit folosind rădăcina primitivă α a polinomului $1 + X + X^3$. Definim funcţia $f : S \to Z_2$ prin*

X	$\mathbf{0}$	$\mathbf{1}$	α	α^2	α^3	α^4	α^5	α^6
$f(X)$	0	0	1	1	0	1	0	0

Atunci $f(x)$ se poate reprezenta prin

$$\mathbf{v_f} = f(\mathbf{0})f(\mathbf{1})f(\alpha)f(\alpha^2)f(\alpha^3)f(\alpha^4)f(\alpha^5)f(\alpha^6) = 00110100.$$

Exemplul 9.6 *Fie $S = GF(2^3)$ şi funcţia $f : S \to S$ definită $\mathbf{v_f} = \alpha^4\mathbf{01}\alpha^2\mathbf{1}\alpha\mathbf{00}$. Folosind metoda coeficienţilor nedeterminaţi, pe baza valorilor punctelor din S, obţinem reprezentarea lui f sub formă de polinom:*

$$f(X) = \alpha^4 + \alpha^2 X + \alpha^3 X^2 + X^3.$$

Mulţimea $A = \{f(X) \in GF(2^m)[X] | grad(f(X)) \leq k-1\}$ se poate structura ca un spaţiu liniar de dimensiune k, cu baza $\{\mathbf{1}, X, X^2, \ldots, X^{k-1}\}$. Deoarece toate aceste polinoame pot fi considerate funcţii definite pe o anumită mulţime $S \subseteq GF(2^m)$, vom numi A "*spaţiul funcţiilor pe S*".

De remarcat că în această fază S nu este definit explicit.

Teorema 9.4 *Fie mulţimea S, $card(S) = n$ şi $k \leq n$. Mulţimea tuturor funcţiilor $: S \to GF(2^m)$ reprezentate prin polinoame de grad $\leq k-1$ formează un spaţiu liniar de funcţii, cu baza $\{\mathbf{1}, X, \ldots, X^{k-1}\}$.*

Demonstraţie: Evident, orice polinom de grad cel mult $k-1$ este generat de $\{\mathbf{1}, X, \ldots, X^{k-1}\}$. Tot ce trebuie arătat este că fiecare funcţie polinomială $f : S \to GF(2^m)$ are o reprezentare unică sub formă de polinom.

Să presupunem (prin absurd) că $p(X)$ şi $q(X)$ sunt egale ca funcţii pe S. Deci $\forall \alpha \in S$, $p(\alpha) = q(\alpha)$. Dar atunci $p(X) - q(X)$ este un polinom de grad $< k$ care are n $(n \geq k)$ rădăcini, deci $p(X) - q(X) \equiv 0$, adică $p(X) \equiv q(X)$. □

Exemplul 9.7 *Fie $S = GF(2^3)$ construit folosind rădăcina α a ecuaţiei $1 + X + X^3 = 0$, şi să considerăm toate polinoamele din $GF(2^3)[X]$, de grad cel mult doi. O bază pentru acest spaţiu de funcţii este $\{\mathbf{1}, X, X^2\}$, cu reprezentarea corespunzătoare:*

$$\begin{array}{lcl} \mathbf{1} & - & \mathbf{11111111} \\ X & - & \mathbf{01}\alpha\alpha^2\alpha^3\alpha^4\alpha^5\alpha^6 \\ X^2 & - & \mathbf{01}\alpha^2\alpha^4\alpha^6\alpha\alpha^3\alpha^5 \end{array}$$

Atunci, orice polinom $f(X) = \mathbf{a_0} + \mathbf{a_1}X + \mathbf{a_2}X^2$ considerat ca o funcţie polinomială, poate fi reprezentat sub formă de vector prin

$$\mathbf{v_f} = (\mathbf{a_0}\ \mathbf{a_1}\ \mathbf{a_2}) \begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ \mathbf{0} & \mathbf{1} & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 \end{pmatrix}.$$

Sunt $8^3 = 512$ astfel de funcţii polinomiale.

Teorema 9.5 *Spaţiul funcţiilor pe $S \subseteq GF(2^m)$ $(card(S) = n)$ al tuturor polinoamelor din $GF(2^m)[X]$ de grad $\leq k-1$ formează un (n, k) - cod liniar DMS.*

Demonstraţie: Alegem o mulţime cu n elemente $S \subseteq GF(2^m)$, şi considerăm spaţiul liniar al tuturor funcţiilor polinomiale $p : S \to GF(2^m)$ de grad cel mult $k-1$. Lungimea fiecărui cuvânt - cod este n, iar dimensiunea spaţiului este k (cu Teorema 9.4). Mai rămâne de arătat că acest cod este DMS (are distanţă maximă separabilă), adică $d = n - k + 1$. Pentru aceasta, să observăm că orice polinom $p(X)$ are maxim $k-1$ rădăcini distincte; deci reprezentarea $\mathbf{v_p}$ are cel mult $k-1$ zero-uri, adică $w(\mathbf{v_p}) \geq n - k + 1$. În particular, $d = min(w(\mathbf{v_p}) \geq n - k + 1$. Pe de-altă parte, conform Teoremei 2.9, (Capitolul 2) $d \leq n - k + 1$, deci $d = n - k + 1$. □

Fie n un divizor al lui $2^m - 1$ şi $S = \{\alpha \in GF(2^m) | \alpha^n = 1\}$ mulţimea rădăcinilor de ordinul n ale unităţii din $GF(2^m)$. Evident, S conţine toate rădăcinile din $GF(2^m)$ ale ecuaţiei $X^n + 1 = 0$ şi - fiind un corp finit - conţine cel puţin un element primitiv (Teorema 7.1, Capitolul 7).

Teorema 9.6 *Fie $p(X), q(X) \in GF(2^m)[X]$ şi $S \subseteq GF(2^m)$ mulţimea rădăcinilor de ordinul n ale unităţii. Atunci $p(X)$ şi $q(X)$ reprezintă aceeaşi funcţie $f : S \to GF(2^m)$ dacă şi numai dacă*

$$p(X) \equiv q(X) \pmod{X^n + 1}.$$

Demonstraţie: " $\Longleftarrow$": Fie $q(X) = h(X)(X^n + 1) + p(X)$. Atunci pentru orice $\alpha \in S$ avem $q(\alpha) = h(\alpha)(1 + \alpha^n) + p(\alpha)$. Dar $1 + \alpha^n = 0$, deci $q(\alpha) = p(\alpha)$.

" $\Longrightarrow$": Fie $\alpha \in S$ primitiv. Deci orice element din S are forma α^i $(0 \leq i < n)$. Dacă $q(\alpha^i) = p(\alpha^i)$, atunci α^i $(0 \leq i < n)$ sunt rădăcini ale polinomului $p(X) - q(X)$. Putem scrie atunci

$$p(X) - q(X) = h(X) \prod_{i=0}^{n-1} (1 + \alpha^i) = h(X)(X^n + 1). \qquad \square$$

Teorema 9.7 *Fie S mulţimea rădăcinilor de ordinul n ale unităţii din $GF(2^m)$. Atunci spaţiul funcţiilor tuturor polinoamelor din $GF(2^m)[X]$ de grad $\leq k - 1$ pe S formează un (n, k) - cod ciclic peste $GF(2^m)$.*

Demonstraţie: Să notăm cu C acest spaţiu liniar, şi fie S (ca în ipoteză) generat de un element primitiv α. Atunci, dacă $p(X) \in GF(2^m)[X]$ este un polinom de grad cel mult $k - 1$, avem $\mathbf{v_p} = p(\mathbf{1})p(\alpha) \ldots p(\alpha^{n-1}) \in C$.

Fie $q(X) = p(\alpha X)$. Cum $grad(q(X)) \leq k - 1$, rezultă $\mathbf{v_q} \in C$. Dar

$$q(\mathbf{1})q(\alpha) \ldots q(\alpha^{n-1}) = p(\alpha)p(\alpha^2) \ldots p(\alpha^{n-1})p(\mathbf{1}) \in C,$$

care reprezintă - conform definiţiei - un cod ciclic. $\square$

Exemplul 9.8 *Să considerăm $GF(2^3)$ construit ca în Exemplul 9.7 şi S spaţiul rădăcinilor de ordinul 7 ale unităţii, generat de α. Fie de exemplu $p(X) = \alpha^4 + \alpha^2 X + \alpha^3 X^2 + X^3$, care are reprezentarea $\mathbf{01\alpha^2 1\alpha 00}$. Permutarea ei ciclică este $\mathbf{1\alpha^2 1\alpha 000}$, care corespunde funcţiei $p(\alpha X) = \alpha^4 + \alpha^3 X + \alpha^5 X^2 + \alpha^3 X^3$.*

Exemplul 9.9 *Fie $GF(2^4)$ construit cu rădăcina α a ecuaţiei $1 + X + X^4 = 0$ şi fie $\beta = \alpha^5$ o rădăcină de ordinul 3 a unităţii. Deci $S = \{\mathbf{1}, \alpha^5, \alpha^{10}\}$.*

Pentru $k = 1$ se obţine un cod cu repetiţie de lungime 3.

Pentru $k = 2$, codul este generat de toate polinoamele $p(X) = \mathbf{a_0} + \mathbf{a_1} X \in GF(2^4)[X]$, considerate ca funcţii : $S \to GF(2^4)$. De exemplu, pentru $p(X) = \alpha^6 + \alpha^{10} X$, obţinem cuvântul - cod

$$\mathbf{v_p} = p(\mathbf{1})p(\alpha^5)p(\alpha^{10}) = \alpha^7 \alpha^{13} \alpha^9.$$

Definiţia 9.3 *Fie polinomul $V(X) = \mathbf{V_0} + \mathbf{V_1} X + \ldots + \mathbf{V_{n-1}} X^{n-1} \in GF(2^m)[X]$ şi $\alpha \in GF(2^m)$ o rădăcină primitivă de ordinul n a unităţii. Spunem că $v(X) = \mathbf{v_0} + \mathbf{v_1} X + \ldots + \mathbf{v_{n-1}} X^{n-1}$ este "transformata" lui $V(X)$ dacă*

$$V(\alpha^j) = \sum_{i=0}^{n-1} \mathbf{V_i} \alpha^{ji} = v_j, \quad j = 0, 1, \ldots, n - 1.$$

Altfel spus, dacă se notează cu

$$M_{n,n} = \begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \alpha & \alpha^2 & \alpha^{n-1} \\ \mathbf{1} & \alpha^2 & \alpha^4 & \alpha^{2(n-1)} \\ & & \vdots & \\ \mathbf{1} & \alpha^{n-1} & \alpha^{2(n-1)} & \alpha^{(n-1)^2} \end{pmatrix},$$

atunci

$$(\mathbf{V_0}\ \mathbf{V_1}\ \ldots\ \mathbf{V_{n-1}})M_{n,n} = (\mathbf{v_0}\ \mathbf{v_1}\ \ldots\ \mathbf{v_{n-1}}).$$

Matricea $M_{n,n}$ este numită *transformata Fourier finită*. Ea este nesingulară, deci se poate scrie $(\mathbf{V_0}\ \mathbf{V_1}\ \ldots\ \mathbf{V_{n-1}}) = (\mathbf{v_0}\ \mathbf{v_1}\ \ldots\ \mathbf{v_{n-1}})M_{n,n}^{-1}$ sau

$$\mathbf{V_i} = \sum_{i=1}^{n-1} \mathbf{v_j}\alpha^{-ij} = v(\alpha^{-i}).$$

Vom demonstra această formulă în mod direct, fără a folosi inversa matricii $M_{n,n}$.

Teorema 9.8 *Fie $\alpha \in GF(2^m)$ o rădăcină primitivă de ordin n a unităţii. Dacă $\mathbf{v_j} = V(\alpha^j)$ pentru $V(X) = \mathbf{V_0} + \mathbf{V_1}X + \ldots + \mathbf{V_{n-1}}X^{n-1} \in GF(2^m)[X]$, atunci $\mathbf{V_i} = v(\alpha^{-i})$ unde $v(X) = \mathbf{v_0} + \mathbf{v_1}X + \ldots + \mathbf{v_{n-1}}X^{n-1}$.*

Demonstraţie: Deoarece α este primitiv, el generează grupul multiplicativ $S = \{\mathbf{1}, \alpha, \ldots, \alpha^{n-1}\}$. Conform Teoremei 7.3 (Capitolul 7), orice element din S este rădăcină a ecuaţiei $X^n - 1 = 0$.

Din $(X-1)(X^{n-1} + X^{n-2} + \ldots + X + 1) = 0$ rezultă că α^p este rădăcină pentru $X - 1 = 0$ când $p = 0$ respectiv pentru $(X-1)(X^{n-1} + X^{n-2} + \ldots + X + 1 = 0$ când $p \neq 0$. Altfel spus, vm avea

$$\sum_{j=0}^{n-1} \alpha^{(k-i)j} = \begin{cases} n\ (mod\ 2) & \text{pentru} \quad k = i \\ 0 & \text{pentru} \quad k \neq i \end{cases}$$

Atunci

$$v(\alpha^{-i}) = \sum_{j=0}^{n-1} \mathbf{v_j}\alpha^{-ij} = \sum_{j=0}^{n-1} \left(\sum_{k=0}^{n-1} \mathbf{V_k}\alpha^{kj} \right) \alpha^{-ij} = \sum_{k=0}^{n-1} \mathbf{V_k} \left(\sum_{j=0}^{n-1} \alpha^{(k-i)j} \right) =$$

$\mathbf{V_i}$. □

Deci, fiind dat un vector de valori $(\mathbf{v_0}, \mathbf{v_1}, \ldots, \mathbf{v_{n-1}})$, se pot regăsi coeficienţii polinomului $V(X) = \mathbf{V_0} + \mathbf{V_1}X + \ldots + \mathbf{V_{n-1}}X^{n-1}$, lucru esenţial în definirea unui algoritm de decodificare.

Teorema 9.9 *Fie $S \subseteq GF(2^m)$ mulţimea rădăcinilor de ordinul n ale unităţii din $GF(2^m)$, generat de rădăcina primitivă $\alpha \in GF(2^m)$. Spaţiul*

liniar al funcţiilor polinomiale de grad $< n-d+1$ *pe* S *formează un cod ciclic DMS cu polinomul generator*

$$g(X) = (\alpha + X)(\alpha^2 + X)\dots(\alpha^{d-1} + X).$$

Demonstraţie: Funcţia polinomială $V(X)$ a cărei transformată corespunde lui $v(X) = a(X)g(X)$ este $V(X) = \sum_{i=0}^{n-1} v(\alpha^{n-i})X^i$. Deoarece $g(\alpha^p) = 0$ $(1 \le p \le d-1)$, vom avea $v(\alpha^{n-i}) = 0$ pentru $i = n-d+1, n-d+2, \dots, n-1$. Rezultă că pentru orice i $(n-d+1 \le i \le n-1)$, coeficientul lui X^i din $V(X)$ este zero, deci $V(X)$ are gradul $< n-d+1$. □

Dacă se ia $n = 2^m - 1$, se ajunge la o altă modalitate de construcţie a codurilor RS, bazată pe altă matrice generatoare (deci o altă dispunere a caracterelor de informaţie).

Exemplul 9.10 *Fie* α *rădăcina polinomului* $1 + X + X^3$, *cu care se construieşte* $GF(2^3)$. *Considerăm codul RS de polinom generator* $g(X) = (\mathbf{1} + X)(\alpha + X)(\alpha^2 + X)(\alpha^3 + X) = \alpha^6 + \alpha^5 X + \alpha^5 X^2 + \alpha^2 X^3 + X^4$ *(care corespunde reprezentării* $\alpha^6\alpha^5\alpha^5\alpha^2\mathbf{100}$*). Transformata lui* $g(X)$ *este polinomul* $G(X) = \sum_{i=0}^{6} g(\alpha^{7-i})X^i$.

Pentru că $g(\alpha^0)g(\alpha)\dots g(\alpha^6) = \mathbf{00001}\alpha\alpha^4$, *avem* $G(X) = g(\mathbf{1})X^0 + g(\alpha^{7-1})X + g(\alpha^{7-2})X^2 + g(\alpha^{7-3})X^3 = \alpha^4 X + \alpha X^2 + X^3$.

Se verifică uşor că $G(X)$ *este o funcţie polinomială cu reprezentarea*

$$G(\alpha^0)G(\alpha)\dots G(\alpha^6) = \alpha^6\alpha^5\alpha^5\alpha^2\mathbf{100}.$$

Putem gândi acest cod RS ca spaţiul tuturor polinoamelor de grade $1, 2$ *şi* 3, *adică*

$$\{Xp(X) | p(X) \in GF(2^3)[X],\ grad(p(X)) < 3\}.$$

Baza pentru acest spaţiu liniar este $\{X, X^2, X^3\}$, *iar matricea generatoare se defineşte imediat:*

$$\begin{pmatrix} \mathbf{1} & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ \mathbf{1} & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 \\ \mathbf{1} & \alpha^3 & \alpha^6 & \alpha^2 & \alpha^5 & \alpha & \alpha^4 \end{pmatrix}$$

Deci $G(X) = \alpha^4 X + \alpha X^2 + X^3$ *va avea reprezentarea*

$$(\alpha^4\ \alpha\ \mathbf{1}) \begin{pmatrix} \mathbf{1} & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ \mathbf{1} & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 \\ \mathbf{1} & \alpha^3 & \alpha^6 & \alpha^2 & \alpha^5 & \alpha & \alpha^4 \end{pmatrix} = (\alpha^6\ \alpha^5\ \alpha^5\ \alpha^2\ \mathbf{1}\ \mathbf{0}\ \mathbf{0}).$$

9.4.1 Algoritmul Berlekamp - Massey

Fie $g(X)$ polinomul generator al unui cod RS, $c(X) = a(X)g(X)$ un polinom - cod transmis (care codifică mesajul de informaţie $a(X)$), şi $v(X) = c(X) + e(X)$ polinomul recepţionat. Notăm $V(X), C(X)$ şi $E(X)$ transformatele lui $v(X), c(X)$ şi respectiv $e(X)$. Cum transformata Fourier finită este o aplicaţie liniară, se verifică imediat $V(X) = C(X) + E(X)$.

Deoarece $g(X)$ are $d-1$ rădăcini consecutive α^i ($m_0 \leq i \leq m_0 + d - 2$), sindromul polinomial $s(X)$ va avea $s(\alpha^{n-i}) = e(\alpha^{n-i}) = E_i$ pentru $n - m_0 - d + 1 \leq i \leq n - m_0$. Deci sindromul va putea determina $d-1$ coeficienţi ai transformatei $E(X)$. Pentru ceilalţi coeficienţi va trebui să reluăm polinomul de localizare a erorilor $\sigma(X)$ (din algoritmul Peterson de decodificare a codurilor BCH).

$\sigma(X)$ are proprietatea că $\sigma(\alpha^k) = 0 \iff e_k \neq 0$. Deoarece $E(\alpha^k) = e_k$, vom avea $\sigma(\alpha^k)E(\alpha^k) = 0\ \forall k$, deci - lucrând în algebra polinoamelor modulo $X^n + 1$:

$$\{\sigma(X)\}\{E(X)\} = 0$$

şi

$$\{\sigma(X)\}\{E(X)\} = \left\{ \left(\sum_{i=0}^{t} \sigma_i X^i \right) \left(\sum_{k=0}^{n-1} \mathbf{E_k} X^k \right) \right\},$$

unde $t = [(d-1)/2]$. Calculând în ambele relaţii coeficientul lui X^{t+k}, se obţine:

$$\sigma_t \mathbf{E_k} + \sigma_{t-1} \mathbf{E_{k+1}} + \ldots + \sigma_0 \mathbf{E_{k+t}} = 0.$$

Pentru că ştim deja $d-1$ valori consecutive ale lui $\mathbf{E_k}$, putem determina recursiv coeficienţii σ_i, pe care îi folosim la generarea tuturor valorilor $\mathbf{E_k}$.

De remarcat că în [7] se defineşte algoritmul Berlekamp - Massey sub o altă formă.

Exemplul 9.11 *Să reluăm codul RS definit în Exemplul 9.4, în care considerăm că s-a recepţionat cuvântul* $\mathbf{v} = \alpha^6\alpha\alpha^5\alpha^2\mathbf{10}\alpha^2$. *Deoarece* $d = 5$, *avem* $t \leq 2$ *şi* $\mathbf{E_0} = v(\alpha^0) = \mathbf{1}$, $\mathbf{E_6} = v(\alpha) = \alpha^3$, $\mathbf{E_5} = v(\alpha^2) = \alpha^3$, $\mathbf{E_4} = v(\alpha^3) = \mathbf{1}$. *Polinomul de localizare a erorii este* $\sigma(X) = X^2 + \sigma_1 X + \sigma_0$. *Determinăm* σ_0, σ_1 *ca în Exemplul 9.4 şi găsim* $\sigma_0 = \mathbf{1}$, $\sigma_1 = \alpha^5$. *Se poate determina astfel relaţia de recurenţă:*

$$\mathbf{E_k} = \alpha^5 \mathbf{E_{k+1}} + \mathbf{E_{k+2}}.$$

Deoarece $(\mathbf{E_0}, \mathbf{E_6}, \mathbf{E_5}, \mathbf{E_4}) = (\mathbf{1}, \alpha^3, \alpha^3, \mathbf{1})$, *vom avea:*

$$\mathbf{E_3} = \alpha^5 \mathbf{E_4} + \mathbf{E_5} = \alpha^5 + \alpha^3 = \alpha^2,$$
$$\mathbf{E_2} = \alpha^5 \mathbf{E_3} + \mathbf{E_4} = \alpha^5 \cdot \alpha^2 + \mathbf{1} = \mathbf{0},$$

$$\mathbf{E_1} = \alpha^5\mathbf{E_2} + \mathbf{E_3} = \mathbf{0} + \alpha^2 = \alpha^2.$$

Am determinat transformata lui $e(X)$, *adică* $E(X) = \mathbf{1} + \alpha^2 X + \alpha^2 X^3 + X^4 + \alpha^3 X^5 + \alpha^3 X^6$.

De aici rezultă $\mathbf{e} = (E(\alpha^0), E(\alpha), E(\alpha^2), \ldots, E(\alpha^6)) = \mathbf{0}\alpha^6\mathbf{0000}\alpha^2$, *deci cuvântul decodificat este*

$$\mathbf{c} = \mathbf{v} + \mathbf{e} = \alpha^6\alpha^5\alpha^5\alpha^2\mathbf{100}.$$

Pe de-altă parte, dacă s-ar fi folosit matricea generatoare din Exemplul 9.10, nu ar mai fi fost necesar să se determine vectorul eroare $\mathbf{e}$, *ci doar să se calculeze toate valorile lui* $v(\alpha^k)$, $0 \leq k \leq 6$, *din care se obţine direct mesajul transmis; mai precis, se află transformata lui* $v(X)$, *care se adună la transformata lui* $e(X)$:

$\mathbf{v_0v_1}\ldots\mathbf{v_6} = v(\alpha^0)v(\alpha^6)v(\alpha^5)\ldots v(\alpha) = \mathbf{1}\alpha\alpha\alpha^6\mathbf{1}\alpha^3\alpha^3$,

$\mathbf{E_0E_1}\ldots\mathbf{E_6} = \mathbf{1}\alpha^2\mathbf{0}\alpha^2\mathbf{1}\alpha^3\alpha^3$, *şi deci*

$\mathbf{C_0C_1}\ldots\mathbf{C_6} = \mathbf{V_0V_1}\ldots\mathbf{V_6} + \mathbf{E_0E_1}\ldots\mathbf{E_6} = \mathbf{0}\alpha^4\alpha\mathbf{1000}$.

S-a obţinut $C(X) = \alpha^4 X + \alpha X^2 + X^3$. *Deci caracterele de informaţie sunt* $(\alpha^4, \alpha, \mathbf{1})$. *De aici se obţine forma cuvântului - cod transmis*

$$\mathbf{c} = \alpha^6\alpha^5\alpha^5\alpha^2\mathbf{100}.$$

9.5 Ştergeri

O *ştergere* este o eroare detectată, pentru corectarea ei trebuind determinată doar valoarea erorii. *Numărul locaţiei de ştergere* este poziţia elementului perturbat. Acest număr este în general cunoscut (din citirea fizică a mesajului primit sau din structura codului).

Astfel, fie C un cod RS peste $GF(2^m)$ şi să considerăm codul $\hat{C}$ format din reprezentarea binară a cuvintelor codului C (fiecare element din $GF(2^m)$ se scrie ca un cuvânt binar de lungime m). Se defineşte şi codul $\hat{C}'$ obţinut prin adăugarea câte unui bit de paritate la fiecare simbol din componenţa cuvintelor codului $\hat{C}$.

Exemplul 9.12 *Să considerăm codul RS din Exemplul 9.2. Pentru a forma* $\hat{C}'$, *se înlocuieşte fiecare element* $\mathbf{0}, \mathbf{1}, \alpha, \alpha^2$ *respectiv cu* $000, 101, 011, 110$ *(al treilea bit este bitul de paritate).*

Astfel, pentru $\alpha\mathbf{10} \in C$ *avem* $011101000 \in \hat{C}'$.

Deoarece fiecare element dintr-un cuvânt - cod $\mathbf{c} \in C$ este reprezentat printr-un cuvânt binar de lungime $m+1$, cuvântul - cod corespunzător $\hat{\mathbf{c}}' \in \hat{C}'$ va avea lungimea $(2^m - 1)(m+1)$. Se observă că fiecare element nenul din $\mathbf{c}$ este înlocuit cu un cuvânt de pondere pară. Deci, $\hat{\mathbf{c}}'$ va fi

format din $2^m - 1$ cuvinte binare de lungimi $m+1$, fiecare de pondere pară. În acest fel, dacă în cuvântul recepţionat $\hat{\mathbf{v}}'$ există un grup de pondere impară, ştim că a apărut o eroare printre cei $m+1$ biţi corespunzători. Considerat din nou ca un cuvânt cu elemente din $GF(2^m)$, ştim că acest cuvânt are un element perturbat; nu cunoaştem valoarea erorii, dar ştim poziţia sa. Avem deci o "ştergere".

Exemplul 9.13 *Reluând construcţia din exemplul anterior, să presupunem că s-a primit* 011100000. *Deci, au apărut erori în al doilea grup de* 3 *biţi (acesta are pondere impară), aşa că* α^1 *este un numărul locaţiei de ştergere. Ştiind acest număr, putem înlocui al doilea element cu* **0** *(uşurează calculul sindromului), şi vom încerca decodificarea lui* $\mathbf{v} = \alpha\mathbf{00}$ *în cel mai apropiat cuvânt - cod din* C, *ştiind că poziţia erorii este* $X_1 = \alpha$.

Teorema 9.10 *Fie* C *un cod RS peste* $GF(2^m)$ *de distanţă minimă* d, *şi* $\mathbf{v}$ *un cuvânt recepţionat care are* s *ştergeri şi* t *erori care nu sunt ştergeri. Atunci* $\mathbf{v}$ *poate fi decodificat corect dacă* $2t + s \leq d - 1$.

Demonstraţie: De remarcat că în Teorema 2.13 (Capitolul 2), o condiţie mai slabă ($t+s \leq d-1$) asigură corectarea celor t erori, şi detectează (numai) celelalte s ştersături. Această teoremă realizează corectarea completă a cuvântului.

Fie B mulţimea poziţiilor ştersăturilor şi A mulţimea poziţiilor erorilor; deci $A \setminus B$ este mulţimea poziţiilor erorilor care nu sunt ştersături. Definim $\sigma_B(X) = \prod_{i \in B}(\alpha^i + X)$ ca fiind polinomul de localizare a ştersăturilor. Atunci polinomul de localizare a erorilor se poate scrie

$$\sigma_A(X) = \sigma_B(X)\sigma_{A\setminus B}(X).$$

Aflarea poziţiilor erorilor revine la determinarea rădăcinilor polinomului $\sigma_{A\setminus B}(X)$. Pentru aceasta se poate folosi algoritmul Peterson de decodificare a codurilor BCH (Capitolul 8), cu câteva schimbări, corespunzătoare sindromurilor "modificate".

Astfel, fie $\sigma_B(X) = B_0 + B_1X + \ldots + B_{s-1}X^{s-1} + X^s$ şi $\sigma_{A\setminus B}(X) = A_0 + A_1X + \ldots + A_{t-1}X^{t-1} + X^t$.

Similar algoritmului Peterson, vom înmulţi ambii membri ai egalităţii $\sigma_B(X)\sigma_{A\setminus B}(X) = \sigma_A(X)$ cu $Y_iX_i^j$ ($m_0 \leq j \leq m_0 + d - 2$) (reamintim, X_i corespund locaţiilor erorilor, iar Y_i sunt valorile acestor erori).

Apoi se înlocuieşte $X = X_i$ şi se face suma după $i = 1, \ldots, t+s$. Se obţine

$$S'_j A_0 + S'_{j+1} A_1 + \ldots + S'_{j+t-1} A_{t-1} + S'_{j+t} = 0 \ (m_0 \leq j \leq m_0 + d - 2), (1)$$

unde S'_k sunt sindromurile modificate, de forma

$$S'_k = B_0 S_k + B_1 S_{k+1} + \ldots + B_{s-1} S_{k+s-1} + S_{k+s}. \qquad (2)$$

Pentru că se ştiu valorile lui S_j pentru $j = m_0, \ldots, m_0 + d - 2$ iar $B_0, B_1, \ldots, B_{s-1}$ sunt cunoscute, se pot determina S'_j pentru $m_0 \leq j \leq d - 1 - s$. Deoarece $2t + s \leq d - 1$, deci $2t \leq d - 1 - s$, se poate rezolva sistemul (1) scris matricial

$$\begin{pmatrix} S'_{m_0} & S'_{m_0+1} & & S'_{m_0+t-1} \\ S'_{m_0+1} & S'_{m_0+2} & \cdots & S'_{m_0+t} \\ & \vdots & & \\ S'_{m_0+t-1} & S'_{m_0+t} & \cdots & S_{m_0+2t-1} \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_{t-1} \end{pmatrix} = \begin{pmatrix} S'_{m_0+t} \\ S'_{m_0+t+1} \\ \vdots \\ S_{m_0+2t} \end{pmatrix} \qquad (3)$$

cu t necunoscute $A_0, A_1, \ldots, A_{t-1}$.

După determinarea polinomului de localizare a erorilor, algoritmul Peterson se poate aplica în continuare nemodificat. □

Fie C un cod RS peste $GF(2^m)$, de polinom generator $g(X) = (\alpha^{m_0} + X)\ldots(\alpha^{m_0+d-2} + X)$. Se transmite cuvântul - cod $\mathbf{c}$ şi se recepţionează $\mathbf{v}$ cu s ştergeri localizate în elementele mulţimii $B = \{X_1, \ldots, X_s\}$.

Fie $\sigma_B(X) = B_0 + B_1 X + \ldots + B_{s-1} X^{s-1} + X^s$ polinomul de localizare al ştergerilor.

Polinomul de localizare a erorilor $\sigma_A(X) = \sigma_{A\setminus B}(X)\sigma_B(X)$ se poate construi aflând $\sigma_{A\setminus B}(X) = A_0 + A_1 X + \ldots + A_{t-1} X^{t-1} + X^t$ astfel:

1. Se calculează $S_j = v(\alpha^j)$ pentru $j = m_0, m_0+1, \ldots, m_0+d-2$;
2. Se determină S'_j $(m_0 \leq j \leq m_0 + d - 2 - s)$ cu relaţiile (2);
3. Se rezolvă sistemul liniar (3) de necunoscute $A_0, \ldots, A_{t-1}$.

Exemplul 9.14 *Fie C codul RS peste $GF(2^4)$ construit de rădăcina α a ecuaţiei $1 + X + X^4 = 0$, definit de polinomul generator $g(X) = (1 + X)(\alpha + X)\ldots(\alpha^4 + X)$ şi $\hat{C}'$ codul binar asociat în care sunt codificate mesajele. S-a primit mesajul*

$$\hat{\mathbf{v}}' = 11101\ 11001\ 00101\ 00110\ 10010\ 0\ldots0,$$

deci $\mathbf{v} = \alpha^{10}\mathbf{0}\alpha^2\mathbf{0}\alpha^6\alpha^{14}\mathbf{0}\ldots\mathbf{0}$.

Singura ştergere localizată este α^1, *deci* $\sigma_B(X) = \alpha + X$.

Trebuie să determinăm polinomul de localizare a erorilor.

Din $v(X) = \alpha^{10} + \alpha^2 X^2 + \alpha^6 X^4 + \alpha^{14} X^5$ *calculăm sindromurile*

$$S_0 = \alpha^5,\ S_1 = \mathbf{0},\ S_2 = \alpha^3,\ S_3 = \alpha^4,\ S_4 = \alpha^3.$$

Deoarece $B_0 = \alpha$ *şi* $s = 1$, *relaţia de recurenţă pentru sindromurile modificate este* $S'_j = B_0 S_j + S_{j+1} = \alpha S_j + S_{j+1}$ $(0 \le j \le 3)$.

Se determină astfel $S'_0 = \alpha^6,\ S'_1 = \alpha^3,\ S'_2 = \mathbf{0},\ S'_3 = \alpha^{11}$.

Deoarece $d = 5$ *avem* $t = 2$, *deci sistemul* (3) *este*

$$\begin{pmatrix} S'_0 & S'_1 \\ S'_1 & S'_2 \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \end{pmatrix} = \begin{pmatrix} S'_2 \\ S'_3 \end{pmatrix},$$

sau

$$\begin{pmatrix} \alpha^6 & \alpha^3 \\ \alpha^3 & \mathbf{0} \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \alpha^{11} \end{pmatrix}$$

cu soluţia $A_0 = \alpha^8,\ A_1 = \alpha^{11}$.

S-a obţinut polinomul $\sigma_{A\setminus B}(X) = \alpha^8 + \alpha^{11} X + X^2$. *Deci,*

$\sigma_A(X) = \sigma_B(X)\sigma_{A\setminus B}(X) = (\alpha + X)(\alpha^8 + \alpha^{11} X + X^2) = (\alpha + X)(\alpha^3 + X)(\alpha^5 + X)$.

Rezultă că erorile sunt localizate pe poziţiile $X_1 = \alpha,\ X_2 = \alpha^3,\ X_3 = \alpha^5$.

Mai departe se lucrează pentru determinarea valorilor erorilor, folosind algoritmul Peterson (secţiunea 8.2, *Capitolul* 8*) şi sindromurile originale. Trebuie rezolvat sistemul:*

$$\begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \alpha & \alpha^3 & \alpha^5 \\ \alpha^2 & \alpha^6 & \alpha^{10} \end{pmatrix} \begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \end{pmatrix} = \begin{pmatrix} \alpha^5 \\ \mathbf{0} \\ \alpha^3 \end{pmatrix}$$

cu soluţia $Y_1 = \alpha^{12},\ Y_2 = \mathbf{1},\ Y_3 = \alpha^3$.

Deci $v(X)$ *se decodifică în* $c(X) = v(X) + e(X) = (\alpha^{10} + \alpha^2 X^2 + \alpha^6 X^4 + \alpha^{14} X^5) + (\alpha^{12} X + X^3 + \alpha^3 X^5)$, *adică* $\mathbf{c} = \alpha^{10}\alpha^{12}\alpha^2\mathbf{1}\alpha^6\mathbf{10}\ldots\mathbf{0}$, *sau - în* $\hat{C}'$: 11101 11110 00101 10001 00110 10001 0...0.

9.6 Exerciţii

9.1 *Construiţi matricea generatoare şi listaţi toate cuvintele - cod ale codului RS din Exemplul 9.2. Scrieţi şi forma binară a acestor cuvinte.*

9.2 *Fie codul definit de* $g(X) = (1 + X)(\alpha + X)(\alpha^2 + X)(\alpha^3 + X)$, *bazat pe extensia Galois* $GF(2^3)$ *generată de* α, *rădăcină a polinomului* $1 + X + X^3$.

1. *Determinaţi* n, k, d *şi numărul de cuvinte - cod.*
2. *Construiţi o matrice generatoare şi o matrice de control a codului.*

3. *Codificaţi mesajele de informaţie* $\mathbf{10}\alpha^2$, $\mathbf{111}$, $\alpha^2\alpha^4\alpha^6$.

9.3 *Construiţi coduri RS cu următorii parametri:*
(a) $m = 2$, $d = 3$, $m_0 = 2$; (b) $m = 3$, $d = 3$, $m_0 = 2$;
(c) $m = 3$, $d = 5$, $m_0 = 0$; (d) $m = 4$, $d = 5$, $m_0 = 0$;
(e) $m = 5$, $d = 7$, $m_0 = 0$.

9.4 *Pentru fiecare cod determinat la exerciţiul anterior determinaţi* n, k *şi numărul de cuvinte - cod.*

9.5 *Fie codul RS de polinom generator* $g(X) = (1+X)(\alpha+X)(\alpha^2+X)(\alpha^3+X)(\alpha^4+X)(\alpha^5+X) \in GF(2^4)[X]$ *unde* α *este rădăcina polinomului* $1+X+X^4$. *Decodificaţi mesajele:*
(a) $\mathbf{0}\alpha^3\alpha\alpha^5\alpha^3\alpha^2\alpha^6\alpha^{10}\alpha\mathbf{000000}$; ($b$) $\mathbf{1}\alpha^4\alpha^2\alpha\mathbf{0010}\alpha\alpha^5\alpha^3\alpha^2\mathbf{0}\alpha^{10}\alpha$;
(c) $\alpha\mathbf{0}\alpha^7\mathbf{0}\alpha^{12}\alpha^3\alpha^3\mathbf{10000000}$.

9.6 *Fie codul RS generat de* $g(X) = (\alpha+X)(\alpha^2+X)(\alpha^3+X)(\alpha^4+X)$ *unde* $\alpha \in GF(2^4)$ *este rădăcina polinomului* $1+X+X^4$. *Decodificaţi mesajele:* $\mathbf{001}\alpha^8\mathbf{00}\alpha^5\mathbf{00000000}$;
$\mathbf{0}\alpha^{10}\mathbf{0}\alpha^6\alpha^{13}\mathbf{0}\alpha^8\alpha^{11}\alpha^3\alpha^5\mathbf{00000}$; $\alpha^4\mathbf{0100}\alpha^2\alpha^5\alpha^{12}al^{14}\mathbf{000000}$.

9.7 *Plecând de la codul RS din exerciţiul precedent, se formează codul scurt* $A(4)$ *de lungime* $n = 11$ *(şi* $k = 7$*). Decodificaţi mesajele:* $\mathbf{001}\alpha^8\mathbf{00}\alpha^5\mathbf{0000}$; $\mathbf{0}\alpha^{10}\mathbf{0}\alpha^6\alpha^{13}\mathbf{0}\alpha^8\alpha^{11}\alpha^3\alpha^5\mathbf{0}$; $\alpha^4\mathbf{0100}\alpha^2\alpha^5\alpha^{12}\alpha^{14}\mathbf{00}$.

9.8 *Arătaţi că transformata Fourier finită* $M_{n,n}$ *este inversabilă.*

9.9 *Fiind dat* $GF(2^3)$ *construit cu polinomul* $1+X+X^3$, *găsiţi matricea generatoare a unui cod DMS de lungime* 7, *pentru spaţiul funcţiilor definit pe* $S = GF(2^3) \setminus \{0\}$, *cu baza:*
(a) $\{X, X^2, X^3\}$; (b) $\{\mathbf{1}, X, X^2, X^3, X^4\}$; ($c$) $\{X, X^2, X^3\}$.
Arătaţi că toate aceste coduri sunt ciclice. Determinaţi polinomul generator pentru fiecare cod.

9.10 *Fiind dat* $GF(2^3)$ *construit cu* $1+X+X^3$, *determinaţi pentru fiecare* $G(X)$ *reprezentarea* $\mathbf{v_g}$:
(a) $G(X) = X + \alpha X^3$; (b) $G(X) = 1 + X^2 + X^4$.

9.11 *În aceleaşi condiţii din exerciţiul precedent, determinaţi* $G(X)$ *ştiind valorile lui* $\mathbf{v_g}$: $\mathbf{v_g} = \alpha^3\alpha\alpha^4\mathbf{0}\alpha^6\alpha^5\alpha^2$; $\mathbf{v_g} = \alpha^4\alpha^2\alpha\alpha^3\mathbf{0}\alpha^6\mathbf{1}$.

9.12 *Folosind codul definit în Exemplul 9.14, decodificaţi mesajele:*
(a) 11101 11110 11010 00111 11110 10100 10110 10100 0...0;
(b) 11000 00000 01010 11111 11011 00000 10001 00101 0...0;
(c) 00000 10000 10000 10000 00101 10100 11101 0...0.

Capitolul 10

Alte clase de coduri ciclice

10.1 Coduri Hamming ciclice binare

Să considerăm spaţiul nul al matricii

$$H = (\mathbf{1}\ \alpha\ \alpha^2\ \ldots\ \alpha^{n-1}),$$

unde $\alpha \in GF(2^p)$ este primitiv, deci $n = ord(\alpha) = 2^p - 1$.

Cum α poate fi considerat ca un vector coloană cu p componente, H este o matrice $p \times (2^p - 1)$ în care toate coloanele sunt distincte. Acesta este după cum se ştie (Capitolul 3, secţiunea 3.1) un $(2^p - 1, 2^p - p - 1)$ - cod Hamming binar, construit însă ca un cod ciclic.

Exemplul 10.1 *Fie $\alpha \in GF(2^3)$, α rădăcină a polinomului ireductibil $1 + X + X^3$. Am văzut că $ord(\alpha) = 7$, deci α este primitiv. Construim matricea*

$$H = (\mathbf{1}\ \alpha\ \alpha^2\ \alpha^3\ \alpha^4\ \alpha^5\ \alpha^6) = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Ea este matricea de control pentru un $(7, 4)$ - cod Hamming binar.

Deci codul studiat în Exemplul 3.1 (Capitolul 3) poate fi tratat şi ca un cod ciclic, de polinom generator $h(X) = 1 + X + X^3$.

Un cod Hamming ciclic binar se defineşte cerând ca orice polinom - cod $\{f(X)\}$ să admită ca rădăcină un element primitiv $\alpha \in GF(2^p)$. Generatorul unui asemenea cod este chiar polinomul minimal al lui α : $g(X) = m(X)$.

Deoarece şi α^2 este rădăcină a polinoamelor - cod (Teorema 7.9, Capitolul 7), codul Hamming ciclic binar este un caz particular de cod BCH binar cu $d = 3$, capabil să corecteze o eroare.

Codificarea se realizează folosind un circuit liniar cu $p = n - k$ elemente de înmagazinare (vezi secţiunea 5.4.2, Capitolul 5), corespunzător polinomului minimal $m(X)$. Calculul sindromului cuvântului recepţionat $\{r(X)\}$ - adică $\mathbf{r}H^T = r(\alpha)$ se obţine folosind acelaşi circuit de împărţire cu $m(X)$. Acest sindrom - în cazul în care este nenul, deci a intervenit o eroare simplă - este una din coloanele matricii H, adică una din valorile $\mathbf{1}, \alpha, \alpha^2, \ldots, \alpha^{n-1}$.

Pentru a afla care componentă a fost afectată de eroare, se poate folosi acelaşi circuit în felul următor: neglijăm intrarea şi lăsăm circuitul să funcţioneze, având iniţial în elementele de înmagazinare sindromul $r(\alpha)$, care este de forma

$$\alpha^j \ (0 < j \leq 2^p - 2).$$

La fiecare tact, circuitul înmulţeşte conţinutul elementelor de înmagazinare cu α; deci, după $n - j = 2^p - 1 - j$ momente, în elementele de înmagazinare se obţine

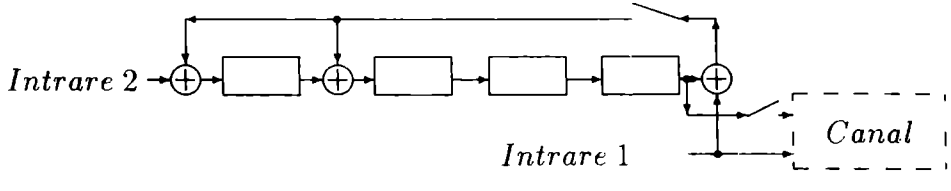
$$\alpha^{j+2^p-1-j} = \alpha^{2^p-1} = \alpha^n = \alpha^0 = \mathbf{1} = (1, 0, \ldots, 0).$$

Cum componentele cuvântului recepţionat sunt scrise în ordinea descrescătoare a puterilor (prima componentă este coeficientul termenului de rang maxim), înseamnă că va trebui să corectăm componenta $n - j$ din cuvânt, înlocuind 0 cu 1 sau 1 cu 0.

Exemplul 10.2 *Fie $\alpha \in GF(2^4)$, rădăcină a polinomului $1 + X + X^4$. Codul Hamming binar $(15, 11)$, obţinut folosind ca rădăcină pe α - deci de polinom generator $g(X) = 1 + X + X^4$, are matricea de control*

$$H = (\mathbf{1}\ \alpha\ \ldots\ \alpha^{14}) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Circuitul care realizează (în funcţie de necesităţi) codificarea, calculul sindromului şi corectarea erorilor simple (vezi şi Capitolul 5, secţiunea 5.4.2) este:



Codificarea se obţine introducând la "Intrare 1" mesajul de informaţie de lungime k, mesaj considerat ca un polinom de gradul $n-1$ cu primii coeficienţi (corespunzători termenilor de rang mare) elementele mesajului informaţional şi cu ultimii $n-k$ coeficienţi nuli.

La momentul iniţial în elementele de înmagazinare se află numai 0, circuitul feedback este închis, iar circuitul spre canal este deschis. De asemenea, "Intrare 2" este ignorată.

Simultan cu transmiterea pe canal, elementele de informaţie se introduc şi în circuitul liniar. După k momente, aici se află poziţiile de control.

În acest moment se decuplează feedback-ul, se cuplează legătura spre canal şi se ignoră cele două intrări. Timp de $n-k$ tacţi, elementele de control înmagazinate sunt trimise pe canal, imediat după mesajul de informaţie.

Pentru calculul sindromului: se neglijează canalul, se închide feedback-ul, iar în elementele de înmagazinare se găseşte iniţial $(0,0,\ldots,0)$. Cuvântul recepţionat este introdus prin "Intrare 2". După n tacţi, aici se va găsi $r(\alpha)$.

Pentru corectarea erorii: se neglijează cele două intrări, se închide feedback-ul şi se lasă să funcţioneze circuitul până când în elementele de înmagazinare se obţine vectorul $(1,0,\ldots,0)$. Numărul de tacţi scurşi dă indicele componentei din mesajul recepţionat, care trebuie corectată prin complementare.

Să considerăm spaţiul nul al matricii

$$H = \begin{pmatrix} \mathbf{1} & \alpha & \alpha^2 & & \alpha^{2^p-2} \\ 1 & 1 & 1 & & 1 \end{pmatrix},$$

unde $\alpha \in GF(2^p)$ este primitiv. H are $p+1$ linii şi $n = 2^p - 1$ coloane (pe prima coloană, primele p poziţii formează vectorul $\mathbf{1} = \alpha^0$, iar pe a $p+1$-a poziţie se află scalarul $1 \in Z_2$). Codul obţinut este codul Hamming binar extins, capabil să corecteze o eroare şi să detecteze două erori (vezi şi Algoritmul **B**, Capitolul 3).

Un asemenea cod se poate defini cerând ca orice cuvânt - cod $\{f(X)\}$ să aibă ca rădăcini α şi $\mathbf{1}$. Generatorul unui astfel de cod ciclic este $g(X) = (\mathbf{1}+X)m(X)$, unde $m(X)$ este polinomul minimal al lui α.

Codificarea în acest caz se face - ca mai sus - folosind circuitul cu $n-k+1 = p+1$ elemente de înmagazinare. Pentru calculul sindromului cuvântului recepţionat $\{r(X)\}$ se folosesc însă practic două circuite: unul

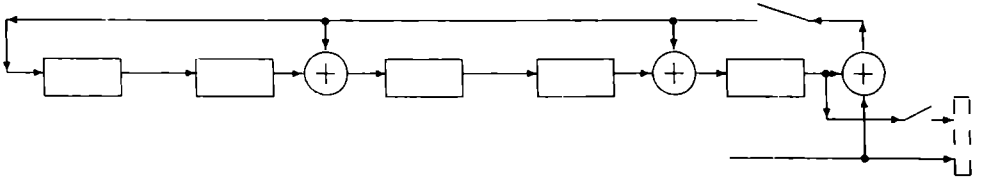
de împărţire la $m(X)$ (pentru calculul lui $r(\alpha)$) şi altul de împărţire la $\mathbf{1} + X$ (pentru $r(\mathbf{1})$). Discuţia este următoarea:

- Dacă $r(\alpha) = r(\mathbf{1}) = \mathbf{0}$, atunci nu a intervenit nici o eroare (sau avem o eroare nedetectabilă);
- Dacă $r(\alpha) \neq \mathbf{0}$, $r(\mathbf{1}) \neq \mathbf{0}$, avem o eroare simplă care se corectează cu circuitul de împărţire cu $m(X)$, după procedeul descris în Exemplul 10.2;
- Dacă $r(\alpha) \neq \mathbf{0}$, $r(\mathbf{1}) = \mathbf{0}$, atunci au fost detectate două erori.

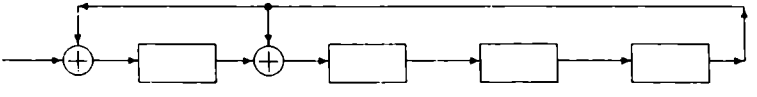
Exemplul 10.3 *Să extindem codul Hamming binar din Exemplul 10.2. Vom avea polinomul generator*

$$g(X) = (1 + X)(1 + X + X^4) = 1 + X^2 + X^4 + X^5.$$

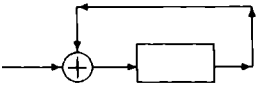
Circuitul liniar care codifică mesajele de informaţie (de lungime $k = 10$) este:



Sindromul se calculează cu ajutorul circuitului liniar de împărţire prin polinomul $m(X) = 1 + X + X^4$:



iar $r(\mathbf{1})$ se obţine din circuitul de împărţire la $\mathbf{1} + X$:



Circuitul liniar construit pentru $m(X)$ realizează şi corectarea erorilor simple.

10.2 Coduri Hamming ciclice nebinare

Codurile Hamming binare au fost definite sub formă liniară, ca un caz particular de coduri Reed - Muller sau sub formă de coduri ciclice. În

această ultimă situaţie, ele se pot implementa cu ajutorul circuitelor liniare şi – fapt important – admit o variantă de generalizare naturală la cazul nebinar.

Definiţia 10.1 *Fie $\alpha \in GF(q^p)$ un element primitiv. Se defineşte codul Hamming ciclic C astfel:*

$$\{f(X)\} \in C \iff f(X) \text{ are rădăcina } \beta = \alpha^{q-1}$$

sau – echivalent – C are matricea de control $H = (\mathbf{1}\ \beta\ \beta^2 \dots \beta^{n-1})$ unde $n = ord(\beta)$.

În cazul $q = 2$ se obţine definiţia codurilor Hamming binare ciclice.

Din

$$\beta^n = \mathbf{1} = (\alpha)^{q^p-1} = \left(\alpha^{q-1}\right)^{\frac{q^p-1}{q-1}} \text{ rezultă } n = \frac{q^p-1}{q-1}.$$

Se observă imediat că H are un număr maxim de coloane distincte şi nenule.

Propoziţia 10.1 $(n, q-1) = 1$ *dacă şi numai dacă* $(p, q-1) = 1$.

Demonstraţie: Din $q-1 | q^j - 1$, $(1 \le j \le p)$ rezultă $q^j = (q-1)s_j + 1$. Deci

$$n = \frac{q^p-1}{q-1} = q^{p-1} + q^{p-2} + \dots + q + 1 = (q-1)(s_{p-1} + \dots + s_1) + p$$

şi atunci

$$(p, q-1) = 1 \iff (n, q-1) = 1. \qquad \square$$

Teorema 10.1 *Spaţiul nul al matricii $H = (\mathbf{1}\ \beta\ \beta^2 \dots \beta^{n-1})$ unde $\beta = \alpha^{q-1}$, $n = \dfrac{q^p-1}{q-1}$ şi $\alpha \in GF(q^p)$ primitiv, are distanţa minimă $d \ge 3$ dacă şi numai dacă $(n, q-1) = 1$.*

Demonstraţie: "$\Longleftarrow$": Să presupunem că există două coloane liniar dependente în matricea H. Deci $\beta^i = a\beta^j$ cu $a \in Z_q$, adică $\beta^{i-j} = a$.

Elementele nenule din Z_q sunt rădăcinile ecuaţiei $X^{q-1} - 1 = 0$ (Teorema 7.3, Capitolul 7). Mai mult, ele sunt chiar primele $q-1$ puteri ale lui α^n. Într-adevăr, pentru orice $s = 0, 1, \dots, q-2$ avem

$$(\alpha^{ns})^{q-1} = \left(\alpha^{q^p-1}\right)^s = \mathbf{1}^s = \mathbf{1}.$$

Deci pentru orice $a \in Z_q \setminus \{0\}$ există s $(0 \le s < q-1)$ cu $a = \alpha^{ns}$. Se poate scrie atunci $\beta^{i-j} = \alpha^{ns}$, adică $\alpha^{(i-j)(q-1)} = \alpha^{ns} \Rightarrow (i-j)(q-1) = ns$.

Dar $s < q-1$ iar $(n, q-1) = 1$, deci $i = j$, de unde rezultă $a = 1$, $\beta^i = \beta^j$, imposibil pentru H are toate coloanele distincte. Rezultă că

orice două coloane din H sunt liniar independente, adică (Teorema 2.4, Capitolul 2) $d \geq 3$.

"$\Longrightarrow$": Dacă $q-1$ şi n nu sunt prime între ele, atunci ecuaţia anterioară admite o soluţie cu $i \neq j$; deci există două coloane distincte liniar dependente β^i şi β^j în matricea H, ceea ce contrazice faptul că $d \geq 3$. □

Pentru operaţia de codificare se foloseşte circuitul de împărţire cu polinomul minimal $m(X)$ al lui β (care aici este chiar polinomul generator al codului Hamming ciclic nebinar C). Acelaşi circuit va calcula sindromul $r(\beta)$ al polinomului recepţionat $\{r(X)\}$.

În cazul în care a intervenit o eroare, $r(\beta)$ este de forma $b\beta^j$ (eroarea b a apărut pe poziţia β^j). Funcţionarea secvenţială a circuitului liniar (ignorând intrarea) revine la înmulţirea succesivă cu β. Deci, după $n-j$ tacţi, în elementele de înmagazinare se obţine vectorul $(b\ 0\ \ldots\ 0)$. Atunci, din componenta $n-j$ a cuvântului recepţionat se va scădea eroarea b. De observat că - deoarece coloanele lui H sunt nenule şi distincte - primul tact în care se obţine un vector cu toate componentele nule înafară de prima, este $n-j$.

Exemplul 10.4 *Să considerăm $p = q = 3$ (astfel, $(m, q-1) = 1$). Folosind rădăcina α a polinomului primitiv $1 + 2X + X^3$ vom genera extensia $GF(3^3)$. Fie $\beta = \alpha^{q-1} = \alpha^2$. Deoarece $\alpha^{3^3-1} = \alpha^{26} = 1$, avem $ord(\beta) = 13$, deci $n = 13$. Pentru a construi polinomul minimal $m(X)$ al lui β, să observăm (Secţiunea 7.3) că el are ca rădăcini α^2, α^6 şi α^{18} ($\alpha^{54} = \alpha^2$). Deci $m(X) = a + bX + cX^2 + X^3$. Punând condiţia $m(\alpha^2) = a + b\alpha^2 + c\alpha^4 + \alpha^6 = \mathbf{0}$, obţinem ecuaţia matricială:*

$$a\begin{pmatrix}1\\0\\0\end{pmatrix} + b\begin{pmatrix}0\\0\\1\end{pmatrix} + c\begin{pmatrix}0\\2\\1\end{pmatrix} + \begin{pmatrix}1\\1\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\end{pmatrix}$$

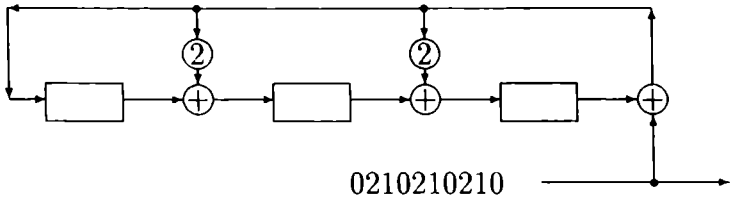
cu soluţia $a = 2$, $b = c = 1$. Deci $m(X) = 2 + X + X^2 + X^3$.

Codul Hamming ciclic ternar de polinom generator $g(X) = m(X) = 2 + X + X^2 + X^3$ are matricea de control

$$H = (\mathbf{1}\ \alpha^2\ \alpha^4\ \ldots\ \alpha^{24}) = \begin{pmatrix} 1 & 0 & 0 & 1 & 2 & 0 & 2 & 0 & 1 & 1 & 1 & 2 & 1 \\ 0 & 0 & 2 & 1 & 0 & 1 & 0 & 2 & 2 & 2 & 1 & 2 & 1 \\ 0 & 1 & 1 & 1 & 2 & 1 & 1 & 0 & 0 & 1 & 2 & 0 & 2 \end{pmatrix}$$

Am definit astfel $(13, 10)$ - codul Hamming ternar, evocat în Exemplul 3.5 (Capitolul 3) relativ la Problema Pronosportului.

Să considerăm mesajul de informaţie 0120120120. Pentru operaţia de codificare se va folosi circuitul (pentru detalii, a se revedea Capitolul 5, secţiunea 5.4.2):

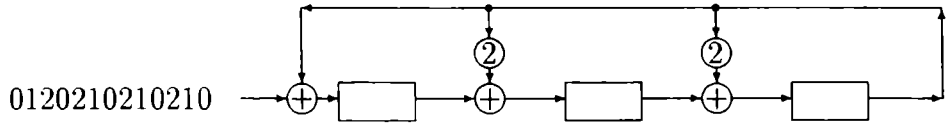


(s-a ţinut cont că se lucrează în Z_3, deci $-1 = 2$, $-2 = 1$). În primii 10 *tacţi de funcţionare, biţii de informaţie sunt trimişi pe canal şi - simultan - intră în circuitul liniar, a cărui funcţionare este (pe prima coloană se află informaţia, iar pe celelalte trei - conţinutul elementelor de înmagazinare):*

0	0	0	0		2	2	1	2
1	1	2	2		0	2	0	2
2	1	0	1		1	0	2	0
0	1	0	2		2	2	1	0
1	0	1	0		0	0	2	1

În elementele de înmagazinare au rămas caracterele de control 021. *Ele sunt trimise prin canal la tacţii* $11 - 13$ *de la dreapta spre stânga, cu semn schimbat: în ordine* $-1 = 2$, $-2 = 1$, $-0 = 0$. *Deci cuvântul - cod transmis a fost* $\mathbf{a} = 0120120120210$. *Se verifică imediat că* $\mathbf{a}H^T = \mathbf{0}$.

La recepţie, primele 10 *caractere sunt cele de informaţie. Pentru a verifica dacă au apărut sau nu erori, se foloseşte acelaşi circuit, dar cu altă intrare:*



Funcţionarea acestui circuit liniar timp de 13 *tacţi, este dată de tabelul:*

	0	0	0		0	1	2	0
0	0	0	0		1	1	1	2
1	1	0	0		2	1	2	2
2	2	1	0		0	2	2	0
0	0	2	1		2	2	2	2
1	2	2	1		1	0	0	0
2	0	1	1		0	0	0	0

Sindromul $a(\alpha^2)$ *obţinut este* 000, *deci nu a apărut nici o eroare. Dacă s-ar fi recepţionat însă cuvântul* $\mathbf{a}' = 0120120100210$, *după funcţionarea*

circuitului timp de 13 *tacţi s-ar fi ajuns în elementele de înmagazinare la* 220. *Cum acesta este nenul, circuitul se lasă să funcţioneze în continuare (ignorând intrarea) până se ajunge la un sindrom de forma* $b00$ *cu* $b \neq 0$. *Mai exact, funcţionarea circuitului va fi (pe prima coloană se numără tacţii):*

0	2	2	0		5	1	0	1
1	0	2	2		6	1	0	2
2	2	1	0		7	2	2	1
3	0	2	1		8	1	1	1
4	1	2	1		9	1	0	0

Deci, după 9 *tacţi s-a ajuns la* 100. *Corecţia se face astfel: din componenta a* 9 *- a a cuvântului recepţionat se scade* $b = 1$, *obţinându-se* $0 - 1 = 2 \ (mod\ 3)$.

Observaţia 10.1 *Definiţia codurilor Hamming ciclice nebinare nu este echivalentă cu cea a codurilor Hamming nebinare dată în Secţiunea* 3.1.1 *(Capitolul* 3*). Astfel,* $(4, 2)$ *- codul Hamming ternar definit în Exemplul* 3.4, *cu matricea de control*

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

este format din 9 *cuvinte*

$$\{0000,\ 1012,\ 2021,\ 0111,\ 1120,\ 2102,\ 0222,\ 1201,\ 2210\},$$

care - evident - nu formează un cod ciclic.

Deci între cele două definiţii ale codurilor Hamming nebinare (liniare şi ciclice) există doar o relaţie de incluziune strictă (orice cod Hamming ciclic este un cod Hamming liniar, dar nu şi invers).

10.3 Coduri Goppa

Această clasă de coduri a fost introdusă de Goppa în 1970. Deşi în general sunt coduri neciclice, ele constituie o generalizare a codurilor BCH şi constituie o punte între teoria codurilor şi criptografie. Din acest motive, prezentarea lor este deosebit de utilă.

Fie q un număr prim şi $g(X) \in Z_q[X]$. Dacă $a \in Z_q$ este un element cu proprietatea $g(a) \neq 0$, vom nota

$$\frac{1}{X - a} = -\frac{g(X) - g(a)}{X - a} g(a)^{-1}.$$

Propoziţia 10.2 *În algebra polinoamelor din $Z_q[X]$ modulo $g(X)$, polinomul $\frac{1}{X-a}$ este inversul lui $X-a$.*

Demonstraţie: Faptul că $\frac{1}{X-a}$ este polinom se verifică imediat. Pe de altă parte, $(X-a)\frac{1}{X-a} = [g(a)-g(X)]g(a)^{-1} = 1-g(a)^{-1}g(X) \equiv 1 \ (mod\ g(X))$. □

Definiţia 10.2 *Un cod Goppa de lungime n peste Z_q este determinat prin:*

1. *Un polinom $g(X) \in Z_q[X]$ de grad t $(t \geq 2)$, definit peste o extensie $GF(q^m)$ a lui Z_q;*

2. $\mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_n} \in GF(q^m)$ *cu* $g(\mathbf{a_i}) \neq \mathbf{0}$ $(1 \leq i \leq n)$.

Codul conţine toate cuvintele $\mathbf{v} = v_1 v_2 \ldots v_n \in Z_q^n$ pentru care este verificată egalitatea $\sum_{i=1}^{n} \frac{v_i}{X-\mathbf{a_i}} = \mathbf{0}$.

Teorema 10.2 *Codul Goppa conţine numai cuvintele $\mathbf{v} = v_1 v_2 \ldots v_n$ $(v_i \in Z_q)$ pentru care*

$$\sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} \mathbf{a_i}^s = \mathbf{0}, \quad s = 0, 1, \ldots, t-1. \qquad \textit{unde } t = grad(g(X)).$$

Demonstraţie: Să explicităm polinomul $\frac{1}{X-\mathbf{a}}$. Dacă $g(X) = g_0 + g_1 X + \ldots + g_t X^t$ $(g_t \neq 0)$, atunci pentru orice $\mathbf{a} \in GF(q^m)$ cu $g(\mathbf{a}) \neq \mathbf{0}$, are loc descompunerea

$$\frac{g(X)-g(\mathbf{a})}{X-\mathbf{a}} = g_t X^{t-1} + (\mathbf{a}g_t + g_{t-1})X^{t-2} + (\mathbf{a}^2 g_t + \mathbf{a}g_{t-1} + g_{t-2})X^{t-3} + \ldots + (\mathbf{a}^{t-1} g_t + \mathbf{a}^{t-2} g_{t-1} + \ldots + g_1).$$

Deci coeficientul lui X^{p-1} $(1 \leq p \leq t)$ în polinomul $\frac{1}{X-\mathbf{a}}$ este

$$-g(\mathbf{a})^{-1} \sum_{j=p}^{t} \mathbf{a}^{t-j} g_j.$$

Ecuaţia caracteristică $\sum_{i=1}^{n} \frac{v_i}{X-\mathbf{a_i}} = \mathbf{0}$ a codurilor Goppa implică faptul că toţi coeficienţii lui X^{p-1} $(1 \leq p \leq t)$ sunt nuli. Deci

$$\sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} \sum_{j=p}^{t} \mathbf{a_i}^{t-j} g_j = \mathbf{0} \quad (1 \leq p \leq t).$$

Cazul $p = t$ conduce la $g_t \sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} = \mathbf{0}$ şi - deoarece $g_t \neq 0$ - se obţine $\sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} = \mathbf{0}$.

Din cazul $p = t-1$ se obţine $g_t \sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} a_i + g_{t-1} \sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} = \mathbf{0}$.

Cum a doua sumă este zero, iar $g_t \neq 0$, va rezulta $\sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} \mathbf{a_i} = \mathbf{0}$ ş.a.m.d.

Am obţinut echivalenţa

$$\sum_{i=1}^{n} \frac{v_i}{X - \mathbf{a_i}} = \mathbf{0} \iff \sum_{i=1}^{n} v_i g(\mathbf{a_i})^{-1} \mathbf{a_i}^s = 0 \quad (0 \leq s \leq t-1). \ \square$$

Corolarul 10.1 *Codul Goppa determinat de un polinom $g(X)$ de gradul t este un cod liniar cu matricea de control*

$$H = \begin{pmatrix} \frac{1}{g(\mathbf{a_1})} & \frac{1}{g(\mathbf{a_2})} & \frac{1}{g(\mathbf{a_3})} & & \frac{1}{g(\mathbf{a_n})} \\ \frac{\mathbf{a_1}}{g(\mathbf{a_1})} & \frac{\mathbf{a_2}}{g(\mathbf{a_2})} & \frac{\mathbf{a_3}}{g(\mathbf{a_3})} & & \frac{\mathbf{a_n}}{g(\mathbf{a_n})} \\ & & \vdots & & \\ \frac{\mathbf{a_1}^{t-1}}{g(\mathbf{a_1})} & \frac{\mathbf{a_2}^{t-1}}{g(\mathbf{a_2})} & \frac{\mathbf{a_3}^{t-1}}{g(\mathbf{a_3})} & & \frac{\mathbf{a_n}^{t-1}}{g(\mathbf{a_n})} \end{pmatrix}.$$

Demonstraţie: Scriind sub formă matricială relaţiile din enunţul Teoremei 10.2, avem $\mathbf{v}H^T = \mathbf{0}$, unde H este definită mai sus. $\square$

Exemplul 10.5 *Să considerăm extensia $GF(2^3)$, generată folosind rădăcina α a polinomului ireductibil $1 + X + X^3 \in Z_2[X]$. Definim $g(X) = 1 + X + X^2$ şi luăm drept elemente $\mathbf{a_i}$ cele 8 componente ale lui $GF(2^3)$. Valorile lui $g(X)$ sunt*

a	$\mathbf{0}$	$\mathbf{1}$	α	α^2	α^3	α^4	α^5	α^6
$g(a)$	$\mathbf{1}$	$\mathbf{1}$	α^5	α^3	α^5	α^6	α^6	α^3
$g(a)^{-1}$	$\mathbf{1}$	$\mathbf{1}$	α^2	α^4	α^2	α	α	α^4

Se obţine un cod Goppa de lungime 8 cu matricea de control

$$H = \begin{pmatrix} \frac{1}{g(\mathbf{0})} & \frac{1}{g(1)} & \frac{1}{g(\alpha)} & \cdots & \frac{1}{g(\alpha^6)} \\ \mathbf{0} & \frac{1}{g(1)} & \frac{\alpha}{g(\alpha)} & & \frac{\alpha^6}{g(\alpha^6)} \end{pmatrix} =$$

$$= \begin{pmatrix} \mathbf{1} & \mathbf{1} & \alpha^2 & \alpha^4 & \alpha^2 & \alpha & \alpha & \alpha^4 \\ \mathbf{0} & \mathbf{1} & \alpha^3 & \alpha^6 & \alpha^5 & \alpha^5 & \alpha^6 & \alpha^3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Acest cod este format din numai patru cuvinte:

$$\{00000000,\quad 00111111,\quad 11001011,\quad 11110100\}.$$

El are distanţa minimă 5; s-a obţinut deci un $(8,2)$ *- cod binar corector de două erori. De remarcat că acesta nu este un cod ciclic.*

Observaţia 10.2 *Să considerăm un cod BCH definit prin rădăcinile*

$$\alpha^{m_0}, \alpha^{m_0+1}, \ldots, \alpha^{m_0+d-2},$$

unde $\alpha \in GF(q^m)$ *are ordinul* n*. Acest cod se poate scrie ca un cod Goppa astfel: Se ia polinomul* $g(X) = X^{m_0+d-2}$ *şi elementele* $\alpha^0, \alpha^{-1}, \ldots,$ $\alpha^{-(n-1)}$*. Vom avea pentru orice* $i = 0, 1, \ldots, d-2$ *şi* $p = 0, 1, \ldots, n-1$*:*

$$\frac{(\alpha^{-p})^i}{(\alpha^{-p})^{m_0+d-2}} = (\alpha^p)^{m_0+d-2-i} = (\alpha^{m_0+d-2-i})^p.$$

Pe baza aceasta, codul Goppa rezultat are matricea de control

$$H = \begin{pmatrix} 1 & \alpha^{m_0+d-2} & (\alpha^{m_0+d-2})^2 & (\alpha^{m_0+d-2})^{n-1} \\ 1 & \alpha^{m_0+d-3} & (\alpha^{m_0+d-3})^2 & (\alpha^{m_0+d-3})^{n-1} \\ & & \vdots & \\ 1 & \alpha^{m_0} & (\alpha^{m_0})^2 & (\alpha^{m_0})^{n-1} \end{pmatrix}$$

care coincide cu matricea de control a codului BCH considerat, în care s-a efectuat o inversare a liniilor.

Se poate arăta (Van Lint) că singurele coduri Goppa ciclice sunt codurile BCH.

Propoziţia 10.3 *Codul Goppa binar de lungime* n*, determinat de un polinom de grad* t *peste* $GF(2^m)$ *are* $k \geq n - mt$ *simboluri de informaţie şi distanţa minimă* $\geq t + 1$*.*

Demonstraţie: Matricea de control a codului Goppa binar are t linii în $GF(2^m)$, deci mt linii scrise în binar. Rezultă că numărul $n - k$ de simboluri de control este cel mult mt : $n - k \leq mt$.

Pe de-altă parte, se arată uşor că orice combinaţie liniară de t coloane din H este liniar independentă. □

Deci un cod Goppa este capabil să corecteze orice combinaţie de maxim $t/2$ erori independente. Această capacitate creşte semnificativ în cazul codurilor ireductibile.

Definiţia 10.3 *Fie $g(X)$ un polinom ireductibil peste $GF(q^m)$. Codul Goppa determinat de $g(X)$ şi de toate elementele lui $GF(q^m)$ se numeşte ireductibil.*

Ideea de "ireductibil" este similară codurilor ciclice ireductibile (Capitolul 7, paragraful 7.4). Un cod Goppa este ireductibil dacă nu are subcoduri Goppa proprii.

Codul prezentat în Exemplul 10.5 este un cod Goppa ireductibil. Despre codurile Goppa ireductibile se poate prezenta următorul rezultat:

Teorema 10.3 *Un cod Goppa binar ireductibil, determinat de un polinom de gradul t poate corecta cel puţin t erori independente.*

Pentru demonstraţie va trebui să facem o construcţie suplimentară.

Definiţia 10.4 *Derivata formală a polinomului $a(X) = a_0 + a_1X + \ldots + a_nX^n$ este definită prin $a'(X) = a_1 + 2a_2X + 3a_3X^2 + \ldots + na_nX^{n-1}$.*

Lema 10.1 *Pentru derivata formală sunt adevărate egalităţile:*

1. $(a(X) + b(X))' = a'(X) + b'(X)$;
2. $(a(X)b(X))' = a'(X)b(X) + a(X)b'(X)$

Demonstraţie: Exerciţiu.

Observaţia 10.3 *Noţiunea de "derivată formală" a mai fost evocată în demonstrarea Propoziţiei 7.9 (Capitolul 7).*

Lema 10.2 *Într-un corp de caracteristică 2, derivata formală a unui polinom este un pătrat perfect.*

Demonstraţie: Deoarece pentru n par $(X^n)' = nX^{n-1} = 0$, derivata formală a unui polinom va avea numai puteri pare. Cum fiecare coeficient se poate scrie ca un pătrat perfect (se lucrează într-un corp de caracteristică 2), avem $f'(X) = f_0^2 + f_2^2X^2 + f_4^2X^4 + \ldots$ şi, dacă se ia $g(X) = f_0 + f_2X + f_4X^2 + \ldots$, avem $f'(X) = g^2(X)$. □

Să revenim acum la *Demonstraţia Teoremei 10.3*: Fie $g(X) \in Z_2[X]$ un polinom ireductibil de grad t ($t \geq 2$) peste $GF(2^m) = \{\mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_n}\}$

$(n = 2^m)$, α o rădăcină a lui g, şi $\mathbf{v} = v_1 v_2 \ldots v_n$ un cuvânt al codului Goppa, $w(\mathbf{v}) = s$. Notăm $v_{i_1}, v_{i_2}, \ldots, v_{i_s}$ biţii egali cu 1 din $\mathbf{v}$. Deoarece $\mathbf{v}$ este cuvânt - cod, avem $\sum_{i=1}^{n} \frac{v_i}{X - \mathbf{a_i}} = \sum_{p=1}^{s} \frac{1}{X - \mathbf{a_{i_p}}} = \mathbf{0}$. Polinomul $f(X) = (X - \mathbf{a}_{i_1})(X - \mathbf{a}_{i_2}) \ldots (X - \mathbf{a}_{i_s})$ nu este divizibil cu $g(X)$ (deoarece $g(X)$ este ireductibil). Rezultă că, în algebra claselor de resturi $Z_2[X]/g(X)$, $f(\alpha) \neq 0$ (Teorema 4.1) deci admite invers. Deci există un polinom $r(X)$ cu $\{f(\alpha)\}\{r(\alpha)\} = 1$.

Să calculăm derivata formală $f'(X) = \sum_{p=1}^{s} \frac{f(X)}{X - \mathbf{a_{i_p}}}$. Făcând $X = \alpha$ şi înmulţind relaţia cu $r(\alpha)$, obţinem

$$\{f'(\alpha)\}\{r(\alpha)\} = \sum_{p=1}^{s} \frac{\{f(\alpha)\}\{r(\alpha)\}}{\alpha - \mathbf{a_{i_p}}} = \sum_{p=1}^{s} \frac{1}{\alpha - \mathbf{a_{i_p}}} = \mathbf{0}\ (mod\ g(X)).$$

Aceasta înseamnă că $g(X) | f'(X) r(X)$. Dar cum $g(X)$ este ireductibil şi nu divide $r(X)$ (deoarece $r(\alpha) \neq 0$), rezultă $g(X) | f'(X)$.

Deoarece suntem într-un corp de caracteristică 2, conform Lemei 10.2 derivata formală este un pătrat perfect: $f'(X) = h^2(X)$. Din faptul că $g(X)$ este divizor ireductibil al lui $h^2(X)$ rezultă $g(X) | h(X)$. Deci $g^2(X) | h^2(X) = f'(X)$.

Se ştie că $grad(f'(X)) = s - 1$ şi $s \geq 1$; deci $grad(f'(X)) \geq grad(g^2(X))$, adică $s - 1 \geq 2t$. Cum ponderea oricărui cuvânt - cod este $\geq 2t + 1$, rezultă că distanţa minimă a codului este cel puţin $2t + 1$, deci codul corectează sigur t erori independente. □

Exemplul 10.6 *Fie $GF(2^4)$ generat de rădăcina α a polinomului $1 + X + X^4 \in Z_2[X]$. Vom construi un cod Goppa ireductibil folosind elementele acestei extensii şi polinomul $g(X) = 1 + X + X^3$. Valorile polinomului sunt:*

a	$g(a)$	$\frac{1}{g(a)}$	a	$g(a)$	$\frac{1}{g(a)}$	a	$g(a)$	$\frac{1}{g(a)}$	a	$g(a)$	$\frac{1}{g(a)}$
$\mathbf{0}$	$\mathbf{1}$	$\mathbf{1}$	α^3	α^4	α^{11}	α^7	α^5	α^{10}	α^{11}	α^{10}	α^5
$\mathbf{1}$	$\mathbf{1}$	$\mathbf{1}$	α^4	α^{13}	α^2	α^8	α^{11}	α^4	α^{12}	α	α^{14}
α	α^7	α^8	α^5	α^5	α^{10}	α^9	α^2	α^{13}	α^{13}	α^5	α^{10}
α^2	α^{14}	α	α^6	α^8	α^7	α^{10}	α^{10}	α^5	α^{14}	α^{10}	α^5

Matricea de control a codului este dată în Figura 10.1:

Acesta este un (16, 4) - cod liniar care are 16 cuvinte - cod:

A = {0000000000000000, 0001100011010110, 0010011100101001, 0011111111111111, 0100100110101111, 0101000101111001,

Figura 10.1:

$$H = \begin{pmatrix} \mathbf{11}\alpha^8\alpha & \alpha^{11}\alpha^2\alpha^{10}\alpha^7 & \alpha^{10}\alpha^4\alpha^{13}\alpha^5 & \alpha^5\alpha^{14}\alpha^{10}\alpha^5 \\ \mathbf{01}\alpha^9\alpha^3 & \alpha^{14}\alpha^6\mathbf{1}\alpha^{13} & \alpha^2\alpha^{12}\alpha^7\mathbf{1} & \alpha\alpha^{11}\alpha^8\alpha^4 \\ \mathbf{01}\alpha^{10}\alpha^5 & \alpha^2\alpha^{10}\alpha^5\alpha^4 & \alpha^9\alpha^5\alpha\alpha^5 & \alpha^{12}\alpha^8\alpha^6\alpha^3 \end{pmatrix} =$$

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

0110111010000110, 0111011001010000, 1000001010011011, 1001101001001101, 1010010110110010, 1011110101100100, 1100101100110100, 1101001111100010, 1110110000011101, 1111010011001011}.

Distanţa minimă este $d = 7$, deci codul poate corecta orice combinaţie de maxim 3 erori independente.

Exemplul 10.7 *Să considerăm corpul $GF(2^3)$, generat de o rădăcină a lui $g(X) = 1 + X + X^3$. Deoarece $g(X)$ are 3 rădăcini în $GF(2^3)$, el nu va avea nici o rădăcină în corpul $GF(2^5)$ ($GF(2^5)$ nu este o extensie a lui $GF(2^3)$), deci aici $g(X)$ este ireductibil.*

Codul Goppa ireductibil corespunzător are $n = 32$, $k \geq 32 - 5 \cdot 3 = 17$, $d \geq 7$; deci este un $(32, 17)$ - cod corector de 3 erori.

O proprietate remarcabilă a codurilor Goppa este aceea că ele sunt *asimptotic bune* .

Definiţia 10.5 *Fie $C_1, C_2, \ldots$ o secvenţă de (n_i, k_i) coduri de distanţe minime d_i. Ele sunt asimptotic bune dacă există două numere reale pozitive α, β, astfel ca $\lim\limits_{i\to\infty} \frac{k_i}{n_i} = \alpha$, $\lim\limits_{i\to\infty} \frac{d_i}{n_i} = \beta$.*

Altfel spus, o secvenţă de coduri este asimptotic bună dacă odată cu creşterea lungimii cuvintelor - cod, rata de informaţie şi capacitatea de corectare a erorilor se stabilizează la valori pozitive.

De exemplu, codurile BCH nu au această proprietate, deci nu pot fi folosite plecând de la fixarea arbitrară a lungimii cuvintelor - cod. În schimb, MacWilliams şi Sloane demonstrează că există o secvenţă asimptotic bună de coduri Goppa binare ireductibile. Din păcate, această demonstraţie este doar existenţială, nu algoritmică.

Decodificarea codurilor Goppa se face în manieră similară decodificării codurilor BCH. Astfel, fie C un cod Goppa definit de $g(X) \in Z_q[X]$ $(grad(g(X)) = t)$ şi $\mathbf{a_1}, \mathbf{a_2}, \ldots \mathbf{a_n} \in GF(q^m)$; considerăm cuvântul - cod $\mathbf{v} = v_0 v_1 \ldots v_{n-1}$ şi fie $\mathbf{r} = r_0 r_1 \ldots r_{n-1}$ cuvântul recepţionat după transmiterea lui $\mathbf{v}$.

Vom nota cu $\mathbf{e} = \mathbf{r} - \mathbf{v} = e_0 e_1 \ldots e_{n-1})$ secvenţa - eroare.

Pentru $t/2 \leq w(\mathbf{e}) \leq t$ se poate folosi un algoritm de decodificare uzual (definit pentru codurile liniare). În practică însă - deoarece codurile Goppa sunt asimptotic bune - se folosesc coduri (nu neapărat ireductibile) cu $t = grad(g(X))$ mare; conform Propoziţiei 10.3, ele pot corecta orice combinaţie de maxim $t/2$ erori independente. Vom prezenta un algoritm de complexitate mică, specific codurilor Goppa, care funcţionează în cazul $w(\mathbf{e}) < t/2$.

Fie $M = \{i | e_i \neq 0\}$, $card(M) = j < t/2$; considerăm sindromul polinomial

$$\{s(X)\} = \sum_{i=0}^{n-1} \frac{e_i}{X - \mathbf{a_i}} \pmod{g(X)}.$$

De remarcat că $\{s(X)\}$ se poate determina prin calcul la primirea cuvântului $\mathbf{r}$. Se definesc polinoamele $\sigma(X)$ (de localizare a erorii) şi $\omega(X)$ prin

$$\sigma(X) = \prod_{i \in M} (X - \mathbf{a_i}), \qquad \omega(X) = \sum_{i \in M} e_i \prod_{k \in M \setminus \{i\}} (X - \mathbf{a_k}).$$

Din definiţie rezultă că $\sigma(X)$ şi $\omega(X)$ sunt prime între ele. În plus, $grad(\sigma(X)) = j$, $grad(\omega(X)) < j$. Între ele există relaţia

$$\{s(X)\}\{\sigma(X)\} = \sum_{i=0}^{n-1} \frac{e_i}{X - \mathbf{a_i}} \prod_{i \in M} (X - \mathbf{a_i}) \equiv \{\omega(X)\} \pmod{g(X)}. \quad (1)$$

$\sigma(X)$ şi $\omega(X)$ sunt polinoamele normate de grad minim care verifică relaţiile de sus. Într-adevăr, să presupunem că există $\sigma_1(X)$ un polinom

nenul normat de grad $< j$ şi $\omega_1(X)$ polinomul corespunzător, astfel ca

$$\{s(X)\}\{\sigma_1(X)\} = \{\omega_1(X)\} \pmod{g(X)}.$$

Fie $d(X) = cmmdc(s(X), g(X))$. Din relaţiile

$s(X)\sigma(X) = a(X)g(X) + \omega(X)$, $s(X)\sigma_1(X) = b(X)g(X) + \omega_1(X)$,

rezultă că $d(X)$ va divide şi polinoamele $\omega(X)$, $\omega_1(X)$. Deci,

$$\left\{\frac{s(X)}{d(X)}\right\}\{\sigma(X)\} - \left\{\frac{\omega(X)}{d(X)}\right\} \equiv 0,$$

$$\left\{\frac{s(X)}{d(X)}\right\}\{\sigma_1(X)\} - \left\{\frac{\omega_1(X)}{d(X)}\right\} \equiv 0 \qquad \left(mod\ \frac{g(X)}{d(X)}\right)$$

sau

$$\left\{\frac{s(X)}{d(X)}\right\}\frac{\{\omega_1(X)\sigma(X) - \omega(X)\sigma_1(X)\}}{\{d(X)\}} \equiv 0 \pmod{g(X)}.$$

Deci $\{\omega_1(X)\sigma(X) - \omega(X)\sigma_1(X)\} \equiv 0 \pmod{g(X)}$. Deoarece polinomul din membrul stâng are grad mai mic decât t, rezultă că membrul stâng este 0. Apoi, cum $cmmdc(\sigma(X), \omega(X)) = 1$, deducem $\sigma(X)|\sigma_1(X)$ - ceea ce contrazice faptul că $grad(\sigma_1(X)) < grad(\sigma(X))$ şi $\{\sigma_1(X)\} \not\equiv 0$.

Rezumând, procedeul de decodificare rapidă a mesajelor la codurile Goppa are loc astfel:

1. Din mesajul recepţionat $\mathbf{r}$ se calculează sindromul $s(X)$;
2. Se determină polinomul normat $\sigma(X)$ de grad minim $< t/2$ care verifică relaţia (1) (un algoritm eficient a fost dat de Berlekamp).

   Din rezolvarea ecuaţiei $\sigma(X) = 0$ se află locaţiile erorilor;
3. Se determină e_i $(i \in M)$ din relaţiile

$$\omega(\mathbf{a_i}) = e_i \prod_{j \in M \setminus \{i\}} (\mathbf{a_i} - \mathbf{a_j}).$$

   Pe baza acestor valori se construieşte secvenţa - eroare $\mathbf{e}$;
4. Cuvântul decodificat este $\mathbf{v} = \mathbf{r} - \mathbf{e}$.

Exemplul 10.8 *Codul Goppa binar ireductibil definit în Exemplul 10.6 poate corecta până la 3 erori independente (Teorema 10.3). Algoritmul construit asigură însă corectarea unei singure erori. Să exemplificăm funcţionarea lui.*

Fie $\mathbf{r} = 1111\ 0101\ 1100\ 1011$ *cuvântul recepţionat. Calculul sindromului este* $s(X) = \sum_{i=1}^{16} \frac{r_i}{X - a_i} = \alpha^7 X^2 + \alpha^{13} X + \alpha^3$ *(s-a ţinut cont că se lucrează într-un corp de caracteristică* 2*). Deoarece nu este* 0*, înseamnă că au apărut erori. Din formulă, deoarece* $card(M) = 1$*, vom avea* $\omega(X) = 1$*, iar* $\sigma(X) = X + b$*, unde* b *trebuie determinat folosind relaţia* (1)*. Prin identificarea coeficienţilor se obţine* $\sigma(X) = X + \alpha^6$*; cum* $a_8 = \alpha^6$*, rezultă că a fost modificat al* 8*-lea bit din cuvântul recepţionat. Fiind în binar, corectarea se realizează imediat şi se obţine* $\mathbf{v} = 1111\ 0100\ 1100\ 1011$.

10.4 Sistemul de criptare McElliece

Criptografia (sau *teoria ascunderii mesajelor*) prezintă multe elemente comune cu teoria codurilor; de fapt, numeroase sisteme de criptare sunt numite şi *coduri*. Ambele se ocupă cu transmiterea de mesaje asupra cărora se aplică anumite transformări şi care la destinaţie se regăsesc prin transformări inverse. Scopurile diferă însă: dacă rolul codurilor corectoare de erori este de a corija modificări aleatoare care apar în transmiterea unui set de date printr-un canal, în criptografie ideea este de a proteja mesajul faţă de persoanele neautorizate care-l pot intercepta, modifica şi utiliza. Un sistem de criptare (cu cheie publică) este în general cunoscut de toată lumea: oricine poate cripta un mesaj. Pentru decriptare însă, singura persoană care poate acest lucru este destinatarul legal (desemnat de obicei prin numele *Bob*); acesta este unicul deţinător al cheii de decriptare. Oricine altcineva poate doar eventual să "ghicească" cheia: o metodă sistematică de a o afla conduce la algoritmi nepolinomiali (probleme NP - complete).

Să privim procesul descris în această carte din punctul de vedere al criptografiei. În linii mari, acesta funcţionează astfel: dacă $\mathbf{a}$ este un mesaj de informaţie de k biţi, expeditorul (numit de obicei *Alice*) îl codifică într-un cuvânt de n biţi $\mathbf{b} = \mathbf{a}G$, unde G este matricea generatoare a codului A.

Bob primeşte un mesaj $\mathbf{r} \in Z_2^n$ (eventual $\mathbf{r} = \mathbf{b}$) şi caută un cuvânt $\mathbf{b}' \in A$ cu $d(\mathbf{r}, \mathbf{b}')$ minimă posibil. Va decodifica $\mathbf{r}$ în $\mathbf{b}'$ după care va calcula un mesaj de informaţie $\mathbf{a}'$ astfel ca $\mathbf{b}' = \mathbf{a}'G$.

Dacă *Bob* caută cuvântul - cod cel mai apropiat comparând $\mathbf{r}$ pe rând cu fiecare element din A, cum sunt 2^k astfel de cuvinte, algoritmul va fi

exponenţial, deci nefuncţional.

Se poate folosi o metodă alternativă de decodificare, bazată pe sindrom. Se calculează întâi $\mathbf{s} = H\mathbf{r}^T$. Dacă $\mathbf{s} = \mathbf{0}$, decodificarea lui $\mathbf{r}$ este tot $\mathbf{r}$. Altfel, se încearcă toate cuvintele de pondere 1. Pentru fiecare astfel de cuvânt $\mathbf{e}$ se calculează $H\mathbf{e}^T$. Dacă s-a găsit un $\mathbf{e}$ cu $H\mathbf{e}^T = \mathbf{s}$, $\mathbf{r}$ se decodifică în $\mathbf{r} - \mathbf{e}$. În caz contrar se încearcă cuvintele de pondere $2, 3, \ldots, [(d-1)/2]$. Dacă nu s-a găsit nici un cuvânt $\mathbf{e}$ cu $H\mathbf{e}^T = \mathbf{s}$, se deduce că au apărut mai mult de $[(d-1)/2]$ erori în cursul transmisiei.

Metoda prezentată funcţionează pentru toate codurile liniare. Pentru anumite clase speciale de coduri există algoritmi polinomiali de decodificare şi corectare a erorilor; în cazul general însă, problema este NP - completă. Algoritmul de criptare *McElliece* se bazează pe această idee. Trapa sa secretă o constituie o clasă de coduri pentru care există algoritmi eficienţi de decodificare: codurile Goppa. În plus, există un număr mare de coduri Goppa neechivalente, având aceiaşi parametri n, k, d.

Codurile Goppa binare recomandate de algoritmul McElliece au parametrii $n = 2^m$, $d = 2t + 1$, $k = n - mt$. Pentru o implementare eficientă McEliece sugerează $m = 10$, $t = 50$, ceea ce corespunde unui $(1024, 524)$ - cod Goppa cu $d = 101$. Un text clar este o secvenţă de 524 biţi, iar un text criptat este o secvenţă de 1024 biţi. Cheia publică este o matrice binară de dimensiuni 524×1024.

Algoritmul de criptare *McEliece* este următorul:

Fie G matricea generatoare a unui (n, k) - cod Goppa A cu $n = 2^m$, $k = n - mt$, $d = 2t + 1$.
Se definesc: S o matrice inversabilă $k \times k$ peste Z_2 şi P o matrice de permutare $n \times n$ (matrice în care pe fiecare linie şi coloană există o valoare 1, iar restul elementelor sunt 0).
Fie $G' = SGP$ şi $\mathcal{K} = \{(G, S, P, G')\}$ o cheie de criptare.
G' este publică iar G, S, P sunt secrete (cunoscute numai de *Bob*).
Pentru $K = (G, S, P, G')$ se defineşte criptarea mesajului $\mathbf{a} \in Z_2^k$ prin:

$$e_K(\mathbf{a}, \mathbf{e}) = \mathbf{a}G' + \mathbf{e},$$

unde $\mathbf{e} \in Z_2^n$ este un cuvânt aleator de pondere t.
Bob decriptează un mesaj $\mathbf{b} \in Z_2^n$ astfel:

1. Calculează $\mathbf{b_1} = \mathbf{b}P^{-1}$;
2. Decodifică $\mathbf{b_1}$ obţinând $\mathbf{b_1} = \mathbf{a_1} + \mathbf{e_1}$ unde $\mathbf{a_1} \in \mathbf{A}$;
3. Calculează $\mathbf{a_0} \in Z_2^k$ astfel ca $\mathbf{a_0}G = \mathbf{a_1}$;
4. Calculează $\mathbf{a} = \mathbf{a_0}S^{-1}$.

Exemplul 10.9 *Vom exemplifica algoritmul pe un* $(8,2)$ *- cod Goppa cu* $d = 5$. *Acest cod - extrem de mic (are doar* 4 *cuvinte) este generat de matricea*

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

Să presupunem că Bob alege matricile

$$S = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \qquad cu \qquad S^{-1} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

şi

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad P^{-1} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Matricea publică generată este deci

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

Să presupunem că Alice vrea să cripteze textul clar $\mathbf{a} = (0,1)$ *folosind vectorul - eroare* $\mathbf{e} = (0,0,1,0,0,1,0,0)$ *(ales aleator) de pondere* 2. *Textul criptat este* $\mathbf{b} = \mathbf{a}G' + \mathbf{e} = (1,1,1,1,0,0,1,0)$.

După recepţionarea mesajului, Bob calculează întâi

$$\mathbf{b_1} = \mathbf{b}P^{-1} = (1,1,1,1,1,0,0,0),$$

pe care îl scrie sub forma $\mathbf{a_1} + \mathbf{e_1}$ *unde* $\mathbf{a_1} = (1,1,1,1,0,1,0,0)$ *este un cuvânt - cod, iar* $\mathbf{e_1} = (0,0,0,0,1,1,0,0) \neq \mathbf{e}$ *(din cauza înmulţirii cu* P^{-1}*).*

Bob calculează apoi mesajul $\mathbf{a_0} = (1,1)$, *singurul cu proprietatea* $\mathbf{a_0}G = \mathbf{a_1}$.

Ultimul pas este determinarea lui $\mathbf{a} = S^{-1}\mathbf{a_0} = (0,1)$, *care este textul clar expediat de Alice.*

10.5 Exerciţii

10.1 *Să se construiască matricea generatoare a* $(13,10)$ *- codului Hamming ternar din Exemplul 10.4.*

10.2 *În codul din Exemplul 10.4 să se codifice mesajele de informaţie:*
(*a*) 1122002211; (*b*) 0000111122; (*c*) 1020012210;
(*d*) 2200211101; (*e*) 2222222222; (*f*) 0000000000.

10.3 *Să se construiască toate codurile Hamming peste Z_3 şi Z_5.*

10.4 *Demonstraţi Lema 10.1*

10.5 *Pentru $g(X) = X^{2t}$ arătaţi că* $\dfrac{1}{X-a} = \sum_{j=1}^{2t} a^{-j} X^{j-1}$.

10.6 *Găsiţi distanţa minimă a codului Goppa binar determinat de $g(X) = 1 + X^2$ şi de elementele $\mathbf{0}, \alpha, \alpha^2 \in GF(2^2)$.*

10.7 *Polinomul generator este $g(X) = 1+X^2$ iar parametrii sunt cele 15 elemente nenule ale extensiei $GF(2^4)$, generată de rădăcina polinomului $1 + X + X^4$. Analizaţi codul Goppa binar astfel construit.*

10.8 *Găsiţi o matrice de control a (16,8) - codului Goppa binar ireductibil. dat de $g(X) = \alpha^3 + X + X^2$, unde $\alpha \in GF(2^4)$ este primitiv.*

10.9 *Pentru codul Goppa definit în Exemplul 10.6 să se determine matricea generatoare.*

10.10 *Folosind codul Goppa construit în Exemplul 10.6, să se decodifice cuvintele. folosind algoritmi de decodificare pentru codurile liniare:*

(*a*) 0110110010011110; (*b*) 0111011001000000;
(*c*) 1011110101100100; (*d*) 1111111111111111;
(*e*) 1011011111001011; (*f*) 1010010110000010.

10.11 *Pentru acelaşi cod Goppa, folosind algoritmul de decodificare rapidă Berlekamp, să se decodifice cuvintele*
(*a*) 1101000101111001; (*b*) 1100101101110100;
(*c*) 0011100011010110; (*d*) 0001100010010110.

10.12 *Verificaţi că polinomul $1 + X + X^2$ este ireductibil în $GF(2^5)$. Determinaţi parametrii şi matricea de control a codului Goppa ireductibil corespunzător.*

10.13 *Să se construiască un sistem de criptare McElliece pentru codul Goppa definit în Exemplul 10.5, cu matricile S şi P din Exemplul 10.9.*

10.14 *Să se definească un algoritm de inversare a matricilor de permutare de dimensiuni $2^m \times 2^m$, $m \leq 4$.*

Capitolul 11

Coduri Preparata

În 1968 Preparata a introdus o clasă de coduri neliniare corectoare de două erori, care s-au dovedit foarte interesante prin proprietăţile şi prin legăturile pe care le au cu diverse alte coduri deja studiate. Ele sunt o combinaţie între codurile Reed - Muller şi codurile BCH corectoare de două erori. Definiţia pe care o vom folosi aparţine lui Baker şi permite o abordare mai simplă a algoritmilor de codificare/decodificare.

11.1 Coduri Preparata extinse

Să revenim la modalitatea de marcare a poziţiilor cuvintelor de lungime 2^r folosind elementele lui $GF(2^r)$, aşa cum s-a definit în algoritmul de decodificare Peterson pentru codurile BCH (secţiunea 8.2, Capitolul 8). Deci:

Fie $r \geq 2$ un număr întreg, $\alpha \in GF(2^r)$ primitiv, şi $U \subseteq GF(2^r)$; se notează $\chi(U)$ cuvântul de lungime 2^r definit prin

1 pe poziţia i dacă $\alpha^i \in U$ $(0 \leq i \leq 2^r - 2)$,
1 pe poziţia $2^r - 1$ dacă $\mathbf{0} \in U$,
0 în rest.

Exemplul 11.1 *Fie $\alpha \in GF(2^3)$ primitiv. Atunci*

$$\begin{aligned} \chi(\{\mathbf{0}\}) &= 00000001 \\ \chi(\{\alpha^2, \alpha^5, \alpha^6\}) &= 00100110 \\ \chi(\emptyset) &= 00000000. \end{aligned}$$

Se definesc în mod natural operaţiile:

$U + \beta = \{u + \beta | u \in U\}$, $\beta U = \{\beta u | u \in U\}$,

$U\Delta V = \{u | u \in U \cup V, u \notin U \cap V\}$, $U, V \subseteq GF(2^r)$, $\beta \in GF(2^r)$.
Egalităţile de mai jos se verifică imediat:

$$\chi(U) + \chi(V) = \chi(U\Delta V),$$
$$w(\chi(U)) = card(U). \quad (1)$$

Exemplul 11.2 *Fie $GF(2^3)$ construit folosind rădăcina (primitivă) α a polinomului $1 + X + X^3$, şi mulţimile $U = \{\alpha^2, \alpha^5, \alpha^6\}$, $V = \{\alpha^5, \mathbf{0}\}$. Atunci*

$U + \alpha^2 = \{\alpha^2 + \alpha^2, \alpha^5 + \alpha^2, \alpha^6 + \alpha^2\} = \{\mathbf{0}, \alpha^3, \alpha^0\}$,
$\alpha^2 U = \{\alpha^2\alpha^2, \alpha^2\alpha^5, \alpha^2\alpha^6\} = \{\alpha^4, \alpha^0, \alpha\}$,
$\chi(U) + \chi(V) = 00100110 + 00000101 = 00100011 = \chi(\{\alpha^2, \alpha^6, \mathbf{0}\}) =$
$= \chi(U\Delta V)$.

Definiţia 11.1 *Fie $r > 2$ un număr impar. Codul extins Preparata $P(r)$ este mulţimea cuvintelor de forma $\chi(U)\chi(V)$ unde $U, V \subseteq GF(2^r)$ verifică condiţiile*

(1) *$card(U)$ şi $card(V)$ sunt numere pare;*

(2) $\sum_{u \in U} u = \sum_{v \in V} v$;

(3) $\sum_{u \in U} u^3 + \left(\sum_{u \in U} u\right)^3 = \sum_{v \in V} v^3$.

Pentru uşurinţă vom nota cuvintele - cod cu $[\chi(U), \chi(V)]$ (a se vedea şi codul Golay binar, secţiunea 3.2, Capitolul 3).

Exemplul 11.3 *În ipotezele din Exemplul 11.2, să considerăm $U = \{\alpha, \alpha^2, \alpha^5, \mathbf{0}\}$, $V = \{\alpha^0, \alpha, \alpha^2, \alpha^3, \alpha^6, \mathbf{0}\}$. Prima condiţie din definiţie este verificată. Pentru a doua condiţie avem:*

$$\sum_{u \in U} u = \alpha + \alpha^2 + \alpha^5 + \mathbf{0} = 010 + 001 + 111 + 000 = 100 = \alpha^0,$$
$$\sum_{v \in V} v = \alpha^0 + \alpha + \alpha^2 + \alpha^3 + \alpha^6 + \mathbf{0} = 100 + 010 + 001 + 110 + 101 + 000 =$$
$$= 100 = \alpha^0.$$

A treia condiţie este şi ea verificată deoarece

$$\sum_{u \in U} u^3 = \alpha^3 + \alpha^6 + \alpha + \mathbf{0} = 110 + 101 + 010 + 000 = 001 = \alpha^2,$$
$$\sum_{v \in V} v^3 = \alpha^0 + \alpha^3 + \alpha^6 + \alpha^2 + \alpha^4 + \mathbf{0} = 100 + 110 + 101 + 001 + 011 + 000 =$$
$$= 101 = \alpha^6 \text{ şi } \alpha^2 + (\alpha^0)^3 = \alpha^6.$$

Deci $[\chi(U), \chi(V)] = 01100101\ 11110011$ este cuvânt - cod în $P(3)$.

Observaţia 11.1

- *Deoarece* $|\chi(U)| = |\chi(V)| = 2^r$, $P(r)$ *este un cod de lungime* 2^{r+1}.
- *Faptul că* **0** *este sau nu un element al mulţimilor* U *sau* V *nu afectează cu nimic condiţiile* 2 *şi* 3 *din Definiţia 11.1. Acest element se introduce în cele două mulţimi numai pentru a asigura prima condiţie din definiţie. Rezultă că biţii de pe poziţia* $2^r - 1$ *din* $\chi(U)$ *şi* $\chi(V)$ *sunt biţi de paritate (ceea ce justifică termenul de cod Preparata "extins").*
- *Exponentul '3' din condiţia* 3 *(Definiţia 11.1) nu este esenţial. Dacă el se înlocuieşte cu* $s = 2^t + 1$ *astfel ca aplicaţiile* $f, g : GF(2^r) \longrightarrow GF(2^r)$, $f(x) = x^s$, $g(x) = x^{s-2}$ *să fie bijective, toate proprietăţile se păstrează.*

  Aceste coduri se numesc Coduri Preparata generalizate.

  În schimb, este foarte important faptul că toate operaţiile sunt efectuate într-un corp de caracteristică 2.

Lema 11.1 *Fie* $[\chi(U), \chi(V)], [\chi(A), \chi(B)] \in P(r)$ *şi* $\beta = \sum_{u \in U} u$.

Atunci $[\chi(U \Delta A + \beta), \chi(V \Delta B)] \in P(r)$.

Demonstraţie: Vom arăta că noul cuvânt construit verifică condiţiile din Definiţia 11.1.

1. Deoarece $card(U), card(V), card(A), card(B)$ sunt pare, se arată uşor că:
   $card(V \Delta B) = card(V) + card(B) - 2 \cdot card(V \cap B)$,
   $card(U \Delta A + \beta) = card(U \Delta A) = card(U) + card(A) - 2 \cdot card(U \cap A)$. (2)

2. Să observăm întâi că dacă $I, J \subseteq GF(2^r)$, avem $\sum_{x \in I \Delta J} x = \sum_{x \in I} x + \sum_{x \in J} x$ deoarece orice element $\alpha^i \in I \cap J$ apare de două ori în membrul drept şi niciodată în membrul stâng, iar $\alpha^i + \alpha^i = 0$. Deci:
   $\sum_{x \in U \Delta A + \beta} x = \sum_{y \in U \Delta A} (y + \beta) = \sum_{y \in U \Delta A} y + \beta \cdot card(U \Delta A) = \sum_{y \in U} y + \sum_{y \in A} y + \mathbf{0}$ (deoarece $card(U \Delta A)$ este par) $= \sum_{y \in V} y + \sum_{y \in B} y = \sum_{y \in V \Delta B} y$.

3. $$\sum_{x\in U\Delta A+\beta} x^3+\left(\sum_{x\in U\Delta A+\beta} x\right)^3=\sum_{y\in U\Delta A}(y+\beta)^3+\left(\sum_{y\in V\Delta B} y\right)^3=$$

$$=\sum_{y\in U}(y+\beta)^3+\sum_{y\in A}(y+\beta)^3+\left(\sum_{y\in V} y+\sum_{y\in B} y\right)^3=\sum_{y\in U} y^3+\beta\sum_{y\in U} y^2+$$

$$\beta^2\sum_{y\in U} y+\beta^3 card(U)+\sum_{y\in A} y^3+\beta\sum_{y\in A} y^2+\beta^2\sum_{y\in A} y+\beta^3 card(A)+$$

$$\left(\sum_{y\in V} y\right)^3+\left(\sum_{y\in V} y\right)^2\left(\sum_{y\in B} y\right)+\left(\sum_{y\in V} y\right)\left(\sum_{y\in B} y\right)^2+\left(\sum_{y\in B} y\right)^3.$$

Dar $\beta=\sum_{y\in U} y=\sum_{y\in V} y$, $\left(\sum_{y\in V} y\right)^2=\sum_{y\in V} y^2$, iar $card(U)$, $card(V)$ sunt pare. Deci expresia de sus se reduce la

$$\sum_{y\in V} y^3+\sum_{y\in B} y^3=\sum_{y\in V\Delta B} y^3. \qquad \square$$

Definiţia 11.2 *Un cod C de lungime n este invariant la distanţă dacă pentru orice $x,y\in C$, are loc relaţia*

$$\forall i\ (0\le i\le n),\quad card(\{\alpha|d(\alpha,x)=i\})=card(\{\alpha|d(\alpha,y)=i\}).$$

Exemple simple de coduri invariante la distanţă sunt codurile liniare.

Ca o consecinţă imediată, pentru un cod invariant la distanţă care conţine cuvântul nul, distanţa minimă este ponderea minimă a unui cuvânt - cod nenul.

Vom arăta în continuare că un cod Preparata are această proprietate.

Lema 11.2 *$P(r)$ este invariant la distanţă.*

Demonstraţie: Fie cuvintele $[\chi(U),\chi(V)],[\chi(A),\chi(B)]\in P(r)$, aflate la distanţa i unul de altul. Conform Lemei 11.1, $[\chi(U\Delta U+\beta),\chi(V\Delta V)]$ şi $[\chi(U\Delta A+\beta),\chi(V\Delta B)]$ sunt de asemenea cuvinte - cod; se verifică uşor că ele se află tot la distanţa i. Deoarece $U\Delta U=\emptyset$ rezultă $[\chi(U\Delta U+\beta),\chi(V\Delta V)]=\mathbf{0}$, deci $[\chi(U\Delta A)+\beta),\chi(V\Delta B)]$ are pondere i. $\square$

Lema 11.3 *Fie $[\chi(U),\chi(V)]\in P(r)$. Atunci $P(r)$ mai conţine următoarele cuvinte:*

(i) $[\chi(V),\chi(U)]$;

(ii) $[\chi(U+\beta),\chi(V+\beta)]$ *pentru orice* $\beta\in GF(2^r)$;

(iii) $[\chi(\beta U),\chi(\beta V)]$ *pentru orice* $\beta\in GF(2^r)\setminus\{\mathbf{0}\}$.

Demonstraţie: Se rezolvă similar demonstraţiei de la Lema 11.1. □

Exemplul 11.4 *Conform Exemplului 11.3,* $[\chi(U), \chi(V)] \in P(3)$, *unde*
$U = \{\alpha, \alpha^2, \alpha^5, \mathbf{0}\}$, $V = \{\alpha^0, \alpha, \alpha^2, \alpha^3, \alpha^6, \mathbf{0}\}$.
Folosind Lema 11.3 în care se ia $\beta = \alpha^3$, *vom mai găsi drept cuvinte - cod pe*

$$[\chi(V), \chi(U)] = 11110011\ \ 01100101,$$
$$[\chi(U+\beta), \chi(V+\beta)] = [\chi(\{\alpha^0, \alpha^5, \alpha^2, \alpha^3\}), \chi(\{\alpha, \alpha^0, \alpha^5, \mathbf{0}, \alpha^4, \alpha^3\})] =$$
$$= 10110100\ \ 11011101,$$
$$[\chi(\beta U), \chi(\beta V)] = [\chi(\{\alpha^4, \alpha^5, \alpha, \mathbf{0}\}), \chi(\{\alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^2, \mathbf{0}\})] =$$
$$= 01001101\ \ 00111111.$$

Putem folosi Lema 11.3 pentru a simplifica problema determinării distanţei minime a lui $P(r)$. Pentru aceasta însă, este nevoie de un rezultat suplimentar (care justifică condiţia ca r să fie impar).

Lema 11.4 *Fie* $\alpha \in GF(2^r)$ *primitiv. Atunci* α^3 *este primitiv dacă* r *este impar şi nu este primitiv dacă* r *este par.*

Demonstraţie: Se ştie că α^i este primitiv dacă şi numai dacă $(i, 2^r - 1) = 1$. Avem $2^r - 1 = (3-1)^r - 1 = \mathcal{M}3 + (-1)^r - 1$; deci pentru r impar $2^r - 1 = \mathcal{M}3 - 2 = \mathcal{M}3 + 1$, ceea ce înseamnă că α^3 este primitiv. Similar, pentru r par, $2^r - 1 = \mathcal{M}3$, deci α^3 nu este primitiv. □

Corolarul 11.1 *Dacă* $r \geq 2$ *este un număr impar, atunci pentru orice* $x \in GF(2^r) \setminus \{\mathbf{0}\}$ *există* y *unic (numit rădăcina cubică a lui* x*) astfel ca* $y^3 = x$.

Teorema 11.1 $P(r)$ *are distanţa minimă* $d = 6$.

Demonstraţie: Deoarece $P(r)$ este invariant la distanţă, el va conţine un cuvânt - cod $[\chi(U), \chi(V)]$ de pondere d. Deci

$$d = w(\chi(U)) + w(\chi(V)) = card(U) + card(V).$$

Rezultă că d este număr par; mai trebuie verificat că $d \neq 2$, $d \neq 4$ şi există un cuvânt - cod de pondere 6.

Să presupunem $d = 2$; atunci $card(U) = 2$, $card(V) = 0$ (Lema 11.3 asigură că totul se poate reduce la acest caz). Folosind tot Lema 11.3 (ii), putem presupune că $U = \{\mathbf{0}, \mathbf{x}\}$ cu $\mathbf{x} \neq \mathbf{0}$. Dar atunci $\sum_{u \in U} u = \mathbf{0} + \mathbf{x} = \mathbf{x}$, ceea ce contrazice definiţia codurilor Preparata, pentru că $V = \emptyset$.

Să presupunem $d = 4$. Cu Lema 11.3 (i), aceasta înseamnă $card(U) = 4, card(V) = 0$ sau $card(U) = card(V) = 2$. În primul caz avem

$U = \{\mathbf{0}, \mathbf{x}, \mathbf{y}, \mathbf{z}\}$ cu $\mathbf{x}, \mathbf{y}, \mathbf{z} \neq \mathbf{0}$ şi distincte. Ultima condiţie a definiţiei codurilor Preparata este:

$$\mathbf{0^3} + \mathbf{x^3} + \mathbf{y^3} + \mathbf{z^3} + (\mathbf{0} + \mathbf{x} + \mathbf{y} + \mathbf{z})^3 = \mathbf{0} \text{ sau}$$

$(\mathbf{x} + \mathbf{y})(\mathbf{x} + \mathbf{z})(\mathbf{y} + \mathbf{z}) = \mathbf{0}$ ceea ce este imposibil, cele trei numere fiind distincte şi nenule.

În a doua variantă, se poate considera $U = \{\mathbf{0}, \mathbf{x}\}$, $V = \{\mathbf{y}, \mathbf{z}\}$, toate cele patru elemente fiind distincte. Din definiţie rezultă

$$\mathbf{0^3} + \mathbf{x^3} + (\mathbf{0} + \mathbf{x})^3 = \mathbf{y^3} + \mathbf{z^3} \text{ sau } \mathbf{y^3} + \mathbf{z^3} = \mathbf{0}.$$

Folosind acum Corolarul 11.1, din $\mathbf{y^3} = \mathbf{z^3}$ rezultă $\mathbf{y} = \mathbf{z}$, contradicţie.

Să construim acum un cuvânt - cod de pondere 6. Fie $\mathbf{x}, \mathbf{y}, \mathbf{z} \in GF(2^r)$ elemente distincte nenule. Există atunci (Corolarul 11.1) un element $\mathbf{v} \in GF(2^r)$ unic cu $\mathbf{v^3} = \mathbf{x^3} + \mathbf{y^3} + \mathbf{z^3}$. Dacă $\mathbf{v} = \mathbf{x}$, atunci $\mathbf{v^3} = \mathbf{x^3}$ deci $\mathbf{y^3} = \mathbf{z^3}$, ceea ce duce la contradicţia $\mathbf{y} = \mathbf{z}$. Deci $\mathbf{v}$ este distinct de $\mathbf{x}, \mathbf{y}, \mathbf{z}$.

Definim $\mathbf{u} = \mathbf{v} + \mathbf{x} + \mathbf{y} + \mathbf{z}$. Se observă că $\mathbf{u} \neq \mathbf{0}$ (altfel

$$\mathbf{v^3} + (\mathbf{x} + \mathbf{y} + \mathbf{z})^3 = (\mathbf{x} + \mathbf{y})(\mathbf{x} + \mathbf{z})(\mathbf{y} + \mathbf{z}) \neq \mathbf{0},$$

deci - cu Corolarul 11.1 - $\mathbf{v} \neq \mathbf{x} + \mathbf{y} + \mathbf{z}$). Fie acum $U = \{\mathbf{0}, \mathbf{u}\}$, $V = \{\mathbf{v}, \mathbf{x}, \mathbf{y}, \mathbf{z}\}$. Din construcţie, cum toate elementele sunt distincte, cuvântul $[\chi(U), \chi(V)]$ are ponderea 6, şi se verifică uşor că este cuvânt - cod. □

Observaţia 11.2 *Conform Definiţiei* 10.5 *(Capitolul* 10*), codurile Preparata nu sunt coduri "asimptotic bune", deoarece* $\lim_{i\to\infty} \frac{d_i}{n_i} = \lim_{i\to\infty} \frac{6}{n_i} = 0$. *Cum rata de corecţie a erorilor scade cu lungimea codului, este recomandabilă folosirea codurilor Preparata de lungime mică.*

Exemplul 11.5 *Să considerăm* $GF(2^3)$ *construit în Exemplul 11.2. Fie elementele* $\mathbf{x} = \alpha$, $\mathbf{y} = \alpha^3$, $\mathbf{z} = \alpha^5$. *Definim*

$\mathbf{v^3} = \mathbf{x^3} + \mathbf{y^3} + \mathbf{z^3} = \alpha^3 + \alpha^9 + \alpha^{15} = 110 + 001 + 100 = \alpha^4 = (\alpha^6)^3$ *(deoarece* $\alpha^7 = \mathbf{1}$*), deci se poate lua* $\mathbf{v} = \alpha^6$. *Mai departe,* $\mathbf{u} = \mathbf{v} + \mathbf{x} + \mathbf{y} + \mathbf{z} = \alpha^6 + \alpha + \alpha^3 + \alpha^5 = \alpha^4$. *Acum,* $U = \{\mathbf{0}, \mathbf{u}\} = \{\mathbf{0}, \alpha^4\}$, $V = \{\mathbf{v}, \mathbf{x}, \mathbf{y}, \mathbf{z}\} = \{\alpha^6, \alpha, \alpha^3, \alpha^5\}$ *şi se obţine* $[\chi(U), \chi(V)] =$ $= 00001001\ \ 01010110$, *care este un cuvânt cod din* $P(3)$, *de pondere* 6.

Teorema 11.2 *Codul Preparata nu este cod liniar.*

Demonstraţie: Relaţia

$$[\chi(U), \chi(V)] + [\chi(A), \chi(B)] = [\chi(U \Delta A), \chi(V \Delta B)]$$

se verifică imediat.

Din demonstraţia Teoremei 11.1 am văzut cum se pot construi cuvintele - cod $[\chi(U), \chi(V)]$, $[\chi(A), \chi(B)] \in P(r)$ cu $U = \{\mathbf{0}, \mathbf{u_1}\}$, $V = \{\mathbf{x_1}, \mathbf{y_1}, \mathbf{z_1}, \mathbf{v_1}\}$, $A = \{\mathbf{0}, \mathbf{u_2}\}$, $B = \{\mathbf{x_2}, \mathbf{y_2}, \mathbf{z_2}, \mathbf{v_2}\}$. Conform Lemei 11.1, $\mathbf{c} = [\chi(U\Delta A + \mathbf{u_1}), \chi(V\Delta B)] \in P(r)$. Facem următoarele observaţii (uşor de verificat):

(1) $card(U\Delta A + \mathbf{u_1}) \leq 2$;

(2) $d(\mathbf{c}, [\chi(U\Delta A), \chi(V\Delta B)]) \leq 2 \cdot card(U\Delta A + \mathbf{u_1}) \leq 4$;

(3) $P(r)$ are distanţa minimă 6.

Din ele rezultă că $[\chi(U), \chi(V)] + [\chi(A), \chi(B)] \notin P(r)$.

Deci $P(r)$ nu este cod liniar. □

Ca o consecinţă a acestei teoreme, nu se poate da nici o dimensiune pentru $P(r)$, deci se pare că nu se poate determina numărul de cuvinte - cod. Vom reuşi totuşi acest lucru în mod indirect, ca o consecinţă a schemei de codificare.

11.2 Codificarea codurilor Preparata

Fie $\alpha \in GF(2^r)$ primitiv şi să considerăm codul BCH de polinom generator $g(X) = m_1(X)m_3(X)$ unde $m_i(X)$ este polinomul minimal al lui α^i $(i = 1, 3)$. Conform Teoremei 7.7 (Capitolul 7), $grad(m_1(X)) = grad(m_3(X)) = r$, deci $grad(g(X)) = 2r$. Codul BCH astfel definit poate corecta maxim 2 erori şi are matricea de control (transpusă):

$$H^T = \begin{pmatrix} \mathbf{1} & \mathbf{1} \\ \alpha^1 & \alpha^3 \\ \alpha^2 & \alpha^6 \\ & \\ \alpha^{2^r-2} & \alpha^{3(2^r-2)} \end{pmatrix}$$

Deoarece $g(X)$ generează un cod ciclic, rezultă că orice submatrice a lui H^T formată din $2r$ linii consecutive este nesingulară, deci inversabilă. Se defineşte matricea A ca submatrice a lui H^T formată din ultimele $2r$ linii, şi H' matricea rămasă din H^T prin ştergerea lui A.

Exemplul 11.6 *În extensia $GF(2^3)$ generată de rădăcina α a polinomului $1 + X + X^3$ vom avea*

$$A = \begin{pmatrix} \alpha & \alpha^3 \\ \alpha^2 & \alpha^6 \\ \alpha^3 & \alpha^2 \\ \alpha^4 & \alpha^5 \\ \alpha^5 & \alpha \\ \alpha^6 & \alpha^4 \end{pmatrix} = \begin{pmatrix} 010 & 110 \\ 001 & 101 \\ 110 & 001 \\ 011 & 111 \\ 111 & 010 \\ 101 & 011 \end{pmatrix} \quad cu \quad A^{-1} = \begin{pmatrix} 001 & 011 \\ 111 & 010 \\ 011 & 101 \\ 110 & 100 \\ 101 & 110 \\ 111 & 001 \end{pmatrix}.$$

Pentru $GF(2^5)$ generată de rădăcina α a polinomului $1+X^2+X^5$ avem

$$A = \begin{pmatrix} \alpha^{21} & \alpha^{63} \\ \alpha^{22} & \alpha^{66} \\ \vdots & \vdots \\ \alpha^{30} & \alpha^{90} \end{pmatrix} = \begin{pmatrix} 00011 & 01000 \\ 10101 & 00001 \\ 11110 & 00101 \\ 01111 & 10001 \\ 10011 & 00111 \\ 11101 & 11011 \\ 11010 & 01100 \\ 01101 & 10101 \\ 10010 & 10011 \\ 01001 & 01101 \end{pmatrix}, \; A^{-1} = \begin{pmatrix} 00111 & 00010 \\ 00011 & 10001 \\ 10011 & 00011 \\ 11011 & 01010 \\ 01101 & 10101 \\ 10101 & 11001 \\ 00110 & 11111 \\ 11001 & 01110 \\ 11000 & 00111 \\ 10001 & 10100 \end{pmatrix}$$

Fie $\mathbf{a} = [\mathbf{a_L}, \mathbf{a_R}]$ un cuvânt binar de lungime $2^{r+1} - 2r - 2$, unde $|\mathbf{a_L}| = 2^r - 1$, $|\mathbf{a_R}| = 2^r - 2r - 1$. În notaţie polinomială putem scrie

$$\mathbf{a_L}H^T = [a_L(\alpha), a_L(\alpha^3)], \quad \mathbf{a_R}H' = [a_R(\alpha), a_R(\alpha^3)].$$

Se mai defineşte

$$\mathbf{v_R} = [a_L(\alpha) + a_R(\alpha), a_L(\alpha^3) + (a_L(\alpha))^3 + a_R(\alpha^3)]A^{-1}. \quad (3)$$

Teorema 11.3 *Fie $r > 2$ un număr întreg impar. Pentru orice cuvânt binar $\mathbf{a}$, $|\mathbf{a}| = 2^{r+1} - 2r - 2$, dacă $\chi(U) = [\mathbf{a_L}, p_L]$, $\chi(V) = [\mathbf{a_R}, \mathbf{v_R}, p_R]$ unde p_L, p_R sunt biţii de control pentru $\mathbf{a_L}$ respectiv $[\mathbf{a_R}, \mathbf{v_R}]$, atunci $[\chi(U), \chi(V)] \in P(r)$.*

Demonstraţie: $[\mathbf{a_R}, \mathbf{v_R}]H^T = \mathbf{a_R}H' + \mathbf{v_R}A = [a_R(\alpha), a_R(\alpha^3)] + [a_L(\alpha) + a_R(\alpha), a_L(\alpha^3) + (a_L(\alpha))^3 + a_R(\alpha^3)] = [a_L(\alpha), a_L(\alpha^3) + (a_L(\alpha))^3]$.

Dar $[\mathbf{a_R}, \mathbf{v_R}] = \left[\sum_{v \in V} v, \sum_{v \in V} v^3\right]$, $a_L(\alpha) = \sum_{u \in U} u$, $a_L(\alpha^3) + (a_L(\alpha))^3 =$

$= \sum_{u \in U} u^3 + \left(\sum_{u \in U} u\right)^3$, ceea ce verifică condiţiile din Definiţia 11.1.

Deci $[\chi(U), \chi(V)]$ este cuvânt - cod în $P(r)$. □

Corolarul 11.2 *$P(r)$ are $2^{2^{r+1}-2r-2}$ cuvinte - cod.*

Demonstraţie: În Teorema 11.3 există $2^{2^{r+1}-2r-2}$ alegeri posibile pentru $\mathbf{a}$, toate conducând la cuvinte - cod distincte. □

Putem da acum un algoritm de codificare a mesajelor de informaţie în codurile Preparata.

Fie $\mathbf{a_L}, \mathbf{a_R}$ cuvinte de lungimi $2^r - 1$ respectiv $2^r - 2r - 1$.
Fie $\mathbf{v_R}$ definit de (3).
Se construiesc p_L, p_R conform Teoremei 11.3.
Atunci mesajul de informaţie $\mathbf{a} = [\mathbf{a_L}, \mathbf{a_R}]$ se codifică în
$[\mathbf{a_L}, p_L, \mathbf{a_R}, \mathbf{v_R}, p_R]$.

Exemplul 11.7 *Să considerăm* $r = 3$, $\mathbf{a_L} = 0110010$, $\mathbf{a_R} = 1$. *Deci*

$$a_L(\alpha) = \alpha + \alpha^2 + \alpha^5 = \alpha^0, \quad a_R(\alpha) = \alpha^0,$$
$$a_L(\alpha^3) = \alpha^3 + \alpha^6 + \alpha^{15} = \alpha^2, \quad a_R(\alpha^3) = \alpha^0.$$

Din (3) *avem* $\mathbf{v_R} = [\alpha^0 + \alpha^0, \alpha^2 + \alpha^0 + \alpha^0]A^{-1} = [000\ 001]A^{-1} =$ $= 111001$ *unde* A^{-1} *este matricea din Exemplul 11.6. Atunci vom codifica* $\mathbf{a} = [0110010\ 1]$ *în* $\mathbf{c} = [\mathbf{a_L}, p_L, \mathbf{a_R}, \mathbf{v_R}, p_R] = [01100010\ 1\ 1\ 111001\ 1]$.

Deci $\mathbf{c} = [\chi(U), \chi(V)]$ *unde* $\chi(U) = 01100101$, $\chi(V) = 11110011$, *care coincide cu secvenţa - cod construită în Exemplul 11.3.*

11.3 Decodificarea codurilor Preparata

Cum $P(r)$ are distanţa minimă 6, el poate corecta cel mult 2 erori. Fie $\mathbf{b}$ un cuvânt recepţionat; îl scriem $\mathbf{b} = [\mathbf{b_L}, p_L, \mathbf{b_R}, p_R]$ unde $\mathbf{b_L}$, $\mathbf{b_R}$ sunt ambele de lungime $2^r - 1$ iar p_L, p_R sunt biţi de control a parităţii. Se calculează

$$\mathbf{b_L}H^T = [b_L(\alpha), b_L(\alpha^3)], \quad \mathbf{b_R}H^T = [b_R(\alpha), b_R(\alpha^3)].$$

Apar mai multe cazuri, în funcţie de poziţiile unde apar erori.

1. Dacă erorile apar pe poziţiile de control, atunci
$$b_L(\alpha) = b_R(\alpha), \quad b_L(\alpha^3) + (b_L(\alpha))^3 = b_R(\alpha^3)$$
(conform Definiţiei 11.1), ceea ce se verifică uşor.

2. Dacă există o eroare pe poziţia i în $\mathbf{b_R}$ şi cel mult o eroare pe poziţiile de control atunci

$b_L(\alpha) = b_R(\alpha) + \alpha^i$, $\quad b_L(\alpha^3) + (b_L(\alpha))^3 = b_R(\alpha^3) + \alpha^{3i}$, deci
$$(b_L(\alpha) + b_R(\alpha))^3 = b_L(\alpha^3) + (b_L(\alpha))^3 + b_R(\alpha^3).$$

Dacă ultima relaţie este verificată, atunci se scrie $\alpha^i = b_L(\alpha) + b_R(\alpha)$ şi se schimbă bitul i din $\mathbf{b_R}$ (plus cel mult un bit de control).

3. Dacă există o eroare pe poziţia i din $\mathbf{b_L}$ şi cel mult o eroare pe poziţiile de control, similar cu cazul anterior (lucru posibil pe baza Lemei 11.3) se verifică relaţia
$$(b_R(\alpha) + b_L(\alpha))^3 = b_R(\alpha^3) + (b_R(\alpha))^3 + b_L(\alpha^3);$$
în caz afirmativ, se calculează $\alpha^i = b_R(\alpha) + b_L(\alpha)$ şi se schimbă bitul i din $\mathbf{b_L}$ (plus cel mult un bit de control).

4. Dacă apar două erori în $\mathbf{b_R}$ pe poziţiile i şi j, atunci:
$$b_L(\alpha) = b_R(\alpha) + \alpha^i + \alpha^j, \quad b_L(\alpha^3) + (b_L(\alpha))^3 = b_R(\alpha^3) + \alpha^{3i} + \alpha^{3j},$$
deci se determină $\alpha^i + \alpha^j$ şi $\alpha^{3i} + \alpha^{3j}$. Valorile i, j sunt determinate cu un algoritm similar celui de la codurile BCH.

5. Dacă apar două erori în $\mathbf{b_L}$, invocând din nou Lema 11.3 putem aplica analiza de la cazul anterior.

6. Dacă apar două erori: una pe poziţia i în $\mathbf{b_L}$ şi una pe poziţia j în $\mathbf{b_R}$, atunci
$$b_L(\alpha)+\alpha^i = b_R(\alpha)+\alpha^j,\ b_L(\alpha^3)+\alpha^{3i}+(b_L(\alpha)+\alpha^i)^3 = b_R(\alpha^3)+\alpha^{3j}.$$
Acest sistem de necunoscute α^i şi α^j se rezolvă astfel: din prima ecuaţie se scoate $\alpha^j = b_L(\alpha) + \alpha^i + b_R(\alpha)$ şi se înlocuieşte în a doua ecuaţie, ceea ce dă
$$b_L(\alpha^3) + \alpha^{3i} + (b_L(\alpha) + \alpha^i)^3 = b_R(\alpha^3) + (b_L(\alpha) + \alpha^i)^3 + (b_L(\alpha) + \alpha^i)^2 b_R(\alpha) + (b_L(\alpha) + \alpha^i) b_R(\alpha)^2 + b_R(\alpha)^3.$$
După simplificări se ajunge la
$$\alpha^{3i} + \alpha^{2i} b_R(\alpha) + \alpha^i (b_R(\alpha))^2 + (b_R(\alpha))^3 =$$
$$= b_L(\alpha^3) + b_R(\alpha^3) + b_L(\alpha)^2 b_R(\alpha) + b_L(\alpha) b_R(\alpha)^2, \text{ sau}$$
$$(\alpha^i + b_R(\alpha))^3 = (b_L(\alpha^3) + b_R(\alpha^3)) + (b_L(\alpha) + b_R(\alpha))^3 + b_L(\alpha)^3 + b_R(\alpha)^3.$$
Notând membrul drept al acestei expresii cu Δ, obţinem soluţia
$$\alpha^i = b_R(\alpha) + \Delta^{1/3}, \quad \alpha^j = b_L(\alpha) + \Delta^{1/3}.$$

Deci locaţiile erorilor se pot determina în toate situaţiile posibile. Biţii de paritate pentru fiecare jumătate de cuvânt dau posibilitatea de a decide ce caz se aplică lui $\mathbf{b}$. Putem da acum algoritmul de decodificare pentru codurile Preparata $P(r)$:

Fie $\mathbf{b} = [\mathbf{b_L}, p_L, \mathbf{b_R}, p_R]$ cuvântul recepţionat.

0. Se calculează $L_1 = b_L(\alpha)$, $L_3 = b_L(\alpha^3)$, $R_1 = b_R(\alpha)$, $R_3 = b_R(\alpha^3)$:

1. Dacă $L_1 + R_1 = 0$ şi $L_3 + L_1^3 + R_3 = 0$, atunci singurele erori posibile au apărut pe poziţiile de control.

2. Dacă $(L_1+R_1)^3+L_3+L_1^3+R_3 = 0$, calculăm $\alpha^i = L_1+R_1$. Se corectează poziţia i din $\mathbf{b_R}$ şi cel mult un bit de control al parităţii; se cere retransmisia dacă ambii biţi de paritate trebuie modificaţi.

3. Dacă $(L_1+R_1)^3+R_3+R_1^3+L_3 = 0$, calculăm $\alpha^i = L_1+R_1$. Se corectează poziţia i din $\mathbf{b_L}$ şi cel mult un bit de control al parităţii; se cere retransmisia dacă ambii biţi de paritate trebuie modificaţi.

4. Dacă ambele jumătăţi ale lui $\mathbf{b}$ sunt de pondere pară şi
$$X^2+(L_1+R_1)X+\frac{L_3+L_1^3+R_3+(L_1+R_1)^3}{L_1+R_1} = (X+\alpha^i)(X+\alpha^j),$$
se corectează poziţiile i şi j din $\mathbf{b_L}$.

5. Dacă ambele jumătăţi ale lui $\mathbf{b}$ sunt de pondere pară şi
$$X^2+(L_1+R_1)X+\frac{R_3+R_1^3+R_3+(L_1+R_1)^3}{L_1+R_1} = (X+\alpha^i)(X+\alpha^j),$$
se corectează poziţiile i şi j din $\mathbf{b_R}$.

6. Dacă ambele jumătăţi ale lui $\mathbf{b}$ sunt de pondere impară, se calculează
$$\alpha^i = R_1 + (L_1^3 + R_1^3 + (L_1 + R_1)^3 + L_3 + R_3)^{1/3},$$
$$\alpha^j = L_1 + (L_1^3 + R_1^3 + (L_1 + R_1)^3 + L_3 + R_3)^{1/3}.$$
Se corectează poziţia i din $\mathbf{b_L}$ şi poziţia j din $\mathbf{b_R}$.

7. Dacă nu a apărut nici una din situaţiile anterioare, se cere retransmisia cuvântului.

Exemplul 11.8 *Să decodificăm următoarele cuvinte primite, despre care se presupune că au fost codificate folosind* $P(3)$, *unde* $GF(2^3)$ *s-a generat cu rădăcina* α *a polinomului* $1 + X + X^3$*:*

I: $\mathbf{b} = 10010011\ \ 11100111.$

(0) $[L_1, L_3] = \mathbf{b_L}H^T = 111\ 110, \quad [R_1, R_3] = \mathbf{b_R}H^T = 101\ 110;$

(1) $L_1 + R_1 = 111 + 101 = \alpha \neq 0;$

(2) $(L_1 + R_1)^3 + L_3 + L_1^3 + R_3 = \alpha^3 + \alpha^3 + \alpha^{15} + \alpha^3 = \alpha^0 \neq 0;$

(3) $(L_1 + R_1)^3 + R_3 + R_1^3 + L_1 = \alpha^3 + \alpha^3 + \alpha^{18} + \alpha^3 = \alpha^6 \neq 0;$

(4) $X^2 + \alpha X + \dfrac{\alpha^3 + \alpha^{15} + \alpha^3 + \alpha^3}{\alpha} = X^2 + \alpha X + \alpha^6 =$
$= (X + \alpha^2)(X + \alpha^4).$

Deci $\mathbf{b}$ *se decodifică în* 10010011 11001111.

II: $\mathbf{b} = 10100100\ \ 10001001.$

(0) $[L_1, L_3] = \mathbf{b_L}H^T = [010, 011]$, $[R_1, R_3] = \mathbf{b_R}H^T = [111, 011]$;
(1) $L_1 + R_1 = 010 + 111 = \alpha^6 \neq 0$;
(2) $(L_1 + R_1)^3 + L_3 + L_1^3 + R_3 = \alpha^{18} + \alpha^4 + \alpha^3 + \alpha^4 = \alpha^6 \neq 0$;
(3) $(L_1 + R_1)^3 + R_3 + R_1^3 + L_3 = \alpha^{18} + \alpha^4 + \alpha^{15} + \alpha^4 = \alpha^2 \neq 0$.

Ambele jumătăţi ale lui $\mathbf{b}$ *au pondere impară, deci*

(6) $\alpha^i = \alpha^5 + (\alpha^3 + \alpha^{15} + \alpha^{18} + \alpha^4 + \alpha^4)^{1/3} = \alpha^5 + (\alpha^5)^{1/3} =$
$= \alpha^5 + (\alpha^{12})^{1/3} = \alpha^5 + \alpha^4 = \alpha^0$.

S-a obţinut $i = 0$. *Se poate determina apoi imediat* $\alpha^j = \alpha + \alpha^4 = \alpha^2$, *deci* $j = 2$.

Cuvântul $\mathbf{b}$ *se decodifică în* 00100100 10101001.

III: $\mathbf{b} = 10001000\ 11101001$.
(0) $[L_1, L_3] = \mathbf{b_L}H^T = [111, 011]$, $[R_1, R_3] = \mathbf{b_R}H^T = [100, 000]$;
(1) $L_1 + R_1 = 111 + 100 = \alpha^4 \neq 0$;
(2) $(L_1 + R_1)^3 + L_3 + L_1^3 + R_3 = \alpha^{12} + \alpha^4 + \alpha^{15} + \mathbf{0} = \alpha^3 \neq 0$;
(3) $(L_1 + R_1)^3 + R_3 + R_1^3 + L_3 = \alpha^{12} + \mathbf{0} + \alpha^0 + \alpha^4 = \mathbf{0}$.

Calculăm $\alpha^i = L_1 + R_1 = \alpha^4$, *deci* $i = 4$. *Dar modificarea bitului* 4 *în* $\mathbf{b_L}$ *cere ca ambii biţi de control să fie modificaţi; deci se cere retransmisia cuvântului.*

Pentru codul Preparata obţinut prin relaxarea lui $P(r)$, se poate aplica aceeaşi strategie de decodificare folosită la codul Golay; cum acesta are $d = 5$, poate corecta de asemenea cel mult 2 erori independente.

11.4 Coduri înrudite cu codul Preparata

11.4.1 Codul Nördstrom - Robinson

Să plecăm de la codul Golay binar extins C_{24} (a se vedea secţiunea 3.2.1, Capitolul 3), ale cărui cuvinte se pot scrie sub forma $\mathbf{a} = [\mathbf{a_1}, \mathbf{a_2}]$, unde $|\mathbf{a_1}| = 8$, $|\mathbf{a_2}| = 16$. Considerăm submulţimea $GNR \subset C_{24}$ ale cărei elemente verifică condiţiile:

1. Dacă $\mathbf{a} \in GNR$ atunci $w(\mathbf{a_1}) \in \{0, 2\}$;

2. Dacă $\mathbf{a}, \mathbf{b} \in GNR$ atunci $d(\mathbf{a_1}, \mathbf{b_1}) \leq 2$.

Evident, GNR va avea 256 cuvinte. Pe baza lui se construieşte codul

$$NR = \{\mathbf{a_2} | \mathbf{a} = [\mathbf{a_1}, \mathbf{a_2}] \in GNR\}.$$

Acesta este un cod neliniar de lungime $n = 16$ şi distanţă minimă $d = 6$. Numit *codul Nöstrom - Robinson*, el coincide cu codul Preparata $P(3)$.

Un alt cod derivat din acesta este codul *Nadler*, construit astfel: se păstrează din NR doar cuvintele - cod care coincid pe ultimele două poziţii. Apoi la toate aceste cuvinte se şterg ultimele trei caractere (o dublă scurtare urmată de o relaxare). Codul astfel obţinut este unic, are $n = 13$, $d = 5$ şi 64 cuvinte - cod.

11.4.2 Codul Kerdock

Definiţia 11.3 *Fie $r \geq 3$, r impar. Un cod Kerdock $K(r)$ este format din toate cuvintele de forma $[\chi(U), \chi(V)]$ care verifică condiţiile:*

1. *$U, V \subset GF(2^r)$; $card(U)$ şi $card(V)$ pare;*

2. $\forall s\ (1 \leq s \leq 2^r - 2)\ (1 \leq w_2(s) < r - 2) \Rightarrow \sum_{u \in U} u^s = \sum_{v \in V} v^s = \mathbf{0}$;

3. $\forall s\ (\leq s \leq 2^r - 2)\ (w_2(s) = r - 2) \Rightarrow \sum_{u \in U} u^s = \sum_{v \in V} v^s =$

$$= \left(\sum_{u \in U} u^{-1} + \sum_{v \in V} v^{-1} \right)^{-s}$$

S-a notat cu $w_2(s)$ ponderea reprezentării în binar a lui s.

Se consideră prin convenţie $\forall s$, $\mathbf{0^{-s}} = \mathbf{0}$.

Codul Kerdok $K(r)$ (definit în 1972) este de fapt un subcod al codului Reed - Muller $\mathcal{RM}(2, r + 1)$; el este format din perechi de translatări de cuvinte din $\mathcal{RM}(1, r)$ selectate în aşa fel ca să maximizeze distanţa minimă dintre două translatări.

Este interesant că dintre toate codurile cu aceeaşi distanţă minimă, codul Kerdok are numărul maxim de cuvinte - cod.

Nu vom detalia demonstraţii referitoare la proprietăţile acestor coduri. Pentru informaţii suplimentare se pot folosi [4] şi [9], cu menţiunea că justificările pleacă de la o modalitate diferită de definire a codurilor $K(r)$ şi $P(r)$. Rezultatele sunt însă deosebit de interesante. Astfel:

Teorema 11.4 *$K(r)$ este invariant la distanţă.*

Lema 11.5 $K(3) = P(3)$.

Teorema 11.5 *Codurile $P(r)$ şi $K(r)$ sunt duale.*

Ca o consecinţă imediată, $P(3)$ este un cod auto-dual.

11.5 Exerciţii

11.1 *Demonstraţi:*
(a) Egalităţile (1) şi (2); (b) Lema 11.3; (c) Corolarul 11.1.

11.2 *Aplicaţi Lema 11.3 cuvântului - cod definit în Exemplul 11.4, folosind (a) $\beta = \alpha^0$; (b) $\beta = \alpha$; (c) $\beta = \alpha^6$.*

11.3 *De ce $[\chi(\beta U), \chi(\beta V)]$ nu este cuvânt - cod pentru $\beta = \mathbf{0}$?*

11.4 *Arătaţi că cele trei cuvinte obţinute în Exemplul 11.4 satisfac condiţiile definiţiei codurilor Preparata.*

11.5 *Fie $GF(2^3)$ generat de rădăcina α a polinomului $1 + X + X^3$. Folosind următoarele cuvinte $\mathbf{x}, \mathbf{y}, \mathbf{z} \in GF(2^3)$ construiţi cuvinte - cod de pondere 6 din $P(3)$:*
(a) ($\mathbf{x} = \alpha$, $\mathbf{y} = \alpha^2$, $\mathbf{z} = \alpha^3$); (b) ($\mathbf{x} = \alpha$, $\mathbf{y} = \alpha^4$, $\mathbf{z} = \alpha^6$);
(c) ($\mathbf{x} = \alpha^0$, $\mathbf{y} = \alpha^3$, $\mathbf{z} = \alpha^6$).

11.6 *Fie $GF(2^3)$ construit cu $1 + X + X^3$, iar A^{-1} din Exemplul 11.6. Codificaţi următoarele mesaje folosind $P(3)$:*
(a) $\mathbf{a_L} = 1010100$, $\mathbf{a_R} = 1$; (b) $\mathbf{a_L} = 1010100$, $\mathbf{a_R} = 0$;
(c) $\mathbf{a_L} = 1111111$, $\mathbf{a_R} = 1$; (d) $\mathbf{a_L} = 1111111$, $\mathbf{a_R} = 0$;
(e) $\mathbf{a_L} = 0000000$, $\mathbf{a_R} = 1$.

11.7 *Fie $GF(2^5)$ construit cu $1 + X^2 + X^5$, iar A^{-1} din Exemplul 11.6. Codificaţi următoarele mesaje folosind $P(5)$:*
(a) $\mathbf{a_L} = 10100\ldots0$, $\mathbf{a_R} = 000001000100\ldots0$;
(b) $\mathbf{a_L} = 10100\ldots0$, $\mathbf{a_R} = 00\ldots0$;
(c) $\mathbf{a_L} = 10100\ldots0$, $\mathbf{a_R} = 111100\ldots0$;
(d) $\mathbf{a_L} = 0000\ldots0$, $\mathbf{a_R} = 100\ldots0$.
Care este lungimea lui $\mathbf{a_L}$? dar a lui $\mathbf{a_R}$?

11.8 *În aceleaşi condiţii din Exemplul 11.8, să se decodifice cuvintele:*

10000001	11101000	00011010	01000010	00100101	10100100
01010110	00011110	11101000	10001001	10011001	01010101
01000111	11001000	10101101	11010000	11101110	01010101

11.9 *Construiţi cele 64 cuvinte ale codului Nadler.*

Capitolul 12

Coduri convoluţionale

În cazul codurilor prezentate până acum, prin codificarea unui mesaj de informaţie de lungime k se obţine un cuvânt - cod de lungime n, ale cărui caractere nu depind decât de cele k elemente de informaţie curente. Avem de-a face cu un proces de codificare fără memorie, iar aceste coduri sunt numite şi *coduri - bloc*.

P. Elias a introdus în 1965 o clasă de coduri, remarcabile prin siguranţa sporită pe care o asigură în transmiterea informaţiei: *codurile convoluţionale*. În cazul lor, codificarea unui mesaj de lungime k se face ţinând cont şi de codificările mesajelor de informaţie anterioare.

De remarcat că NASA şi Agenţia Spaţială Europeană folosesc în mod curent o combinaţie între aceste coduri şi codurile Reed - Solomon. Fiecare mesaj de informaţie este codificat iniţial cu un cod RS, apoi cu un cod convoluţional.

12.1 Coduri liniare şi convoluţionale

După cum am văzut (Capitolul 2), un (n, k) - cod liniar codifică un mesaj de informaţie $\mathbf{u} \in Z_q^k$ într-un cuvânt - cod $\mathbf{v} = \mathbf{u}G$ de lungime n ($\mathbf{v} \in Z_q^n$). Un cod liniar poate fi privit însă şi ca o aplicaţie care transformă mesaje sursă de lungime $i \cdot k$ ($i = 1, 2, \ldots$) în mesaje - cod de lungime $i \cdot n$.

Exemplul 12.1 *Codul generat de matricea*

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

defineşte o funcţie astfel:

$$a_0a_1a_2a_3a_4a_5\ldots \longmapsto a_0a_1a_2x_0a_3a_4a_5x_1\ldots$$

unde

$$x_0 = a_0 + a_1 + a_2, \quad x_1 = a_3 + a_4 + a_5, \ldots$$

O astfel de funcţie poate fi exprimată prin polinoame de grad nedeterminat. Astfel, fie $a(X) = a_0 + a_1X + a_2X^2 + \ldots$ polinomul reprezentând mesajul sursă, şi $u(X) = u_0 + u_1X + u_2X^2 + \ldots$ mesajul codificat. Atunci un cod liniar este o aplicaţie C definită $C(a(X)) = u(X)$. O astfel de aplicaţie are următoarele proprietăţi:

- C este liniară:

$$C(a(X) + b(X)) = C(a(X)) + C(b(X));$$
$$C(ta(X)) = tC(a(X)).$$

- C este invariantă în timp: întârzierea mesajului sursă cu k tacţi are ca efect decalarea răspunsului cu n tacţi:

$$C(X^k a(X)) = X^n C(a(X)).$$

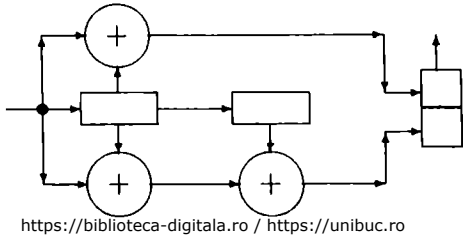
- C nu are memorie: codificarea unui mesaj sursă de lungime k nu depinde de mesajele sursă precedente.

Aceste trei proprietăţi caracterizează complet codurile liniare. Dacă se renunţă la ultima condiţie, se obţine o nouă clasă de coduri, numite *coduri convoluţionale.*

Definiţia 12.1 *Fie q un număr prim. Un (n, k) - cod convoluţional este o aplicaţie liniară $C : Z_q[X] \longrightarrow Z_q[X]$ cu proprietatea (de invarianţă de timp)*

$$C(X^k a(X)) = X^n C(a(X)), \quad \forall\, a(X) \in Z_q[X].$$

Exemplul 12.2 *Să considerăm circuitul liniar:*



El reprezintă un $(2, 1)$ *- cod convoluţional* C. *Acesta este liniar (datorită circuitului), invariant în timp (o întârziere de* 1 *tact la intrare provoacă o întârziere de* 2 *tacţi la ieşire).*

De exemplu, pentru intrarea 1 *se obţine* 11 *la primul tact,* 11 *la al doilea tact şi* 01 *la al treilea (după care urmează* 0000...*). Deci*

$$C(1) = 111101 = 1 + X + X^2 + X^3 + X^5.$$

Datorită invarianţei în timp, avem

$C(X) = C(01) = 00111101$, $C(X^2) = C(001) = 0000111101, \ldots$

(s-a ţinut cont de dualitatea de notare polinom - cuvânt).

Datorită liniarităţii, răspunsul $C(1)$ *caracterizează complet codul.*

De exemplu, mesajul de intrare 101 *se codifică prin*

$C(101) = C(1 + X^2) = C(1) + C(X^2) = C(1) + C(001) = 11110100\ldots + 0000111101\ldots = 1111101101.$

12.2 Coduri $(n, 1)$ - convoluţionale

În cazul particular $k = 1$, codul C este complet determinat de ieşirea pentru mesajul de intrare 1. Notăm cu

$$C(1) = g_0(X)$$

polinomul generator al acestei subclase de coduri. Din invarianţa în timp rezultă

$$C(X^i) = X^{ni} g_0(X), \quad \forall i \geq 1.$$

Deci, pentru orice polinom $a(X) = a_0 + a_1 X + \ldots + a_p X^p$, avem

$C(a(X)) = a_0 C(1) + a_1 C(X) + a_2 C(X^2) + \ldots + a_p C(X^p) = a_0 g_0(X) + a_1 X^n g_0(X) + a_2 X^{2n} g_0(X) + \ldots + a_p X^{np} g_0(X) = a(X^n) g_0(X)$.

Aceasta conduce la ideea că un $(n, 1)$ - cod convoluţional este generat de polinomul $g_0(X)$ conform regulii

$$C(a(X)) = a(X^n) g_0(X). \tag{1}$$

Invers, orice polinom $g_0(X) \in Z_q[X]$ defineşte un $(n, 1)$ - cod convoluţional.

Observaţia 12.1 *Ideea de polinom generator este similară celei de la codurile ciclice, unde codificarea mesajului de informaţie* $a(X)$ *revenea la înmulţirea cu polinomul generator. La codurile convoluţionale nu există însă restricţie referitoare la gradul lui* $a(X)$ *şi - în plus - la înmulţire se foloseşte* $a(X^n)$.

Există o modalitate simplă de construcţie a unui circuit liniar, generator al unui astfel de cod. Vom descrie aceasta pentru cazul binar.

Polinomul $g_0(X)$ se descompune în sume de n termeni consecutivi, fiecare sumă descriind modul de conectare al elementelor de înmagazinare. Astfel, fie m cel mai mic număr întreg cu proprietatea $mn \leq grad(g_0(X)) < (m+1)n$. Atunci

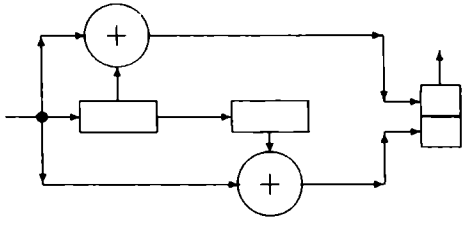
$g_0(X) = (b_0 + b_1X + \ldots + b_{n-1}X^{n-1}) + (b_nX^n + b_{n+1}X^{n+1} + \ldots + b_{2n-1}X^{2n-1}) + \ldots + (b_{mn}X^{mn} + b_{mn+1}X^{mn+1} + \ldots + b_{mn+n-1}X^{mn+n-1})$.(2)
(s-au completat eventual cu 0 coeficienţii lui $g_0(X)$, până la b_{mn+n-1}).

Numărul de $m+1$ paranteze indică faptul că sunt necesare m elemente de înmagazinare (prima paranteză se referă la intrare). A i - a secvenţă de termeni descrie ieşirea celui de-al $(i-1)$ - lea element de înmagazinare ($b_p = 1$ semnifică o legătură directă cu al p - lea element al ieşirii). Toate conexiunile care intră într-unul din cele n elemente ale bufferului de ieşire sunt în prealabil însumate.

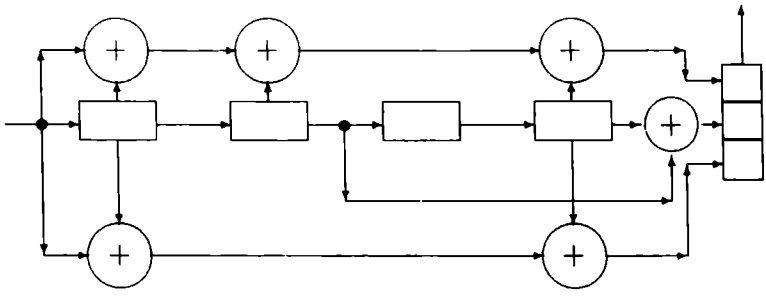
În plus, elementele de înmagazinare sunt legate între ele secvenţial.

Exemplul 12.3 *Să considerăm* $(2,1)$ *- codul convoluţional binar, de polinom generator* $g_0(X) = 1+X+X^2+X^5 = (1+X)+(X^2+0)+(0+X^5)$. *Circuitul liniar va avea două elemente de înmagazinare. Prima paranteză,* $1+X$, *arată că ambii biţi de ieşire sunt legaţi de intrare,* X^2+0 *indică faptul că primul element de înmagazinare este legat direct doar de primul bit de ieşire,* $0+X^5$ *arată că al doilea element de înmagazinare este legat numai de al doilea bit de ieşire.*

Codificatorul este descris de circuitul liniar:



Exemplul 12.4 *Fie* $(3,1)$ *- codul convoluţional binar (deci cu* 3 *biţi de ieşire) generat de polinomul* $g_0(X) = 1 + X^2 + X^3 + X^5 + X^6 + X^7 + X^{12} + X^{13} + X^{14}$. *Relaţia* $3m \leq 14 < 3(m+1)$ *dă* $m = 4$, *deci vor fi necesare patru elemente de înmagazinare. Pentru a construi circuitul liniar, rescriem acest polinom astfel:* $g_0(X) = (1+0+X^2) + (X^3 + 0 + X^5) + (X^6 + X^7 + 0) + (0+0+0) + (X^{12} + X^{13} + X^{14})$. *Se obţine codificatorul*



Similar codurilor ciclice, plecând de la polinomul generator, se poate construi matricea generatoare a $(n,1)$ - codurilor convoluţionale binare. Aceasta este

$$G = \begin{pmatrix} g_0(X) \\ X^n g_0(X) \\ X^{2n} g_0(X) \\ \vdots \end{pmatrix}$$

Numărul liniilor (şi deci şi al coloanelor) lui G nu este fixat, el depinzând de lungimea mesajului sursă. Astfel, pentru un mesaj de i caractere binare, matricea G are i linii şi $1 + grad(g_0(X)) + (i-1) \cdot n$ coloane.

Exemplul 12.5 *Fie* $(3,1)$ - *codul convoluţional binar de polinom generator* $g_0(X) = 1 + X + X^3 + X^4 + X^5 + X^8$. *Pentru un mesaj sursă de lungime* 3, *matricea* G *va avea dimensiunea* 3×15*:*

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

(spaţiile libere sunt completate cu 0*). Astfel, codificarea mesajului* 101 *este*

$$(1\ 0\ 1) \cdot G = (110\ 111\ 111\ 111\ 001).$$

Unele aplicaţii folosesc o descompunere a polinomului generator $g_0(X)$ în n polinoame, fiecare polinom corespunzând unei locaţii din bufferul de ieşire. Construcţia unei astfel de descompuneri este următoarea:

Formula (2) - care stă la baza construcţiei circuitelor liniare corespunzătoare unui $(n,1)$ - cod convoluţional – se poate rescrie

$$g_0(X) = \sum_{i=0}^{n-1} \left(b_i + b_{n+i} X^{n+i} + \ldots + b_{mn+i} X^{mn+i} \right).$$

Definim polinoamele *subgeneratoare*

$$g_0^i(X) = b_i + b_{n+i}X + \ldots + b_{mn+i}X^m.$$

Astfel, fiecare locaţie i a bufferului de ieşire depinde numai de polinomul subgenerator $g_0^i(X)$ şi de polinomul de intrare: pentru o intrare $a(X)$, prin a i-a locaţie a bufferului de ieşire se obţin coeficienţii polinomului $a(X)g_0^i(X)$.

Această observaţie conduce la o nouă reprezentare a operaţiei de codificare pentru un $(n,1)$ - cod convoluţional. Notând polinomul generator ca o secvenţă de polinoame subgeneratoare

$$g_0(X) = (g_0^0(X), g_0^1(X), \ldots, g_0^{n-1}),$$

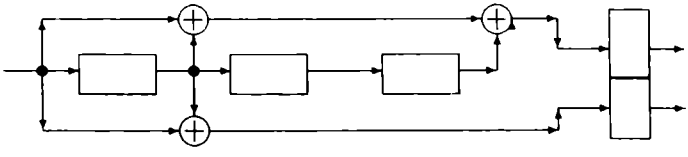
codificarea mesajului de informaţie $a(X)$ poate fi scris ca o secvenţă formată din n componente (infinite)

$$C(X) = (a(X)g_0(X), a(X)g_1(X), \ldots, a(X)g_{n-1}(X)),$$

ale cărui caractere se transmit în paralel în ordinea crescătoare a puterilor lui X.

Exemplul 12.6 *Fie* $(2,1)$- *codul convoluţional binar, având ca generator polinomul* $g(X) = 1 + X + X^2 + X^3 + X^6$.

Trecând prin formula (2), *el se poate scrie succesiv* $g_0(X) = (1 + X) + (X^2 + X^3) + (0 + 0) + (X^6 + 0) = (1 + X^2 + X^6) + (X + X^3)$; *de aici rezultă forma celor doi subgeneratori:* $g_0^0(X) = 1 + X + X^3$ *şi* $g_0^1(X) = 1 + X$. g_0^0 *"alimentează" prima celulă din bufferul de ieşire, iar* g_0^1 *- a doua celulă; se pot considera drept ieşiri paralele cele două săgeţi din dreapta circuitului.*



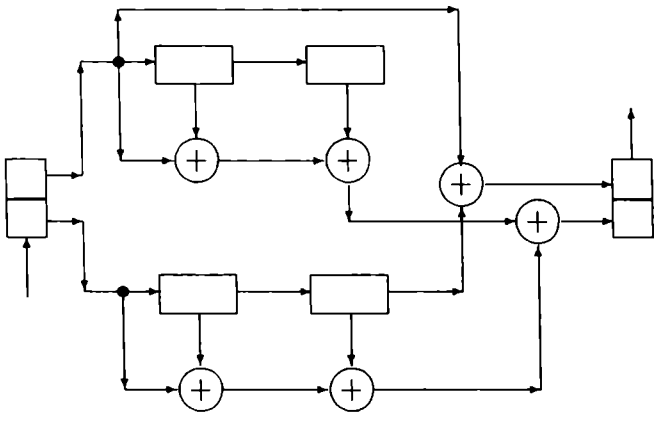
Mesajul $\mathbf{a} = 101$ *(sau - altfel scris -* $a(X) = 1 + X^2$*) este codificat în* $C(X) = ((1 + X^2)(1 + X + X^3), (1 + X^2)(1 + X)) = (1 + X + X^2 + X^5, 1 + X + X^2 + X^3)$.

Deci cuvântul - cod este $\mathbf{c} = 11\ 11\ 11\ 01\ 00\ 10\ 00\ \ldots.$

12.3 Coduri (n,k) - convoluţionale

Să dezvoltăm o descriere a codurilor (n,k) - convoluţionale, similară celei date anterior pentru cazul $k = 1$. Vom începe cu un exemplu:

Exemplul 12.7 *Fie circuitele liniare a două* $(2,1)$ *- coduri convoluţionale, care au ieşirea comună:*



La fiecare tact intrarea este formată din doi biţi care formează intrările în cele două circuite liniare. O întârziere cu doi tacţi la intrare generează o întârziere de doi tacţi la ieşire; deci, acesta este un $(2,2)$ *- cod convoluţional.*

Pentru intrarea 10 *ieşirea este* 110101 *(funcţionează doar circuitul superior). iar pentru* 01 *ieşirea este* 010111 *(numai pe baza circuitului inferior). Formal,*

$$C(1) = 1 + X + X^3 + X^5 = g_0(X),$$
$$C(X) = X + X^3 + X^4 + X^5 = g_1(X).$$

Folosind invarianţa în timp se pot determina şi alte ieşiri:

$$C(X^{2i}) = X^{2i}g_0(X), \quad C(X^{2i+1}) = X^{2i}g_1(X) \quad \forall i \geq 1.$$

Definiţia 12.2 *Pentru un* (n,k) *- cod convoluţional, polinoamele*

$$g_i(X) = C(X^i) \quad (0 \leq i \leq k-1)$$

se numesc "polinoame generatoare".

Teorema 12.1 *Un* (n,k) *- cod convoluţional de polinoame generatoare* $g_i(X)$, $(0 \leq i \leq k-1)$ *este determinat de funcţia de codificare* $C : Z_q[X] \longrightarrow Z_q[X]$ *definită*

$$C(a(X)) = \sum_{i=0}^{k-1} a^{(i)}(X^n)g_i(X), \tag{3}$$

unde pentru secvenţa $\mathbf{a} = a_0a_1a_2\ldots$ *s-a notat* $\mathbf{a}^{(i)} = a_ia_{i+k}a_{i+2k}\ldots$ $(0 \leq i \leq k-1)$, *iar* $a^{(i)}(X)$ *este reprezentarea sa polinomială.*

Reciproc, fiind date polinoamele $g_i(X)$ $i = 0, 1, \ldots, k-1$ *şi un număr* n, *formula* (3) *determină un* (n,k) *- cod convoluţional.*

Demonstraţie: Vom începe prin a arăta ultima afirmaţie. Pentru un i $(0 \leq i \leq k-1)$ fixat, conform relaţiei (1), avem

$$a^{(i)}(X) \mapsto a^{(i)}(X^n)g_i(X).$$

Aplicaţia este liniară (ca o compunere de transformări liniare) şi invariantă în timp (pentru că defineşte un $(n,1)$ - cod convoluţional). Deci, pentru orice secvenţă de intrare $\mathbf{a}$ putem scrie $\left(X^k a(X)\right)^{(i)} \mapsto X^n a^{(i)}(X)$.

Rezultă că ieşirea lui $X^k a(X)$ prin aplicaţia (3) este

$$X^n \sum_{i=0}^{k-1} a^{(i)}(X^n)g_i(X).$$

Să arătăm acum că un (n,k) - cod convoluţional definit de aplicaţia C din Definiţia 12.2 este identic cu cel dat de $C'(a(X)) = \sum_{i=0}^{k-1} a^{(i)}(X^n)g_i(X)$ unde $g_i(X) = C(X^i)$ $(0 \leq i \leq k-1)$ sunt polinoamele generatoare. Deoarece C şi C' sunt ambele liniare, este suficient să arătăm că $C(X^r) = C'(X^r)$ pentru orice $r \geq 0$; atunci vom avea

$$C(a_0 + a_1X + a_2X^2 + \ldots) = \sum_{r\geq 0} a_r C(X^r) = \sum_{r\geq 0} a_r C'(X^r) = C'(a_0 + a_1X + a_2X^2 + \ldots).$$

Deoarece ambele aplicaţii sunt invariante în timp, ne putem restrânge la $0 \leq r \leq k-1$. Pentru $\mathbf{a} = 00\ldots0100\ldots$ (cu 1 pe poziţia r), este evident că $\mathbf{a}^{(r)} = 100\ldots$ şi $\mathbf{a}^{(i)} = 000\ldots$ pentru $i \neq r$. Deci $C'(X^r) = g_r(X) = C(X^r)$. □

Analog codurilor $(n,1)$ - convoluţionale, şi în acest caz se poate defini matricea generatoare, care este o reprezentare matricială a relaţiei (3). Liniile matricii sunt $g_0(X), g_1(X), \ldots, g_{k-1}(X), X^n g_0(X), X^n g_1(X), \ldots$, considerate ca polinoame de grad neprecizat (teoretic infinit). Pentru mesajul $\mathbf{a} = a_0a_1\ldots a_r$ codificarea este

$$\mathbf{a}G = a_0g_0(X) + \ldots + a_{k-1}g_{k-1}(X) + X^n[a_kg_0(X) + \ldots + a_{2k-1}g_{k-1}(X)] + X^{2n}[a_{2k}g_0(X) + \ldots + a_{3k-1}g_{k-1}(X)] + \ldots = (a_0 + a_kX^n + a_{2k}X^{2n} + \ldots)g_0(X) + \ldots + (a_{k-1} + a_{2k-1}X^n + \ldots)g_{k-1}(X) = \sum_{i=0}^{k-1} a^{(i)}(X^n)g_i(X).$$

Exemplul 12.8 *Matricea generatoare a* $(2,2)$ *- codului convoluţional descris în Exemplul 12.7 este*

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

(locurile goale sunt completate cu 0*).*

Pentru intrarea 101 *ieşirea este*

$$(1\,0\,1)\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} = (1\,1\,1\,0\,0\,0\,0\,1),\ \textit{deci}$$

$$C(1 + X^2) = 1 + X + X^2 + X^7.$$

S-a prezentat anterior modul de construcţie al unui circuit liniar pentru $(n, 1)$ - coduri convoluţionale, plecând de la polinomul generator $g_0(X)$. În mod similar se determină circuitul liniar de codificare pentru un (n, k) - cod convoluţional:

1. Pentru $i = 0, 1, \dots, k-1$ se construieşte circuitul liniar al $(n, 1)$ - codului definit de $a(X) \mapsto a(X^n)g_i(X)$;

2. Se foloseşte un buffer comun de ieşire (de n simboluri) în care ajung rezultatele însumate ale celor k circuite liniare;

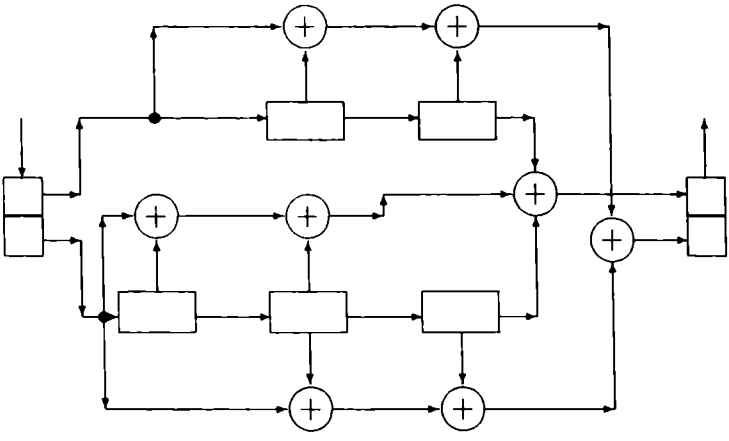
3. Secvenţa de intrare este segmentată în blocuri de câte k simboluri, fiecare al p-lea simbol din bloc constituind intrarea în al p-lea circuit liniar $(p = 1, 2, \dots, k)$.

Exemplul 12.9 *Să construim circuitul liniar pentru* $(2, 2)$ *- codul definit de polinoamele generatoare*

$$g_0(X) = X + X^3 + X^4 + X^5, \quad g_1(X) = 1 + X + X^2 + X^4 + X^5 + X^6 + X^7.$$

Scriem $g_0(X) = (0 + X) + (0 + X^3) + (X^4 + X^5)$ *deci circuitul său liniar are* 2 *elemente de înmagazinare. Pentru* $g_1(X) = (1 + X) + (X^2 + 0) + (X^4 + X^5) + (X^6 + X^7)$ *sunt necesare trei elemente de înmagazinare.*

Reprezentarea sa grafică este:



Definiţia 12.3 *Un (n,k) - cod convoluţional are "memorie" N dacă*

$$\forall i\ (0 \leq i \leq k-1), \qquad grad(g_i(X)) \geq Nn,$$

iar N este maxim cu această proprietate. Numărul Nn se numeşte "lungimea restrânsă" a codului.

După cum s-a văzut în secţiunea anterioară, un $(n,1)$ - cod convoluţional de memorie N poate fi implementat cu o combinaţie de N elemente de înmagazinare. N este numărul de simboluri de ieşire care pot fi modificate la schimbarea unui singur caracter de intrare.

12.4 Coduri convoluţionale sistematice

Un (n,k) - cod convoluţional sistematic are proprietatea că la fiecare ieşire a unui bloc de n caractere, pe primele k poziţii se găsesc simbolurile de infor-maţie. Vom da o construcţie directă a acestor coduri, plecând tot de similitudinea lor cu codurile - bloc.

Fie n, k, N $(k < n)$ numere naturale nenule. Considerăm sistemul de $k(n-k)$ secvenţe cu N componente peste Z_q:

$$g(i,j) = g_0(i,j)g_1(i,j)\dots g_{N-1}(i,j), \quad 1 \leq i \leq k, \quad 1 \leq j \leq n-k.$$

Aceste secvenţe se numesc *subgeneratori.*

Considerăm matricea

$$Q_0 = \begin{pmatrix} g_\infty(1) \\ g_\infty(2) \\ \vdots \\ g_\infty(k) \end{pmatrix} = (I_k P_0 O_k P_1 O_k P_2 O_k \dots P_{N-2} O_k P_{N-1} O_k O_k \dots),$$

unde I_k este matricea unitate, O_k este matricea nulă (ambele de ordin k),

$$P_t = \begin{pmatrix} g_t(1,1) & g_t(1,2) & & g_t(1,n-k) \\ g_t(2,1) & g_t(2,2) & & g_t(2,n-k) \\ & & & \\ g_t(k,1) & g_t(k,2) & & g_t(k,n-k) \end{pmatrix}, \qquad 0 \le t \le N-1$$

iar

$$g_\infty(i) = 0\ldots010\ldots g_0(i,1)\ldots g_0(i,n-k)0\ldots0g_1(i,1)\ldots g_1(i,n-k)$$
$$0\ldots0g_{N-1}(i,1)\ldots g_{N-1}(i,n-k)0\ldots \qquad (1 \le i \le k).$$

Cuvintele obţinute din primele Nn componente ale lui $g_\infty(i)$ $(1 \le i \le k)$:

$$g(i) = 0\ldots010\ldots0g_0(i,1)\ldots g_0(i,n-k)0\ldots0g_{N-1}(i,1)\ldots g_{N-1}(i,n-k)$$

se numesc *generatori.*

Fie p un număr natural. Se defineşte operatorul

$$D^p g_\infty(i) = \underbrace{0\ldots0}_{pn} g_\infty(i)$$

care translatează $g_\infty(i)$ cu $p \cdot n$ poziţii spre dreapta.

În sfârşit, fie

$$G_\infty = \begin{pmatrix} Q_0 \\ DQ_0 \\ D^2Q_0 \\ \vdots \end{pmatrix} = \begin{pmatrix} g_\infty(1) \\ \vdots \\ g_\infty(k) \\ Dg_\infty(1) \\ \vdots \\ Dg_\infty(k) \\ D^2g_\infty(1) \\ \vdots \end{pmatrix} =$$

$$\begin{pmatrix} I_kP_0 & O_kP_1 & O_kP_2 & & O_kP_{N-1} & & \\ & I_kP_0 & O_kP_1 & & O_kP_{N-2} & O_kP_{N-1} & \ldots \\ & & I_kP_0 & & O_kP_{N-3} & O_kP_{N-2} & O_kP_{N-1} \\ & & & & \vdots & & \end{pmatrix}$$

matricea generatoare a unui (n,k) - cod convoluţional.

Observaţia 12.2 *Evident, un (n,k) - cod convoluţional sistematic este un (n,k) - cod convoluţional. Invers, un (n,k) - cod convoluţional se poate transforma într-un cod sistematic adunând la fiecare polinom generator* $g_i(X) = \sum_{p=0}^{mn+n-1} b_{i,p}X^p$ *polinomul*

$$g_i'(X) = \sum_{p=0}^{m} \sum_{j=0}^{k-1} (k - b_{i,np+j}) X^{np+j} + b_i X^i \ (0 \leq i \leq k-1).$$

Deşi are - teoretic - un număr infinit de linii şi coloane, fiind în formă canonică, se poate construi imediat matricea de control a codului:

$$H_\infty = \begin{pmatrix} -P_0^T & I_{n-k} & & & & & \dots \\ -P_1^T & O_{n-k} & -P_0^T & I_{n-k} & & & \dots \\ -P_2^T & O_{n-k} & -P_1^T & O_{n-k} & -P_0^T & I_{n-k} & \dots \\ & & & \vdots & & & \\ -P_{N-1}^T & O_{n-k} & -P_{N-2}^T & O_{n-k} & -P_{N-3}^T & O_{n-k} & \\ & & -P_{N-1}^T & O_{n-k} & -P_{N-2}^T & O_{n-k} & \dots \\ & & & & -P_{N-1}^T & O_{n-k} & \dots \end{pmatrix}$$

a cărei transpusă este

$$H_\infty^T = \begin{pmatrix} -P_0 & -P_1 & -P_2 & & -P_{N-1} & & \dots \\ I_{n-k} & O_{n-k} & O_{n-k} & & O_{n-k} & & \\ & -P_0 & -P_1 & & -P_{N-2} & -P_{N-1} & \\ & I_{n-k} & O_{n-k} & & O_{n-k} & O_{n-k} & \end{pmatrix}$$

Se verifică imediat egalitatea $G_\infty H_\infty^T = 0_\infty$.

Să considerăm un mesaj de informaţie $\mathbf{a}$, sub forma unei succesiuni (teoretic) infinite. El va fi "descompus" iniţial în blocuri de lungime k:

$$\mathbf{a} = \underbrace{a_0(1) \dots a_0(k)}_{\mathbf{a_0}} \underbrace{a_1(1) \dots a_1(k)}_{\mathbf{a_1}} \dots \underbrace{a_p(1) \dots a_p(k)}_{\mathbf{a_p}} \dots$$

Mesajul codificat se obţine similar codurilor liniare, prin înmulţirea lui $\mathbf{a}$ cu matricea generatoare: $\mathbf{c} = \mathbf{a}G_\infty$; textul rezultat va fi descompus în blocuri de lungime n:

$$\mathbf{c} = \mathbf{a}G_\infty = \underbrace{c_0(1) \dots c_0(n)}_{\mathbf{c_0}} \underbrace{c_1(1) \dots c_1(n)}_{\mathbf{c_1}} \dots \underbrace{c_p(1) \dots c_p(n)}_{\mathbf{c_p}} \dots$$

Ţinând cont de forma canonică a matricii G_∞, rezultă următoarele relaţii:

$$c_p(i) = a_p(i), \quad (1 \le i \le k)$$
$$c_p(k+j) = \sum_{i=1}^{k} \sum_{t=0}^{N-1} a_{p-t}(i) g_t(i,j), \quad (1 \le j \le n-k). \tag{4}$$

Aceeaşi formulă de codificare se poate scrie şi în alt mod:

$$\mathbf{c} = \mathbf{a}G_\infty = \sum_{i=1}^{k} a_0(i) g_\infty(i) + \sum_{i=1}^{k} a_1(i) D g_\infty(i) + \sum_{i=1}^{k} a_2(i) D^2 g_\infty(i) + \ldots,$$

deci

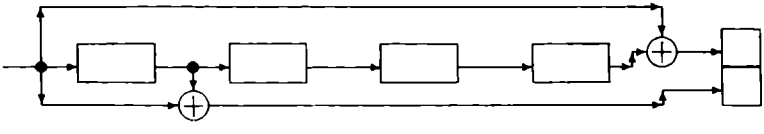
$$\mathbf{c} = \sum_{i=1}^{k} \sum_{t=0}^{\infty} a_t(i) D^t g_\infty(i). \tag{5}$$

Exemplul 12.10 *Să considerăm un* $(2,1)$ *- cod convoluţional binar cu* $N = 4$ *şi subgenerator* $g(1,1) = 1101$. *Generatorul va fi atunci* $g(1) = 110100001$ *iar matricea*

$$G_\infty = \begin{pmatrix} 11 & 01 & 00 & 01 & & & \ldots \\ & 11 & 01 & 00 & 01 & & \\ & & 11 & 01 & 00 & 01 & \\ & & & \vdots & & & \end{pmatrix}.$$

Dacă dorim să se transmită secvenţa de informaţie $\mathbf{a} = 10011\ldots$, *ea va fi codificată în* $\mathbf{c} = \mathbf{a}G_\infty = 1101001010\ldots$

Pentru a construi circuitul liniar de codificare, să observăm că generatorul se poate scrie polinomial sub forma $g_0(X) = 1 + X + X^3 + X^8$. *Sunt* 4 *elemente de înmagazinare şi vom avea*

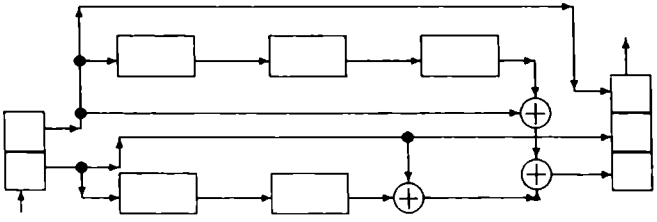


Exemplul 12.11 *Fie* $(3,2)$ *- codul convoluţional binar cu* $N = 3$ *şi subgeneratori* $g(1,1) = 101$, $g(2,1) = 110$. *Calculând generatorii, se obţine* $g(1) = 101000001$, $g(2) = 011001000$ *şi deci*

$$G_\infty = \begin{pmatrix} 101 & 000 & 001 & & & \ldots \\ 011 & 001 & 000 & & & \ldots \\ & 101 & 000 & 001 & & \ldots \\ & 011 & 001 & 000 & & \ldots \\ & & 101 & 000 & 001 & \ldots \\ & & 011 & 001 & 000 & \ldots \\ & & & \vdots & & \end{pmatrix}$$

Fie $\mathbf{a} = 11\ 00\ 10\ldots$ *o secvenţă de informaţie. Rezultatul codificării ei este* $\mathbf{c} = \mathbf{a}G_\infty = 110\ 001\ 100\ldots$. *Se observă că în fiecare grup de* 3 *caractere binare, pe primele două poziţii se află simboluri de informaţie.*

Pentru construcţia codificatorului, scriem cei doi generatori sub forma $g_0(X) = 1 + X^2 + X^8$, $g_1(X) = X + X^2 + X^5$. *Atunci*



12.5 Coduri convoluţionale arborescente

Cuvintele unui cod convoluţional sistematic se pot reprezenta folosind un graf arborescent infinit, cu noduri situate la o distanţă de n poziţii şi cu q^k descendenţi din fiecare nod. Codificarea unui mesaj revine la parcurgerea unui drum în acest arbore.

Pentru simplificare, să considerăm un $(n, 1)$ - cod convoluţional sistematic binar (generalizarea este imediată) de lungime restrânsă N. El este complet caracterizat prin $n-1$ subgeneratori

$$g(1,j) = g_0(1,j)g_1(1,j)\ldots g_{N-1}(1,j), \quad (1 \le j \le n-1).$$

Codul are un singur generator, anume

$$g(1) = 1g_0(1,1)g_0(1,2)\ldots g_0(1,n-1)0g_1(1,1)g_1(1,2)\ldots g_1(1,n-1)$$
$$\ldots 0g_{N-1}(1,1)\ldots g_{N-1}(1,n-1).$$

Vom folosi notaţiile

$$\mathbf{g_0} = 1g_0(1,1)\ldots g_0(1,n-1),$$
$$\mathbf{g_p} = 0g_p(1,1)\ldots g_p(1,n-1), \quad (1 \le p \le N-1).$$

Atunci $g(1) = \mathbf{g_0 g_1}\ldots\mathbf{g_{N-1}}$, iar $\mathbf{g_p}$ este a p - a ramificaţie a lui $g(1)$. Matricea generatoare a codului va fi - cu aceste notaţii:

$$G_\infty = \begin{pmatrix} g_\infty(1) \\ Dg_\infty(1) \\ D^2g_\infty(1) \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{g_0} & \mathbf{g_1} & \mathbf{g_{N-1}} & \mathbf{0} & \mathbf{0} & \ldots \\ \mathbf{0} & \mathbf{g_0} & \mathbf{g_{N-2}} & \mathbf{g_{N-1}} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \mathbf{g_{N-3}} & \mathbf{g_{N-2}} & \mathbf{g_{N-1}} & \\ & & \vdots & & & \end{pmatrix}$$

Fie $\mathbf{a} = a_0a_1a_2\ldots$ secvenţa de informaţie şi $\mathbf{c} = \mathbf{c_0c_1c_2}\ldots$ cuvântul - cod corespunzător, unde $\mathbf{c_p} = c_p(1)c_p(2)\ldots c_p(n)$. Are loc egalitatea

$$\mathbf{c} = \mathbf{c_0c_1c_2}\ldots = \mathbf{a}G_\infty = a_0g_\infty(1) + a_1Dg_\infty(1) + a_2D^2g_\infty(1) + \ldots \quad (6)$$

De aici rezultă că $\mathbf{c_0} = \mathbf{0}$ dacă $a_0 = 0$ şi $\mathbf{c_0} = \mathbf{g_0}$ dacă $a_0 = 1$. Cu alte cuvinte, codul poate fi partiţionat în două submulţimi de mărimi egale: o submulţime S_0 conţine toate secvenţele care corespund lui $a_0 = 0$ (deci toate cuvintele - cod din S_0 au acelaşi prefix $\mathbf{0}$); cealaltă submulţime S_1 conţine toate cuvintele - cod corespunzătoare lui $a_0 = 1$ (deci toate cuvintele - cod din S_1 au prefixul $\mathbf{g_0}$).

La rândul lui, S_0 se împarte în două submulţimi egale S_{00} şi S_{01}. S_{00} corespunde secvenţelor - cod pentru care $a_0 = a_1 = 0$ (deci toate cuvintele - cod din S_{00} încep cu $\mathbf{00}$). S_{01} conţine secvenţele cu $a_0 = 0$, $a_1 = 1$, deci cu prefixul $\mathbf{0g_0}$. Similar, S_1 se partiţionează în S_{10} şi S_{11}, S_{10} fiind pentru $a_0 = 1$, $a_1 = 0$ (deci cuvintele - cod din S_{10} au prefixul $\mathbf{g_0g_1}$); secvenţele din S_{11} încep cu prefixul $\mathbf{g_0(g_0 + g_1)}$ ($\mathbf{g_0 + g_1}$ este suma modulo 2 bit-cu-bit a secvenţelor $\mathbf{g_0}$ şi $\mathbf{g_1}$, operaţie cunoscută şi sub numele de XOR).

Acest procedeu continuă indefinit.

Deci cuvintele - cod pot fi aranjate într-un arbore cu noduri de n caractere şi câte două ramificaţii din fiecare nod. O ramificaţie este marcată de un cod - bloc de n caractere, corespunzătoare unui caracter de informaţie particular, iar întreaga secvenţă cod corespunde unui drum prin arbore (vezi Figura 12.1).

Operaţia de codificare poate fi privită atunci ca un proces în care este trasat un drum particular în arbore, conform instrucţiunilor date de mesajul de informaţie.

Exemplul 12.12 *Să considerăm* $(3,1)$ *- codul convoluţional binar cu* $N = 4$, *generat de* $g(1,1) = 1011$, $g(1,2) = 1101$. *Generatorul codului este* $g(1) = 111\ 001\ 010\ 011$, *iar arborele de codificare este construit în Figura 12.2*

Dacă vom considera de exemplu mesajul de informaţie $\mathbf{a} = 1001\ldots$, *secvenţa - cod generată va fi* $\mathbf{c} = 111\ 001\ 010\ 100\ \ldots$.

Această construcţie se poate extinde la un (n,k) - cod convoluţional sistematic binar. Deoarece la momentul 0 există 2^k secvenţe posibile de lungime k, toate cuvintele - cod se pot partiţiona în 2^k submulţimi $S_0, S_1, \ldots, S_{2^k-1}$. Toate secvenţele din S_i încep cu acelaşi bloc, corespunzător aceluiaşi mesaj de informaţie. Fiecare S_i se împarte la rândul lui în 2^k submulţimi, după tipul celui de-al doilea bloc de la tactul 1. Procesul continuă în mod similar, până la epuizarea mesajului de informaţie.

Figura 12.1:

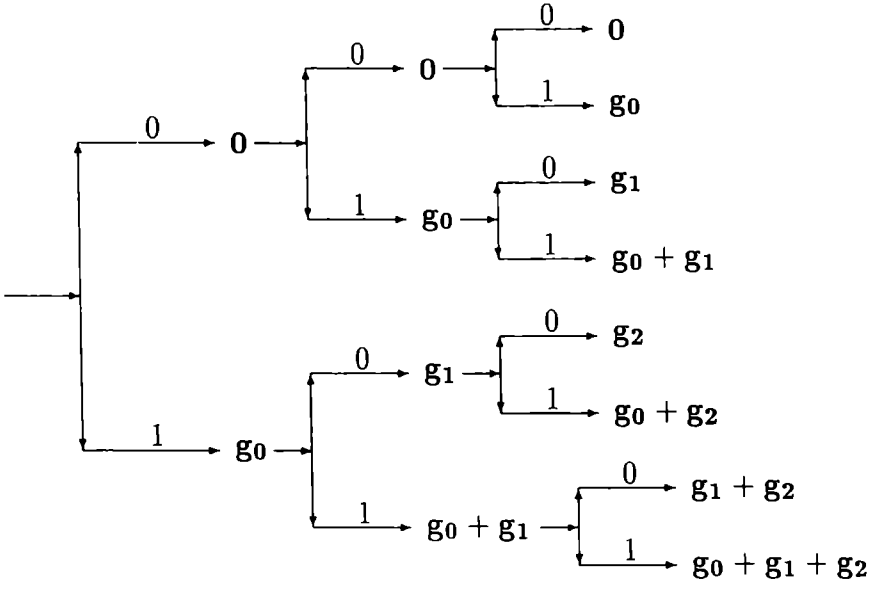
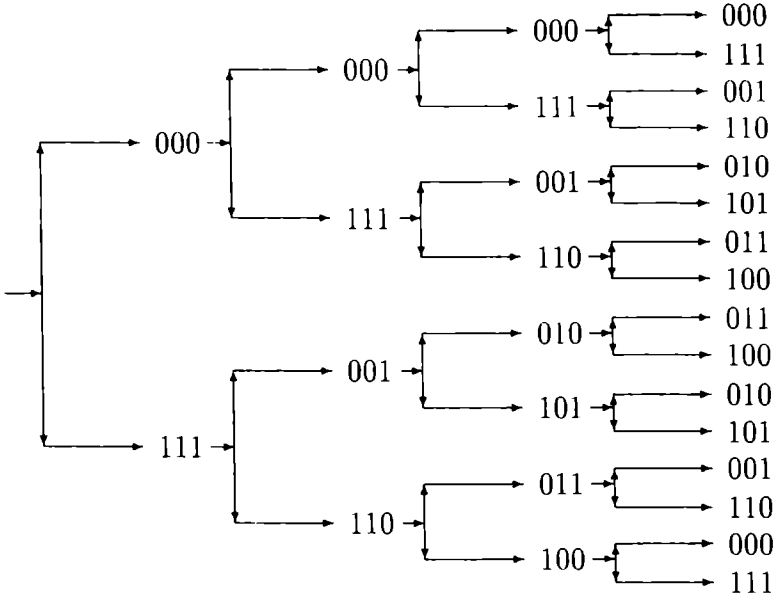


Figura 12.2:



12.6 Strategia de corectare a erorilor

Algoritmii de decodificare pentru codurile convoluţionale folosesc în general aceeaşi strategie de la codurile liniare: cuvântul primit este decodificat în cel mai apropiat cuvânt - cod, în sensul distanţei Hamming.

Specific acestei clase de coduri este faptul că nu se decodifică deodată tot cuvântul recepţionat, ci pas cu pas: la fiecare tact este decodificat un bloc de n caractere (conţinutul unui buffer de ieşire) în k simboluri (mărimea bufferului de intrare). Distanţa Hamming (notată cu d_H) pentru un (n,k) - cod convoluţional se defineşte

$$d_H(a(X),b(X)) = d(a_0a_1\ldots a_{n-1}, b_0b_1\ldots b_{n-1}).$$

Observaţia 12.3 *În cele ce urmează vom continua să identificăm secvenţele de $p+1$ caractere $\mathbf{a} = a_0a_1\ldots a_p$ cu polinoamele de gradul p $a(X) = a_0 + a_1X + \ldots + a_pX^p$. Se va folosi adesea chiar notaţia (neambiguă) $a(X) = \mathbf{a}$.*

Fie $a(X)$ mesajul sursă; un (n,k) - cod convoluţional C îl codifică în $v(X) = C(a(X))$. Să presupunem că se recepţionează cuvântul $u(X)$. La primul tact se determină cel mai apropiat cuvânt - cod $w(X)$ (cu $d_H(u(X),w(X))$ minim), iar $u_0u_1\ldots u_{n-1}$ se transformă în $w_0w_1\ldots w_{n-1}$. Dacă decodificarea este corectă, atunci s-au determinat primele n caractere şi se pot găsi imediat caracterele de informaţie $a_0a_1\ldots a_{k-1}$ (conţinutul primei intrări). Se defineşte apoi

$$u(X) := u(X) - C(a_0 + a_1X + \ldots + a_{k-1}X^{k-1}),$$

care aduce 0 pe primele n poziţii ale cuvântului recepţionat, se ignoră aceste prime n caractere şi procedeul continuă inductiv.

Motivaţia constă în faptul că acum nu se lucrează cu $v(X) = C(a(X))$ ci cu

$v(X) - C(a_0 + a_1X + \ldots a_{k-1}X^{k-1}) = C(a(X) - a_0 - a_1X - \ldots - a_{k-1}X^{k-1}) = C(a_kX^k + a_{k+1}X^{k+1} + \ldots) = C(X^k(a_k + a_{k+1}X + a_{k+2}X^2 + \ldots)) = X^nC(a_k + a_{k+1}X + a_{k+2}X^2 + \ldots)$.

Exemplul 12.13 *Să considerăm $(2,1)$ - codul convoluţional binar, de polinom generator $g_0(X) = 1 + X + X^3$. Primele lui cuvintele - cod sunt*

$$\begin{aligned} C(0) &= \mathbf{0} & 000000000\ldots \\ C(1) &= g_0(X) & 110100000\ldots \\ C(01) &= X^2g_0(X) & 001101000\ldots \\ C(11) &= (1+X^2)g_0(X) & 111001000\ldots \end{aligned}$$

Dacă se primeşte - de exemplu - $\mathbf{u} = 11110001$, vom determina cel mai apropiat cuvânt - cod, care este $(1 + X^2 + X^4)g_0(X)$, adică 11101001. Primul grup de $n = 2$ biţi este 11, deci primul bit de informaţie ($k = 1$) este 1. Fie acum

$$\mathbf{u} := \mathbf{u} - C(1) = 11110001 - 11010000 = 00100001.$$

Se ignoră primele două simboluri din noul $\mathbf{u}$ şi se determină cel mai apropiat cuvânt - cod de 100001; acesta este $000\ldots0$; deci al doilea grup cadru este 00 - adică al doilea simbol de informaţie este 0.

Refacem $\mathbf{u} := \mathbf{u} - X^2C(0) = 00100001$. Se şterg primele patru simboluri din $\mathbf{u}$ şi se caută cel mai apropiat cuvânt - cod de 0001. Acesta este din nou $00\ldots$. Deci întregul mesaj $\mathbf{u}$ se corectează în 11000000, care corespunde mesajului de informaţie 100, pentru că

$$C(1 + 0X + 0X^2) = 11000000.$$

Definiţia 12.4 *Pentru un (n, k) - cod convoluţional se defineşte distanţa liberă prin*

$$d_{lib} = min\{d_H(C(a(X)), C(a'(X)))|a_0a_1\ldots a_{k-1} \neq a'_0a'_1\ldots a'_{k-1}\}.$$

Similar observaţiei de la codurile liniare, d_{lib} este cea mai mică pondere a unui cuvânt - cod $C(a''(X))$ cu proprietatea $a''_0a''_1\ldots a''_{k-1} \neq \mathbf{0}$ (s-a notat $a''(X) = a(X) - a'(X)$ unde $a(X), a'(X)$ sunt două mesaje de informaţie arbitrare distincte).

Exemplul 12.14 *Reluând (2, 1) - codul construit în Exemplul 12.13, acesta are $d_{lib} = 3$, deoarece $g_0(X)$ are ponderea 3 şi orice cuvânt $C(a_0 + a_1X + \ldots)$ cu $a_0 \neq 0$ are ponderea minim 3.*

Propoziţia 12.1 *Un cod convoluţional corectează t erori dacă şi numai dacă $d_{lib} > 2t$.*

Demonstraţie: Să presupunem $d_{lib} > 2t$. La primirea unui cuvânt $v(X)$ cu cel mult t erori, există un cuvânt - cod $u(X) = C(a(X))$ aflat la o distanţă Hamming minimă de acesta: $d_H(u(X), v(X)) = s$. Dacă $u'(X) = C(a'(X))$ este cuvântul - cod trimis prin canal (recepţionat drept $v(X)$), din ipoteză $d_H(v(X), u'(X)) \leq t$. Deci, cu inegalitatea triunghiului, $d(u(X), u'(X)) \leq s + t$. Cum s este cea mai mică distanţă Hamming, avem $s \leq t$, deci $d(C(a(X)), C(a'(X))) \leq s + t \leq 2t < d_{lib}$. Din definiţia distanţei libere rezultă că primele k simboluri din $a(X)$ au drept cel mai apropiat grup de k simboluri generat de cod grupul primelor k simboluri din $a'(X)$, ceea ce permite decodificarea şi corectarea erorilor.

Pentru următoarele grupuri, procedeul decurge analog.

Invers, să presupunem $d_{lib} \leq 2t$ şi fie $u(X) = C(a(X))$, $u'(X) = C(a'(X))$ cu $d_H(u(X), u'(X)) = d_{lib}$ şi $a_0 \dots a_{k-1} \neq a'_0 \dots a'_{k-1}$. Fie $i_1, i_2, \dots i_{d_{lib}}$ toţi indicii j $(0 \leq j \leq n-1)$ pentru care $u_j \neq u'_j$. Construim următorul cuvânt $\mathbf{v} = v_0 v_1 \dots v_{n-1}$:

$$v_j = \begin{cases} u_j & \text{dacă } u_j = u'_j \text{ sau } j = i_{2s+1}, \\ u'_j & \text{dacă } j = i_{2s} \end{cases} \qquad (0 \leq j \leq n-1)$$

Avem $d_H(u'(X), v(X)) \leq d_H(u(X), v(X)) \leq t$. Presupunem că a fost trimis $u(X)$ şi recepţionat $v(X)$. Atunci eroarea, care a modificat cel mult t simboluri - poate conduce la o decodificare incorectă, deoarece $u'(X)$ este cuvântul - cod cel mai apropiat.

Deci, în această ipoteză, codul nu va putea corecta t erori. □

Din analiza de sus rezultă că un parametru important pentru un cod convoluţional este distanţa liberă, care specifică numărul maxim de erori care pot fi corectate. Din nefericire ([1]), nu se cunoaşte nici o metodă analitică de construcţie a unor coduri convoluţionale "bune", în sensul unei maximizări a d_{lib} pentru n şi k date. Folosind calculatorul, au fost găsite însă o serie de astfel de coduri bune. Listăm câteva din ele pentru cazul binar şi $k = 1$:

n	d_{lib}	$g_0(X)$
2	5	$110111 = 1 + X + X^3 + X^4 + X^5$
2	6	11110111
2	7	1101011011
2	8	1101110110 11
2	10	11011111001011
2	10	111011110111
3	8	111011111
3	10	111011101111
3	12	111011101011111
4	10	111101111111
4	13	1111011110011111

12.7 Decodificarea codurilor sistematice

Păstrând similitudinea cu codurile liniare, să folosim şi la codurile convoluţionale sistematice ideea de decodificare bazată pe calculul sindromului.

Fie un (n, k) - cod convoluţional sistematic, definit prin matricea generatoare G_∞ şi cea de control H_∞, construite pe baza subgeneratorilor

de lungime N $g(i,j)$, $1 \le i \le k$, $1 \le j \le n-k$. Să presupunem că $\mathbf{a}$ este mesajul de informaţie, iar $\mathbf{c}$ codificarea sa. Se ştie că din relaţiile $G_\infty H_\infty^T = \mathbf{0}$ şi $\mathbf{c} = \mathbf{a}G_\infty$ se obţine $\mathbf{c}H_\infty^T = \mathbf{0}$.

La recepţie s-a primit secvenţa (infinită) $\mathbf{r} = \mathbf{c} + \mathbf{e}$ unde $\mathbf{e}$ este un vector - eroare. Ca şi la codurile liniare, sindromul va fi

$$\mathbf{s} = \mathbf{r}H_\infty^T = \mathbf{c}H_\infty^T + \mathbf{e}H_\infty^T = \mathbf{e}H_\infty^T.$$

Secvenţa recepţionată se poate structura în blocuri de n caractere:

$$\mathbf{r} = \underbrace{r_0(1)\dots r_0(n)}_{\mathbf{r_0}}\underbrace{r_1(1)\dots r_1(n)}_{\mathbf{r_1}}\dots\underbrace{r_p(1)\dots r_p(n)}_{\mathbf{r_p}}\dots$$

Similar se segmentează sindromul în blocuri de lungime $n-k$:

$$\mathbf{s} = \underbrace{s_0(1)\dots s_0(n-k)}_{\mathbf{s_0}}\underbrace{s_1(1)\dots s_1(n-k)}_{\mathbf{s_1}}\dots\underbrace{s_p(1)\dots s_p(n-k)}_{\mathbf{s_p}}\dots$$

Vom avea

$$s_p(j) = r_p(k+j) - \sum_{i=1}^{k} r_p(i)g_0(i,j) - \sum_{i=1}^{k} r_{p-1}(i)g_1(i,j) - \dots$$

$$- \sum_{i=1}^{k} r_{p-N+1}(i)g_{N-1}(i,j) \quad (1 \le j \le n-k).$$

Această relaţie se poate scrie şi $s_p(j) = r_p(k+j) - A_p(j)$, unde

$A_p(j) = \sum_{t=0}^{N-1}\sum_{i=1}^{k} r_{p-t}(i)g_t(i,j)$ este rezultatul codificării secvenţei $r_p(1)\dots r_p(k)$ folosind matricea G_∞, în funcţie de

$$r_{p-1}(1),\dots,r_{p-1}(k),\dots,r_{p-N+1}(1)\dots,r_{p-N+1}(k).$$

Din aceste relaţii, se poate deduce o formulă pentru calculul elementelor sindromului, în funcţie de caracterele secvenţei - eroare:

$$s_p(j) = e_p(k+j) - \sum_{i=1}^{k} e_p(i)g_0(i,j) - \sum_{i=1}^{k} e_{p-1}(i)g_1(i,j) - \dots$$

$$- \sum_{i=1}^{k} e_{p-N+1}(i)g_{N-1}(i,j),\ (1 \le j \le n-k),$$

unde $\mathbf{e} = \underbrace{e_0(1)\dots e_0(n)}_{\mathbf{e_0}}\underbrace{e_1(1)\dots e_1(n)}_{\mathbf{e_1}}\dots\underbrace{e_p(1)\dots e_p(n)}_{\mathbf{e_p}}\dots$

este secvenţa de eroare - tip, segmentată în blocuri de lungime n. Deci, pentru orice $j = 1,2,\dots,n-k$, putem scrie:

$$s_0(j) = e_0(k+j) - \sum_{i=1}^{k} e_0(i)g_0(i,j),$$

$$s_1(j) = e_1(k+j) - \sum_{i=1}^{k} e_1(i)g_0(i,j) - \sum_{i=1}^{k} e_0(i)g_1(i,j),$$

$$s_{N-1}(j) = e_{N-1}(k+j) - \sum_{p=0}^{N-1}\sum_{i=1}^{k} e_p(i)g_{N-p-1}(i,j),$$

$$s_N(j) = e_N(j) - \sum_{p=0}^{N-1}\sum_{i=1}^{k} e_{p+1}(i)g_{N-p-1}(i,j),$$

$$s_{N+m}(j) = e_{N+m}(k+j) - \sum_{p=1}^{N-1}\sum_{i=1}^{k} e_{p+m-1}(i)g_{N-p-1}(i,j).$$

De remarcat că secvenţa $\mathbf{e_0}$ afectează numai primele N blocuri ale sindromului: $\mathbf{s_0}, \mathbf{s_1}, \ldots, \mathbf{s_{N-1}}$, deoarece apare numai în primele $N(n-k)$ ecuaţii din sistemul de sus.

Pentru corectarea erorilor, se determină iniţial secvenţa $\mathbf{e_0}$; după aceasta, sindromul se recalculează scăzând din el componentele lui $\mathbf{e_0}$. Această operaţie se numeşte *rearanjarea sindromului.* În paralel, $\mathbf{e_0}$ se foloseşte la corectarea primelor n caractere recepţionate; operaţia se realizează identic cu cea de la codurile liniare.

În continuare, se va folosi un nou sindrom, anume:

$$s'_0(j) = 0$$

$$s'_m(j) = e_m(k+j) - \sum_{p=0}^{m-1}\sum_{i=1}^{k} e_{m-p}(i)g_p(i,j),$$

$$(1 \le m \le N-1,\ 1 \le j \le n-k)$$

$$s'_{N+m}(j) = s_{N+m}(j), \quad \forall m \ge 0.$$

De remarcat că secvenţa $\mathbf{e_1}$ figurează numai în expresiile lui $\mathbf{s'_1}, \mathbf{s'_2}, \ldots, \mathbf{s'_N}$ unde $\mathbf{s'_i} = s'_i(1)\ldots s'_i(n-k)$ $(1 \le i \le N)$.

Deci aceste componente apar numai în $N(n-k)$ ecuaţii. După determinarea lui $\mathbf{e_1}$ se corectează blocul $\mathbf{r_1}$ din secvenţa recepţionată, apoi se recalculează sindromul ş.a.m.d.

12.8 Algoritmul de decodificare Viterbi

Cel mai cunoscut algoritm de decodificare pentru codurile convoluţionale aparţine lui Viterbi şi a fost folosit pe scară largă în comunicaţiile spaţiale. Astfel - ca să dăm numai un exemplu, staţia Voyager în misiunea sa din 1974 spre Marte, Saturn şi Jupiter, a folosit pentru transmisii un $(2,1)$ - cod convoluţional binar de polinom generator $g_0(X) = 1 + X + X^2 + X^5 + X^6 + X^7 + X^8 + X^{10} + X^{11} + X^{12}$.

În descrierea algoritmului Viterbi vom folosi o reprezentare în *reţea* a codurilor convoluţionale, care reia sub o altă formă reprezentarea arborescentă. Din nou, ne vom mărgini la construcţii pentru $(n,1)$ - coduri convoluţionale extensia la cazul general fiind imediată.

Definiţia 12.5 *Se numeşte "stare" a unui circuit liniar, secvenţa conţinuturilor elementelor sale de înmagazinare la un anumit moment.*

Un circuit liniar cu m elemente de înmagazinare are q^m stări posibile (se presupune că intrarea şi ieşirea sunt secvenţe de caractere din Z_q).

Starea $00\ldots0$ în care se află circuitul la momentul 0 se numeşte *stare iniţială.*

Să notăm cu $Q = Z_q^N$ mulţimea celor q^N stări posibile ale circuitului liniar asociat unui $(n,1)$ - cod convoluţional de memorie N. Comportarea acestui circuit poate fi exprimată prin două aplicaţii, caracteristice automatelor Mealy.

$$\delta : Q \times Z_q \longrightarrow Q \text{ (funcţia de tranziţie)},$$
$$\lambda : Q \times Z_q \longrightarrow Z_q^n \text{ (funcţia de ieşire)},$$

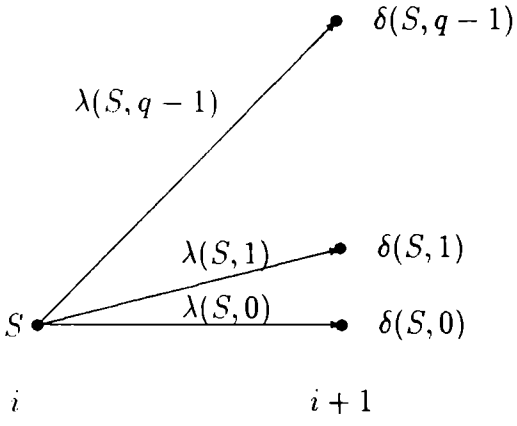
definite astfel:

Fie momentul i, în care automatul se află în starea S şi primeşte la intrare caracterul a; atunci

$\delta(S,a)$ este starea în care trece circuitul la momentul $i+1$,

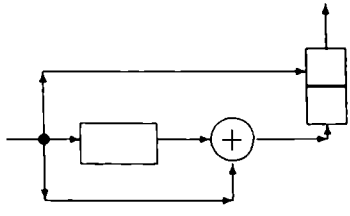
$\lambda(S,a)$ este secvenţa de n caractere care constituie ieşirea circuitului la momentul i. Iniţial $\lambda(S_0,a) = C(a)$ (dacă S_0 este starea iniţială).

O *diagramă reţea* este un graf orientat infinit, care are ca noduri stările unui circuit liniar la fiecare moment (tact). Un nod marcat care corespunde unei stări S la momentul i, este legat direct cu alte q noduri, corespunzătoare stărilor circuitului la momentul $i+1$:



Grafic, arcul $S \longrightarrow \delta(S, p)$ va fi totdeauna trasat sub arcul $S \longrightarrow \delta(S, p+1)$ $(0 \leq p \leq q-1)$. Fiecare arc este marcat cu secvenţa de ieşire a circuitului pentru intrarea corespunzătoare.

Exemplul 12.15 $(2,1)$ - *codul convoluţional binar, definit de circuitul liniar*



are două stări posibile (după conţinutul elementului de înmagazinare); deci $Q = \{0, 1\}$.

Pentru intrarea i $(i = 0, 1)$ *ieşirea este secvenţa*

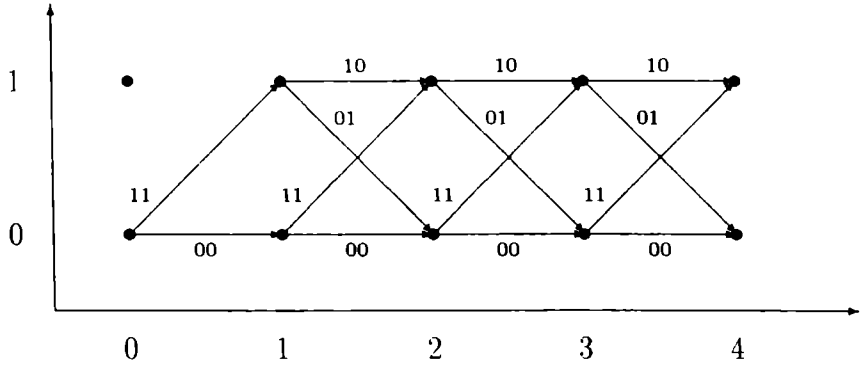
ii	*dacă starea este* 0,
$i(1-i)$	*dacă starea este* 1.

Cele două aplicaţii δ *şi* λ *sunt definite*

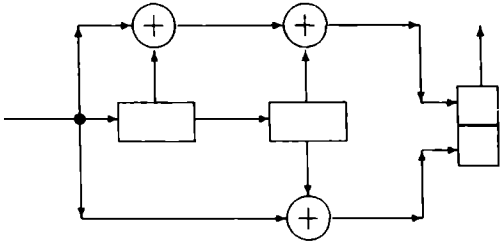
δ	0	1
0	0	1
1	0	1

λ	0	1
0	00	11
1	01	10

De aici rezultă imediat diagrama reţea a codului:



Exemplul 12.16 $(2,1)$ - *codul convoluţional binar reprezentat prin circuitul liniar*



are două elemente de înmagazinare, deci patru stări: $Q = \{00, 01, 10, 11\}$ *Studiind comportarea sa, se construiesc cele două funcţii (reprezentate într-un singur tabel):*

δ/λ	0	1
00	00/00	10/11
01	00/11	10/00
10	00/10	11/01
11	01/01	11/10

Se ajunge deci la diagrama reţea din Figura 12.3:

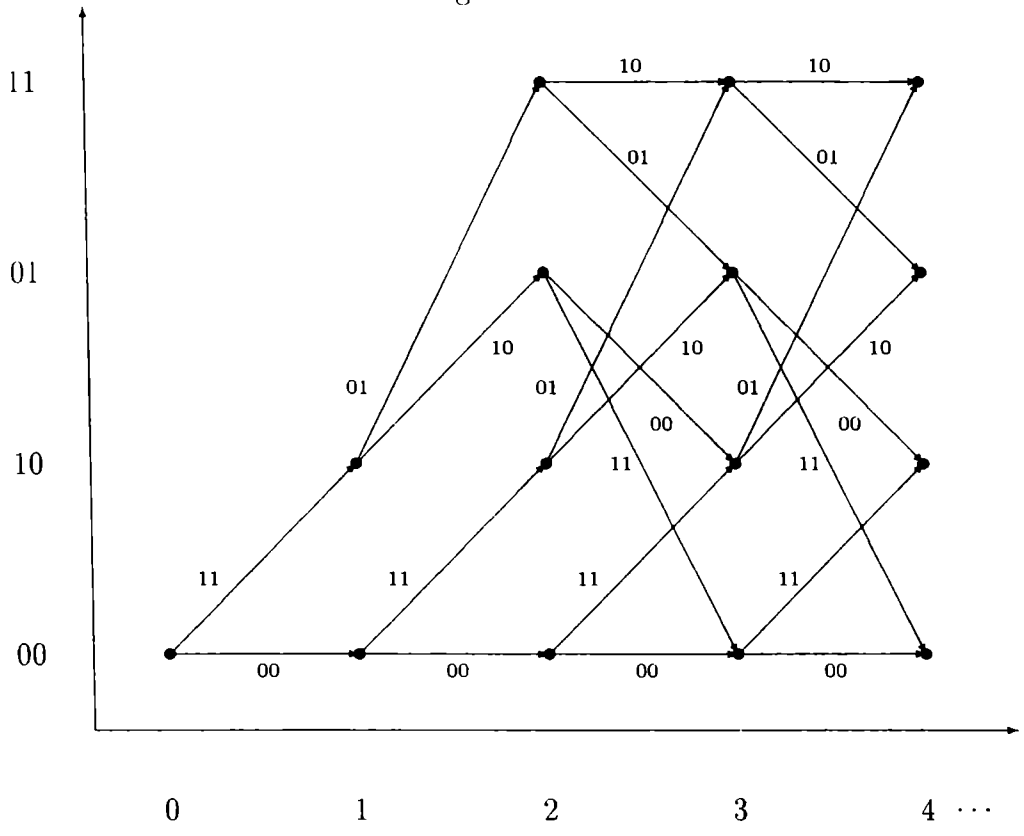
Orice drum prin reţea, care pleacă din momentul 0 *conduce - prin citirea marcajelor arcelor - la un cuvânt - cod. De exemplu, pentru intrarea* 100110, *codul generat este* $C(100110) =$ 11 10 11 11 01 01 *(reamintim, spre dreapta pleacă totdeauna două arce, iar arcul pentru intrarea* 0 *este situat sub arcul pentru intrarea* 1*).*

Invers, orice cuvânt - cod reprezintă marcajul unui drum în reţea.

Codul generat de o secvenţă de intrare poate fi obţinut şi prin utilizarea directă a celor două aplicaţii. Pentru aceasta, va trebui să extindem funcţia de ieşire $\lambda : Q \times Z_q^* \longrightarrow (Z_q^n)^*$, astfel:

$$\lambda(S, a\alpha) = \lambda(S, a)\lambda(\delta(S, a), \alpha), \qquad \forall \alpha \in Z_q^+.$$

Figura 12.3:



Corectitudinea acestei extensii este imediată.

Exemplul 12.17 *Să reluăm exemplul anterior; pentru intrarea* 100110, *ieşirea va fi (reamintim,* 00 *este starea iniţială)*

$\lambda(00, 100110) = \lambda(00,1)\lambda(\delta(00,1), 00110) = 11\lambda(10, 00110) = 11\lambda(10,0)\lambda(\delta(10,0), 0110) = 11\ 10\lambda(01, 0110) = 11\ 10\lambda(01,0)\lambda(\delta(01,0), 110) = 11\ 10\ 11\lambda(00, 110) = 11\ 10\ 11\lambda(00,1)\lambda(\delta(00,1), 10) = 11\ 10\ 11\ 11\lambda(10, 10) = 11\ 10\ 11\ 11\lambda(10,1)\lambda(\delta(10,1), 0) = 11\ 10\ 11\ 11\ 01\ \lambda(11, 0) = 11\ 10\ 11\ 11\ 01\ 01$

Să construim acum algoritmul de decodificare Viterbi.

La recepţia unui cuvânt $\mathbf{v} = \mathbf{v_0 v_1 v_2} \ldots$ vom căuta construcţia unui drum prin diagrama reţea, cât mai apropiat de $\mathbf{v}$.

Pentru fiecare moment i şi stare S vom lista drumurile *active* prin reţea până la starea S la momentul i. Un drum este *activ* dacă *discrepanţa* sa este minimă (prin *discrepanţă* se înţelege distanţa Hamming dintre

cuvântul - cod generat de drumul prin arbore şi cuvântul de aceeaşi lungime primit la intrare). Vom adnota fiecare stare S cu discrepanţa drumului activ de la starea iniţială S_0 la S. Pentru calculul discrepanţei se poate imagina o formulă recursivă, definită astfel:

$$di(S') = \min_{\substack{(S,a) \in Q \times Z_q, \\ \delta(S,a) = S'}} \{di(S) + d_H(\lambda(S,a), \mathbf{v})\} \qquad (6)$$

La primirea secvenţei $\mathbf{v}$ de n caractere, drumul activ - aflat în starea S - continuă cu arcul $S \to S'$, marcat $\lambda(S,a)$ (care a codificat caracterul de informaţie $a \in Z_q$). Sunt păstrate ca drumuri active acele drumuri care determină pentru S' o discrepanţă minimă.

Algoritmul va avea o durată de funcţionare finită numită *fereastră de decodificare*, pe care o notăm cu b (deci va funcţiona b tacţi).

Algoritmul Viterbi:
Intrare: Secvenţa $\mathbf{v_0 v_1} \ldots$, $\mathbf{v_p} = v_{p,0} v_{p,1} \ldots v_{p,n-1}$;
Algoritm:

1. Fie $p := 0$ şi $S_0 := 00 \ldots 0$ starea iniţială a reţelei;
2. Se marchează $di(S_0) = 0$;
3. **for** $i := 1$ **to** b **do**

   Pentru toate stările S accesibile la momentul $p + i$ se determină $di(S)$ folosind formula (6), şi se listează toate drumurile active care duc la S.
4. Dacă toate drumurile active la momentul $p+b$ încep cu acelaşi arc $S_0 \to S_1$, secvenţa de intrare $\mathbf{v_p}$ se corectează în marcajul $\mathbf{u_p}$ al acestui prim arc (şi se decodifică în primul caracter al lui S_1). Altfel, eroarea nu este corectabilă, STOP.
5. $p := p + 1$, $S_0 := S_1$, salt la pasul 3.

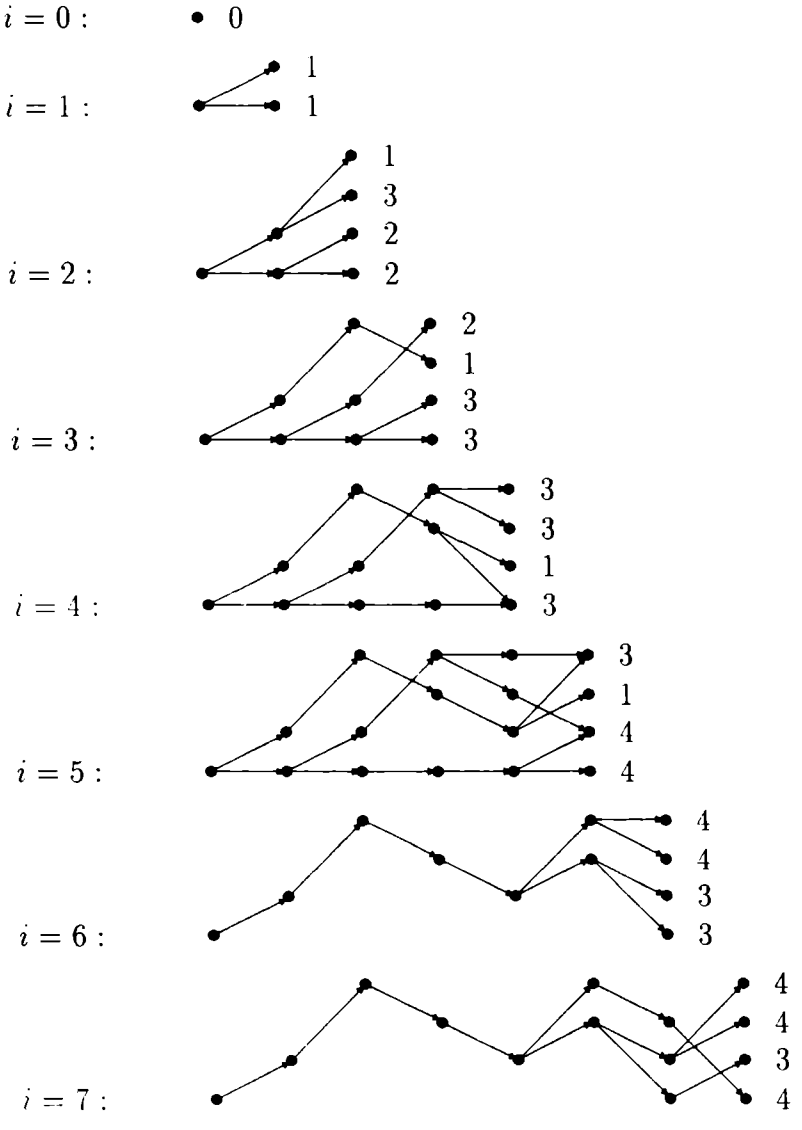
Exemplul 12.18 *Să reluăm codul descris în Exemplul 12.16, împreună cu diagrama sa de reţea. Presupunem că s-a primit secvenţa*

$$\mathbf{v} = 10010100101100000 \ldots$$

Deci

$\mathbf{v_0} = 10,\ \mathbf{v_1} = 01,\ \mathbf{v_2} = 01,\ \mathbf{v_3} = 00,\ \mathbf{v_4} = 10,\ \mathbf{v_5} = 11,\ \mathbf{v_6} = 00\ldots.$

Vom lucra cu o fereastră de decodificare $b = 7$. *Detaliem grafic numai procedura de decodificare a primului bloc (cazul* $p = 0$*).*



Se ajunge la concluzia că toate arcele active au acelaşi început: arcul marcat cu 11. *Deci primul bloc* $\mathbf{v_0} = 10$ *se decodifică în* 11, *după care se ia ca nod iniţial* $S_0 = 10$ *şi procesul se repetă.*

De remarcat că pentru o fereastră de decodificare $b = 5$, *eroarea nu s-ar fi putut corecta (primul arc nu este unic).*

În această prezentare a algoritmului Viterbi, apare o problemă: cât de mare este fereastra b ? Cât de departe mergem prin diagrama reţea până să fim siguri că totuşi este posibilă corectarea erorilor ?

Pentru aceasta să refacem sub o formă finită reprezentarea diagramelor - reţea.

Vom construi un translator (*diagramă de translatare*) asociat unui $(n,1)$ - cod convoluţional, astfel:

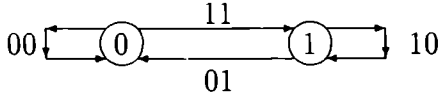
Stările translatorului sunt N - tupluri (elemente din Z_q^N), fiecare stare reprezentând o situaţie posibilă a celor N elemente de înmagazinare. Practic, toate stările aflate pe aceeaşi linie în diagrama reţea se identifică printr-o singură stare, notată cu secvenţa aflată pe coloana de identificare a diagramei.

O stare $S = s_1 \dots s_{N-1}s_N$ este legată direct prin arcul marcat $\mathbf{x} = x_1 \dots x_n$ de starea $S' = is_1 \dots s_{N-1}$ dacă în circuitul liniar corespunzător codului, intrarea $i \in Z_q$ conduce la modificarea conţinuturilor elementelor de înmagazinare, din $(s_1, \dots, s_N)$ în $(i, s_1, \dots, s_{N-1})$ şi are ca efect ieşirea $\mathbf{x}$.

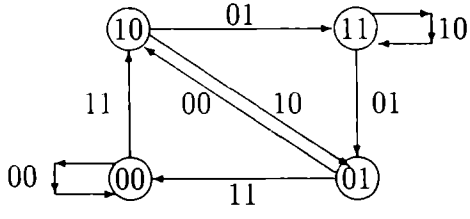
În termeni de automat Mealy,

$$\delta(S,i) = S', \qquad \lambda(S,i) = \mathbf{x}.$$

Exemplul 12.19 *Reţeaua codului definit în Exemplul 12.15 poate fi reprezentată sub formă de diagramă de translatare astfel:*



Pentru codul din Exemplul 12.16, reprezentarea finită este:



De remarcat că pe fiecare arc este suficient să marcăm doar secvenţa de ieşire; caracterul de intrare este reprezentat de primul caracter al stării în care intră arcul respectiv.

Definiţia 12.6 *Se defineşte lungimea unui drum ca fiind numărul de arce care formează acel drum.*

Ponderea unui drum este suma ponderilor marcajelor arcelor care formează drumul.

Reamintim, *ponderea unui cuvânt* este numărul de caractere nenule din cuvântul respectiv.

Conform Definiţiei 12.4, distanţa liberă a unui cod este ponderea nenulă minimă a unui cuvânt cod - deci a unei secvenţe care marchează un drum prin diagrama de translatare.

Propoziţia 12.2 *Distanţa liberă a unui* $(n,1)$ *- cod convoluţional este egală cu ponderea nenulă minimă a unui ciclu care trece prin starea iniţială* $00\ldots0$ *a diagramei de translatare a codului.*

Demonstraţie: Este imediată.

Fie $S_0 = 00\ldots0$ starea iniţială. Pentru orice t, $1 \leq t \leq \left[\frac{d_{lib}-1}{2}\right]$, se defineşte $d(t)$ ca fiind cel mai mic număr întreg pozitiv p cu proprietatea că orice drum $S_0 \to S_1 \to \ldots \to S_{p-1}$, $(S_0 \neq S_1)$ are ponderea mai mare de $2t$. Pentru determinarea algoritmică a lui $d(t)$ se poate adapta imediat un algoritm similar din teoria grafurilor.

Deoarece codul este liniar, valoarea lui $d(t)$ este aceeaşi pentru orice stare S a diagramei de translatare.

Acum, în condiţiile că pot apare maxim t erori, algoritmul Viterbi va funcţiona corect pentru o fereastră $b = d(t)$ (datorită Propoziţiei 12.1).

Reluarea algoritmului Viterbi la fiecare pas cu fereastra maximă $d(t)$ este însă destul de costisitoare ca timp; de aceea, practic, se foloseşte o fereastră mobilă $b \leq d(t)$ şi algoritmul trece la faza de decodificare atunci când toate drumurile active selectate au acelaşi prim arc. Dacă s-a ajuns la $b = t(b)$ şi nu s-a selectat un prim arc comun, algoritmul eşuează.

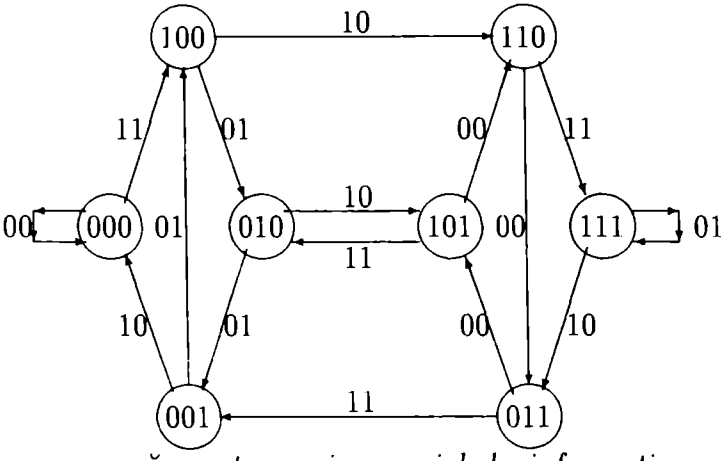
12.9 Coduri convoluţionale catastrofice

Dacă este definit neglijent, un cod convoluţional poate conduce la următoarea situaţie, complet nefericită: un mesaj în care a fost perturbat un număr mic de caractere generează prin decodificare o infinitate de erori.

Vom considera din nou numai codurile $(n,1)$ - convoluţionale binare.

Pentru a vedea în ce constă problema, să luăm următorul exemplu:

Exemplul 12.20 *Fie* $(2,1)$ *- codul convoluţional binar, cu polinoamele subgeneratoare* $g_0^0(X) = 1 + X^3, g_0^1(X) = 1 + X + X^2$. *El are* $N = 3$, *iar diagrama sa de translatare este*



Să presupunem că s-a transmis mesajul de informaţie $\mathbf{a} = 00\ldots$, *căruia îi corespunde cuvântul - cod nul* $\mathbf{c} = 0000\ldots$. *La recepţie s-a primit un cuvânt cu primele trei simboluri greşite:* $\mathbf{v} = 111000\ldots$. *Decodificarea sa nu ridică nici o problemă, acesta fiind de asemenea cuvânt - cod, care marchează drumul prin stările* $000 - 100 - 110 - 011 - 101 - 110 - \ldots$.

Deci, în ipoteza generală admisă (cea a decodificării cele mai probabile), nu au apărut erori, şi mesajul decodificat este $\mathbf{a}' = 110110110\ldots$, *care diferă de* $\mathbf{a}$ *într-o infinitate de poziţii.*

În acest mod, modificarea doar a primelor trei caractere a dus la situaţia în care aceste erori se propagă într-o secvenţă infinită.

Se observă din acest exemplu că tot necazul provine din faptul că diagrama conţine două cicluri distincte cu ieşirea 0: $000 - 000$ şi $110 - 011 - 101 - 110$.

Definiţia 12.7 *Un cod convoluţional se numeşte catastrofic dacă diagrama sa de translatare conţine două cicluri distincte de pondere zero.*

Observaţia 12.4 *Orice cod convoluţional conţine un ciclu de pondere zero, anume bucla care pleacă şi intră în starea iniţială. Reamintim, ponderea unui drum este suma ponderilor marcajelor arcelor sale.*

Teorema 12.2 *Un* $(n, 1)$ *- cod convoluţional este catastrofic dacă şi numai dacă* $cmmdc(g_0^0(X), \ldots, g_0^{n-1}(X)) \neq 1$.

Demonstraţie: Teorema rămâne valabilă dacă o demonstrăm pentru cazul $n = 2$. Fie deci un $(2, 1)$ - cod convoluţional, de polinoame subgeneratoare $g_0^0(X)$, $g_0^1(X)$. Pentru a marca un ciclu, un cuvânt de intrare trebuie să fie de forma $b(X) + X^r a(X)(1 + X^p + X^{2p} + \ldots)$, cu $grad(a(X)) < p$.

Pentru a elimina din discuţie bucla din starea iniţială S_0, vom considera $a(X) \not\equiv 0$. Să presupunem că în diagrama de translatare a codului există un ciclu de lungime p şi pondere 0. Atunci $a(X)g_0^0(X)$ şi $a(X)g_0^1(X)$ trebuie să se dividă cu $1 + X^p$, deci $a(X) \cdot g'(X)$ (unde $g'(X) = cmmdc(g_0^0(X), g_0^1(X))$) se divide cu $1 + X^p$.

Dacă $g'(X) = 1$, se ajunge la o contradicţie (deoarece $a(X)$ are grad mai mic decât p şi nu este polinomul identic nul). Deci $grad(g'(X)) \geq 1$. În acest caz, toate mesajele de informaţie $b(X) + a(X)[g'(X)]^t(1 + X^p + X^{2p} + \ldots)$, $(t \geq 0)$ se codifică identic, ceea ce va conduce la imposibilitatea decodificării.

Reciproca se demonstrează similar. □

12.10 Exerciţii

12.1 *Construiţi un codificator pentru* $(2,1)$ *- codul convoluţional binar de polinom generator* $g_0(X) = 1 + X + X^3 + X^4 + X^6 + X^7 + X^9$. *Cât este* N *? Codificaţi mesajele* 110, 0110, 1010.

12.2 *Construiţi o matrice generatoare pentru codul definit anterior.*

12.3 *Construiţi un codificator pentru* $(2,1)$ *- codul convoluţional ternar de polinom generator* $g_0(X) = 1 + 2X + 2X^3$.

Să se determine o matrice generatoare.

Să se codifice mesajele 102, 200, 0120.

12.4 *Construiţi un codificator pentru un* $(3,2)$ *- cod convoluţional binar cu* $N = 2$.

12.5 *Codificatorul din Exemplul 12.9 are în total* 5 *elemente de înmagazinare. Construiţi un codificator pentru acelaşi cod, care utilizează doar* 4 *elemente de înmagazinare.*

12.6 *Construiţi un codificator pentru* $(4,3)$ *- codul convoluţional binar, de polinoame generatoare*

$$g_0(X) = 1 + X + X^3 + X^5 + X^6,\ g_1(X) = X + X^2 + X^3 + X^4,$$
$$g_2(X) = 1 + X^2 + X^3 + X^5 + X^6.$$

12.7 *Să se construiască matricea generatoare şi de control pentru* $(3,2)$ *- codul convoluţional sistematic ternar de subgeneratori* $g(1,1) = 1200$, $g(2,1) = 2011$. *Să se codifice mesajele* 11220, 1012, 2222.

Să se deseneze codificatorul acestui cod.

12.8 *Aceeaşi problemă pentru* $(4,2)$ *- codul sistematic binar cu* $g(1,1) = 001$, $g(1,2) = 110$, $g(2,1) = 101$, $g(2,2) = 111$.
Să se codifice mesajele 1100, 011, 001100.
Să se construiască reprezentarea arborescentă.

12.9 *Să se demonstreze formulele* (4).

12.10 *Determinaţi* d_{lib} *pentru fiecare din codurile următoare:*
(a) $g_0^0(X) = 1 + X^2$, $g_0^1(X) = 1 + X + X^2$;
(b) $g_0^0(X) = 1 + X + X^2 + X^3$, $g_0^1(X) = 1 + X^2 + X^3$;
(c) $g_0^0(X) = 1 + X^3 + X^4$, $g_0^1(X) = 1 + X + X^2 + X^4$.

12.11 *Folosind* $(2,2)$ *- codul convoluţional definit în Exemplul 12.16, şi fereastra de decodificare* $b = 7$, *decodificaţi (cât este posibil) secvenţele*
$\mathbf{v} = 01000001000\ldots$, $\mathbf{v} = 11000110010010001110010\ldots$

12.12 *Trasaţi diagrama - reţea a* $(2,1)$ *- codului convoluţional binar, generat de* $g_0(X) = 1 = X^2 + X^3 + X^4$. *Folosiţi algoritmul Viterbi pentru decodificarea secvenţei* $\mathbf{v} = 1000001000001000\ldots$

12.13 *Fie* $(2,1)$ *- codul convoluţional binar, generat de* $g_0^0(X) = 1 + X + X^2 + X^3$, $g_0^1(X) = 1 + X^2 + X^3$. *Decodificaţi primele* 4 *caractere de informaţie ale cuvântului recepţionat* $\mathbf{c} = 11000000\ldots$ *pentru*
(a) $b = 2$; (b) $b = 3$; (c) $b = 4$.

12.14 *Demonstraţi Propoziţia 12.2.*

12.15 *Pentru codurile definite în Exerciţiul 12.10, determinaţi diagramele de translatare şi* $d(t)$ *pentru* $1 \leq t \leq \left[\frac{d_{lib} - 1}{2}\right]$.

12.16 *Ce se întâmplă dacă se încearcă să se determine* $d(t)$ *pentru* $t > \left[\frac{d_{lib} - 1}{2}\right]$ *?*

12.17 *Construiţi un algoritm pentru determinarea lui* $d(t)$ *în cazul unui* $(n,1)$ *- cod convoluţional* $(1 \leq t \leq \left[\frac{d_{lib} - 1}{2}\right]$ *dat).*

12.18 *Pentru fiecare din codurile următoare, decideţi dacă sunt catastrofice sau nu. În caz afirmativ, determinaţi ciclurile de pondere* 0.
(a) $g_0^0(X) = 1 + X$, $g_0^1(X) = 1 + X + X^2 + X^3$;
(b) $g_0^0(X) = 1 + X + X^4$, $g_0^1(X) = 1 + X^2 + X^4$;
(c) $g_0^0(X) = 1 + X + X^2$, $g_0^1(X) = 1 + X + X^3 + X^4$.

Capitolul 13

Supra - codificări

Ideea de bază în teoria codurilor constă – după cum s-a văzut – în generarea de secvenţe care să poată păstra nealterată la transmisia prin canal cât mai multă informaţie. Atât codurile liniare (care pot fi numite *coduri bloc* datorită spargerii mesajului în mai multe submesaje de lungimi egale) cât şi cele convoluţionale satisfac această condiţie.

Aplicaţiile actuale ale codurilor (telefonia mobilă, înregistrarea pe *CD* - uri, poşta electronică) conduc la transmiterea cu viteze mari a unei cantităţi de mesaje, incredibilă acum câteva decenii. Tipurile de canal s-au modificat şi ele, apărând filtre de bandă care să elimine anumite perturbaţii. Mai mult, algoritmii de decodificare trebuie să fie rapizi, aceasta fiind uneori o cerinţă expresă a aplicaţiei. În aceste condiţii este adesea necesar ca asupra cuvintelor - cod să se aplice o codificare suplimentară – uneori un alt cod care să corespundă unor cerinţe sporite de securitate. Vom trata în acest capitol unele modalităţi de supra-codificare a codurilor (bloc sau convoluţionale), folosite pe scară largă în acest moment.

13.1 Transpunere

O metodă eficientă de utilizare a capacităţilor de corectare a pachetelor - tip de erori de către codurile bloc este constă în "interpătrunderea" cuvintelor bloc; vom numi această operaţie *transpunere* (*interleaving*).

În mod natural, mesajele de informaţie $\mathbf{m_1}, \mathbf{m_2}, \ldots$ sunt codificate în cuvintele - cod $\mathbf{c_1}, \mathbf{c_2}, \ldots$ şi transmise prin canal în ordinea codificării. Transmisia se poate realiza însă şi după o rearanjare a caracterelor din mai multe cuvinte - cod consecutive.

Formal, prin *transpunere la adâncimea* s se înţelege scrierea a s cuvinte - cod $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_s}$ ca linii ale unei matrici

$$X_{s,n} = \begin{pmatrix} c_{11} & c_{12} & c_{13} & \ldots & c_{1n} \\ c_{21} & c_{22} & c_{23} & & c_{2n} \\ & & \vdots & & \\ c_{s1} & c_{s2} & c_{s3} & & c_{sn} \end{pmatrix}$$

şi transmiterea a n mesaje de lungime s, formate din coloanele matricii $X_{s,n}$:

$$c_{11}c_{21}\ldots c_{s1}, \quad c_{12}c_{22}\ldots c_{s2}, \quad \ldots, \quad c_{1n}c_{2n}\ldots c_{sn}.$$

Exemplul 13.1 *Să considerăm un* $(6,3)$ *- cod liniar binar generat de matricea*

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

şi şase mesaje codificate $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_6}$, *unde*

$$\mathbf{c_1} = 100110, \quad \mathbf{c_2} = 010101, \quad \mathbf{c_3} = 111000,$$
$$\mathbf{c_4} = 010101, \quad \mathbf{c_5} = 100110, \quad \mathbf{c_6} = 111000.$$

Deci - scrisă la nivel de caracter - secvenţa transmisă este:

100110 010101 111000 010101 100110 111000.

Printr-o transpunere la adâncimea 3, *forma mesajelor este:*

101 011 001 110 100 010 011 101 001 110 010 100,

iar cu o transpunere la adâncimea 6*:*

101011 011101 001001 110110 100010 010100.

Ce efect are o transpunere la adîncimea s asupra capacităţii de corecţie de pachete de erori ale codului ? Răspunsul este dat de

Teorema 13.1 *Fie* C *un cod binar corector de pachete standard de* j *erori. Dacă* C *este transpus la adâncimea* s, *atunci toate pachetele de cel mult* sj *erori pot fi corectate, în ipoteza că fiecare cuvânt - cod este afectat de cel mult un pachet standard de* j *erori.*

Demonstraţie: Dacă primul caracter al cuvântului - cod $\mathbf{c}$ este al i - lea caracter transmis, atunci celelalte caractere apar pe poziţiile $i+s, i+2s, \ldots, i+(n-1)s$. Orice pachet de maxim sj erori va produce un pachet de maxim j erori în $\mathbf{c}$, deci $\mathbf{c}$ va putea fi regăsit la recepţie (dacă el nu a mai fost cumva afectat şi de alte pachete de erori). □

Restricţia ca fiecare cuvânt - cod să fie afectat de cel mult un pachet de erori impune condiţia ca pachetele de erori să fie separate de perioade în care transmisia este corectă, perioade suficient de lungi pentru evitarea situaţiei ca un bloc de s cuvinte să fie afectat de două pachete distincte de erori. În acest fel, crescând s, creşte şi lungimea pachetului de erori care poate fi corectat, dar creşte şi lungimea perioadelor necesare de transmisie fără erori.

Exemplul 13.2 *Codul din Exemplul 13.1 corectează o eroare. Folosind o transpunere la adâncimea* 3, *el poate corecta orice pachet standard de maxim* 3 *erori.*

13.1.1 Cadru de transpunere întârziată

Un dezavantaj al transpunerii este acela că nu se poate face transmisia până nu au fost codificate toate cele s cuvinte, fapt care nu va permite folosirea directă a circuitelor liniare. Pentru a evita acest neajuns se va folosi un s - *cadru de transpunere întârziată*, care aranjează caracterele codificate, nu sub forma unei matrici X, ci sub formă de "scară" - ca în tabelul cu n linii şi (teoretic) un număr infinit de coloane:

$$
\begin{array}{cccccccc}
c_{1,1} & c_{2,1} & c_{s+1,1} & c_{s+2,1} & c_{2s+1,1} & c_{2s+2,1} & & c_{(n-1)s+1,1} \\
 & & c_{1,2} & c_{2,2} & c_{s+1,2} & c_{s+2,2} & \cdots & c_{(n-2)s+1,2} \\
 & & & & c_{1,3} & c_{2,3} & & c_{(n-3)s+1,3} \\
 & & & & & & & \vdots \\
 & & & & & & & c_{1,n}
\end{array}
$$

Deoarece transmisia se face pe coloane, spaţiile libere trebuiesc marcate. Acest lucru se realizează introducând pe aceste poziţii un caracter special $* \notin Z_q$.

Exemplul 13.3 *Să reluăm mesajul codificat* $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_6}$ *din Exemplul 13.1. Un* 1 *- cadru de transpunere întârziată va fi*

$$
\begin{array}{cccccccccccc}
1 & 0 & 1 & 0 & 1 & 1 & \ldots & & & & & \\
* & 0 & 1 & 1 & 1 & 0 & 1 & \ldots & & & & \\
* & * & 0 & 0 & 1 & 0 & 0 & 1 & \ldots & & & \\
* & * & * & 1 & 1 & 0 & 1 & 1 & 0 & \ldots & & \\
* & * & * & * & 1 & 0 & 0 & 0 & 1 & 0 & \ldots & \\
* & * & * & * & * & 0 & 1 & 0 & 1 & 0 & 0 &
\end{array}
$$

Scvenţa de caractere transmise este

$$1 * * * * * 00 * * * * 110 * * * 0101 * * 11111 * 100000 \ldots$$

Dacă se foloseşte un 2 - cadru de transpunere întârziată, avem

$$\begin{array}{cccccccccccccccc}
1 & 0 & 1 & 0 & 1 & 1 & \ldots & & & & & & & & & \\
* & * & 0 & 1 & 1 & 1 & 0 & 1 & \ldots & & & & & & & \\
* & * & * & * & 0 & 0 & 1 & 0 & 0 & 1 & \ldots & & & & & \\
* & * & * & * & * & * & 1 & 1 & 0 & 1 & 1 & 0 & \ldots & & & \\
* & * & * & * & * & * & * & * & 1 & 0 & 0 & 0 & 1 & 0 & \ldots & \\
* & * & * & * & * & * & * & * & * & * & 0 & 1 & 0 & 1 & 0 & 0
\end{array}$$

iar secvenţa de caractere transmise este:

$$1 * * * * * 0 * * * * * 10 * * * * 01 * * * * 110 * * * 110 * * * \ldots$$

Este uşor de găsit o teoremă analogă cu Teorema 13.1:

Teorema 13.2 *Fie C un cod capabil să corecteze toate pachetele de cel mult j erori consecutive. Dacă C foloseşte un s - cadru de transpunere întârziată, atunci se poate corecta orice pachet standard de maxim $j(sn+1)$ erori, în ipoteza că orice cuvânt - cod este afectat de cel mult un pachet standard de j erori.*

Demonstraţie: Exerciţiu.

13.1.2 Transpunere încrucişată

Ca o altă facilitate, pentru codificarea unui mesaj se folosesc adesea două coduri. De exemplu, pentru codificarea muzicii pe compact - discuri se utilizează două coduri Reed - Solomon, iar NASA şi Agenţia Spaţială Europeană folosesc două coduri convoluţionale.

Fie C_i două (n_i, k_i) coduri liniare $(i = 1, 2)$. *Transpunerea încrucişată* a lui C_1 cu C_2 se realizează astfel:

1. Mesajele de informaţie sunt codificate cu C_1 iar cuvintele - cod rezultate sunt transpuse la adâncimea k_2.

2. Coloanele obţinute în acest proces de transpunere (care sunt de lungime k_2) privite ca mesaje de informaţie - sunt codificate de codul C_2.

3. Cuvintele - cod rezultate sunt transpuse la o adâncime s - fixată sau folosind un s - cadru de transpunere întârziată.

Avantajul principal al celui de-al doilea proces de codificare este următorul:

Fie d_1, d_2 distanţele minime ale celor două coduri. C_2 poate detecta $d_2 - 1$ erori (nu ne punem problema corectării lor). Dacă s-au detectat erori pentru un cuvânt - cod din C_2, atunci toate caracterele acestui cuvânt sunt marcate şi tratate drept simboluri care pot fi incorecte. Se consideră apoi cuvintele - cod din C_1. Dacă $n_1 - d_1 + 1$ caractere dintr-un cuvânt - cod din C_1 sunt corecte, atunci se pot determina unic celelalte $d_1 - 1$ caractere (deoarece este imposibil ca două cuvinte - cod distincte din C_1 să coincidă pe $n_1 - d_1 + 1$ poziţii, ele diferind pe minim d_1 poziţii; afirmaţia este bazată şi pe faptul că poziţiile caracterelor corecte sunt cunoscute).

Deci, dacă un cuvânt - cod din C_1 are cel mult $d_1 - 1$ simboluri marcate şi se ştie că toate caracterele greşite sunt marcate, cuvintele - cod pot fi decodificate corect.

Cât de mare este un pachet de erori pe care îl poate corecta această schemă ? Să presupunem că C_2 a fost transpus la adâncimea s, şi că orice cuvânt - cod (atât din C_1 cât şi din C_2) a fost afectat de cel mult un pachet de erori. Dacă apare un pachet de cel mult $s(d_2 - 1)$ erori, atunci el va afecta maxim $d_2 - 1$ simboluri din fiecare cuvânt - cod din C_2. Deci aceste erori vor fi detectate şi conduc la marcarea tuturor caracterelor cuvintelor - cod implicate. Dacă $s \leq d_1 - 1$, atunci orice cuvânt - cod din C_1 are cel mult $d_1 - 1$ simboluri marcate. În ipoteza că nu există decât maxim un pachet de erori care l-a perturbat, caracterele marcate pot fi corectate.

Am arătat astfel:

Teorema 13.3 *Fie o codificare care foloseşte transpunerea încrucişată a* (n_1, k_1) *- codului* C_1 *cu* (n_2, k_2) *- codul* C_2, C_2 *fiind transpus la adâncimea*

s ($s \leq d_1 - 1$). Dacă fiecare cuvânt - cod este afectat de cel mult un pachet standard de erori, atunci toate pachetele standard de erori de lungime maxim $s(d_2 - 1)$ pot fi corectate (d_1, d_2 sunt distanţele minime ale celor două coduri).

Exemplul 13.4 *Fie C_1 şi C_2 codurile binare, definite respectiv de matricile generatoare*

$$G_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}, \qquad G_2 = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Deci, $n_1 = 8, k_1 = 4, d_1 = 4, \quad n_2 = 6, k_2 = 3, d_2 = 3$.

Să codificăm mesajele de informaţie $\mathbf{m_1} = 1000$, $\mathbf{m_2} = 1100$, $\mathbf{m_3} = 1010$ folosind o transpunere încrucişată a lui C_1 cu C_2, C_2 fiind transpus la o adâncime $s = 3 = d_1 - 1$.

Codificarea mesajelor de informţie cu codul A_1 dă:

$$\mathbf{c_1} = \mathbf{m_1}G_1 = 10001110,$$
$$\mathbf{c_2} = \mathbf{m_2}G_1 = 11000011,$$
$$\mathbf{c_3} = \mathbf{m_3}G_1 = 10100101.$$

Transpunerea acestor cuvinte la adâncimea $k_2 = 3$ conduce la mesajul

111 010 001 000 100 101 110 011.

Aceste 8 mesaje de informaţie sunt codificate cu C_2 în:

$$\mathbf{c'_1} = 111000,\ \mathbf{c'_2} = 010101,\ \mathbf{c'_3} = 001011,\ \mathbf{c'_4} = 000000,$$
$$\mathbf{c'_5} = 100110,\ \mathbf{c'_6} = 101101,\ \mathbf{c'_7} = 110011,\ \mathbf{c'_8} = 011110.$$

Ele sunt transpuse apoi la adâncimea $s = 3$ ($\mathbf{c'_7}$ şi $\mathbf{c'_8}$ vor fi transpuse împreună cu următorul cuvânt - cod $\mathbf{c'_9}$, obţinute folosind mesajele de informaţie $\mathbf{m_4}, \mathbf{m_5}, \mathbf{m_6}$). Secvenţa de caractere transmise va începe cu

100 110 101 010 001 011 011 000 001 011 010 001 ...

Conform Teoremei 13.3, folosind C_2 pentru a detecta $d_2 - 1 = 2$ erori, şi apoi C_1 pentru a corecta toate caracterele marcate, se pot corecta în ansamblu toate pachetele de cel mult $s(d_2 - 1) = (d_1 - 1)(d_2 - 1) = 6$ erori consecutive.

De exemplu, să presupunem că primii 6 biţi au fost transmişi incorect, deci s-a primit mesajul 011 001 101 010 001 011.... *Eliminând efectul transpunerii la adâncimea $s = 3$, se ajunge la cuvintele*

001000 100101 111011

(care comparate respectiv cu $\mathbf{c'_1}, \mathbf{c'_2}, \mathbf{c'_3}$ au erori fiecare pe primele două poziţii). Deoarece sindromurile sunt nenule, C_2 va detecta erori în toate cele trei cuvinte, deci toţi cei 18 biţi sunt marcaţi (îi vom înlocui cu $*$).

Presupunând că nu mai sunt alte erori, după un procedeu similar se obţin cuvintele $\mathbf{c}'_4, \ldots, \mathbf{c}'_8$, *fără caractere marcate.*

Eliminând acum efectul transpunerii la adâncimea $s = k_2 = 3$, *se obţin cuvintele*

$$\mathbf{c_1} = ***01110, \quad \mathbf{c_2} = ***00011, \quad \mathbf{c_3} = **100101.$$

Există o modalitate unică de a ajunge la cuvinte - cod din C_1 *prin înlocuirea simbolului* $*$ *cu caractere* 0 *sau* 1.

13.1.3 Aplicaţie la înregistrarea CD-urilor

Una din cele mai frecvente utilizări a metodei de transpunere este şi modalitatea de înregistrare a muzicii pe compact discuri. Fiecare compact disc conţine o urmă spiralată în care au fost săpate unele şanţuri. O rază laser care "citeşte" urma, determină modificările de adâncime detectând pur şi simplu schimbările de intensitate ale luminii reflectate de compact disc. În acest mod se obţine un şir de caractere binare, fiecare modificare de adâncime corespunzând lui 1 iar absenţa unei modificări - lui 0.

De exemplu

0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 1 0

La înregistrare, timpul este împărţit în *tacţi*; o secundă conţine 44100 tacţi. Fiecărui tact îi corespunde o secvenţă binară de lungime 16; în acest fel amplitudinea sunetului se va întinde pe o scală cu 2^{16} valori. Înregistrarea stereo va necesita câte două măsurări de amplitudine la fiecare tact.

Pentru cei 16 biţi se folosesc două elemente din $GF(2^8)$ numite *octeţi*. Deci - într-o înregistrare stereo - la fiecare tact sunt realizaţi patru octeţi $a_{4t}, a_{4t+1}, a_{4t+2}, a_{4t+3}$. Cei 24 octeţi din şase tacţi consecutivi sunt grupaţi formând un mesaj de informaţie $\mathbf{a_t} = a_{24t}a_{24t+1}\ldots a_{24t+23}$. Să considerăm un cod Reed - Solomon C cu $n = 2^8$ şi $d = 5$; construim codul scurt $C_1 = C(227)$ obţinut din toate cuvintele - cod din C care au zero pe ultimele 227 poziţii, zero-uri care se ignoră. C_1 are $n_1 = 28$, $k_1 = 24$, $d_1 = 5$ (Teorema 9.3 Capitolul 9). În acest cod, $\mathbf{a_t}$ se codifică în $\mathbf{w_t}$.

Cuvintele - cod din C_1 astfel obţinute sunt aşezate într-un 4 - cadru de transpunere întârziată. Deoarece octeţii cuvântului - cod $\mathbf{w_t}$ apar în acest cadru pe poziţiile pe liniile $1, 2, \ldots, 28$ şi pe coloanele $t, t+4, \ldots, t+108$,

vom renota $\mathbf{w} = w_{1,t}w_{2,t+4}\ldots w_{28,t+108}$. Deci, în general $w_{i,j}$ este al i - lea octet din cuvântul - cod $\mathbf{w_{j-4(i-1)}}$.

Fiecare coloană t din cadru, formată din octeţii $w_{1,t}w_{2,t}\ldots, w_{28,t}$, este considerată un mesaj de informaţie peste $GF(2^8)$ şi va fi codificată din nou, folosind codul Reed - Solomon scurtat $C_2 = C(223)$ (care are - conform Teoremei 9.3 - $n_2 = 32$, $k_2 = 28$, $d = 5$).

La fiecare cuvânt - cod din C_2 se adaugă la sfârşit un al 33 - lea bit de control.

Din considerente fizice, nu este posibil ca în urma de pe compact disc să fie modificări de adâncime prea apropiate. Este deci de dorit ca în reprezentarea binară a fiecărui cuvânt - cod, între orice doi 1 consecutivi să apară cel puţin două şi cel mult zece 0 - uri. Din nefericire, codurile Reed - Solomon nu au această proprietate, aşa că se foloseşte o supra - codificare în modul următor. Există 267 cuvinte binare de lungime 14 care au verifică condiţia cerută. Cei 256 octeţi din $GF(2^8)$ sunt codificaţi (pe baza unei tabele) în astfel de cuvinte de lungime 14. Procesul este numit *modularea opt - paisprezece* (EFM : eight - fourteen modulation). Apoi, pentru a asigura aceeaşi proprietate şi între cuvintele - cod, se mai adaugă la sfârşit încă trei biţi. Rezultă astfel că reprezentarea binară a fiecărui cuvânt - cod are lungimea $33 \cdot 17 = 561$.

Din motive de sincronizare, la sfârşitul fiecărui cuvânt - cod se adaugă încă 27 biţi care au de asemenea proprietatea menţionată. Deci, în total, informaţia audio din 6 tacţi consecutivi este stocată într-o secvenţă de $24 \cdot 8 = 196$ biţi care - la sfârşitul codificării - apare pe compact disc ca un cuvânt - binar de lungime 588.

La decodificare, C_2 este utilizat pentru corectarea unei erori; dacă se detectează mai multe erori într-un cuvânt, toţi octeţii acelui cuvânt sunt marcaţi (în conformitate cu regulile de transpunere încrucişată) . Un octet este corect dacă şi numai dacă nu este marcat. În final, C_1 este folosit pentru corectarea a maxim 4 ştergeri, considerând drept toţi octeţii marcaţi. Un studiu detaliat ([7]) arată că prin acest cod se poate corecta orice deteriorare a unui canal, având cel mult 2,5 mm lungime.

Am prezentat aici doar principalele aspecte ale procesului de codificare pe un compact disc. În practică mai apar şi alte operaţii de transpunere. De exemplu, octeţii aflaţi pe poziţiile impare în cuvintele - cod din C_2 sunt deplasaţi cu $n_2 = 32$ poziţii, deci sunt amestecaţi cu octeţii de pe poziţiile pare din următorul cuvânt - cod. Aceasta va permite ca şi C_2 să poată corecta o eroare independentă.

13.2 Coduri - tablou

13.2.1 Definirea codurilor tablou

În secţiunea 3.8 (Capitolul 3) au fost definite codurile - produs. Deşi construcţia respectivă este cunoscută din anii '60, dezvoltarea acestei clase de coduri s-a realizat abia după trei decenii, odată cu introducerea pe piaţă a telefoniei mobile. Practic, orice cod - bloc poate fi generat finit recursiv, folosind o variantă de coduri - produs.

Definiţia 13.1 *Un cod - tablou este un* $(n_1 \cdot n_2, k_1 \cdot k_2)$ *- cod - produs, în care* $n_1 = k_1 + 1$, $n_2 = k_2 + 1$ *(deci singurul simbol de control în cele două coduri componente este cel de control al parităţii).*

Conform Definiţiei 13.1, un cod - tablou este echivalent cu codul generat de produsul Kronecker a două matrici de forma

$$G_{p,p+1} = \begin{pmatrix} 1 & 0 & 0 & & 0 & 1 \\ 0 & 1 & 0 & \dots & 0 & 1 \\ & & & \vdots & & \\ 0 & 0 & 0 & & 1 & 1 \end{pmatrix} \tag{1}$$

unde $p = k_1$ respectiv $p = k_2$.

Un astfel de $(n_1 \cdot n_2, k_1 \cdot k_2)$ - cod are distanţa $d = d_1 \cdot d_2 = 2 \cdot 2 = 4$, deci corectează o eroare şi detectează două erori. Chiar dacă în scrierea liniară a cuvintelor - cod ultimul simbol de control se poate elimina (el este de fapt suma modulo 2 a biţilor de control de pe ultima linie şi coloană) distanţa se reduce la 3, deci capacitatea de corectare a unei erori rămâne.

Exemplul 13.5 *Fie un* $(3, 2)$ *- cod binar sistematic, generat de matricea*

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

Codul - tablou $(3, 2) \cdot (3, 2) = (9, 4)$ *va avea ca matrice generatoare*

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

Dacă $\mathbf{a} = 1001$ *este un mesaj de informaţie, el se codifică în*

$$\mathbf{x} = \mathbf{a}G' = 101011110,$$

sau folosind forma matricială:

$$\mathbf{x} = \begin{matrix} 1 & 0 & \underline{1} \\ 0 & 1 & \underline{1} \\ \underline{1} & \underline{1} & \underline{0} \end{matrix} = 10\underline{1}011\underline{11}\underline{10},$$

caracterele de control fiind subliniate, iar cuvântul - cod este transmis linie cu linie.

13.2.2 Structura "reţea" a codurilor tablou

În cadrul codurilor convoluţionale s-a definit o structură de reţea pentru facilitarea codificării, decodificării, detectării şi corectării de erori (secţiunea 12.8, Capitolul 12). Această structură se poate extinde cu mici modificări şi la codurile liniare.

Fie C un (n, k) - cod liniar care se poate reprezenta printr-un cod - tablou. Lui i se asociază un graf orientat cu arce marcate. Vârfurile grafului sunt partiţionate în coloane $V_0, V_1, \ldots, V_{N_c-1}$ $(N_c \leq n+1)$. O coloană V_i conţine mulţimea stărilor (neprecizate momentan) în care se poate ajunge la momentul i. Fiecare arc U pleacă dintr-o stare $S_j \in V_i$ şi ajunge la o stare $S_m \in V_{i+1}$ $(i \geq 0)$; în acest caz, S_m este *succesorul* lui S_j, iar S_j este *predecesorul* lui S_m. Arcul este marcat cu o pereche $\delta(U)/\lambda(U)$, unde $\delta(U) \in Z_q^*$ este un mesaj de informaţie, iar $\lambda(U) \in Z_q^*$ este mesajul - cod corespunzător (de obicei ele se reduc la câte un singur caracter).

Proprietăţi:

(a) $V_0 = \{S_0\}$, $V_{N_c-1} = \{S_{N_c-1}\}$; deci, orice reţea pleacă dintr-o stare iniţială unică S_0 (*rădăcina*) şi ajunge de asemenea într-o stare unică S_{N_c-1} (*ţinta*);

(b) Orice stare este accesibilă din rădăcină prin cel puţin un drum;

(c) Din orice stare se poate ajunge la ţintă prin cel puţin un drum.

Numim *drum prin reţea* un drum de la rădăcină la ţintă. Printr-o reţea se pot cataloga toate cuvintele unui cod liniar reprezentabil sub formă de cod - tablou. Orice cuvânt - cod corespunde unui drum unic prin reţea. Deci o reţea va conţine q^k drumuri distincte. Fiecare astfel de drum are două marcaje:

- concatenarea marcajelor de informaţie $\delta(U)$ ale fiecărui arc U al

drumului; această secvenţă formează mesajul de informaţie (care trebuie codificată);

- concatenarea marcajelor $\lambda(U)$ ale fiecărui arc U al drumului; ele formează cuvântul - cod care codifică secvenţa de informaţie.

În acest fel, operaţia de codificare sau decodificare (fără corectare de erori) se reduce pentru fiecare mesaj de informaţie (cod) la determinarea acelui drum prin reţea, marcat de mesajul respectiv.

Să considerăm numai $(n_1 \cdot n_2, k_1 \cdot k_2)$ - coduri tablou binare (generalizarea la cazul nebinar se face fără probleme deosebite). Pentru aceste coduri se pot construi evident mai multe structuri de reţea; vom considera aici numai reţele cu un număr minim de stări pe fiecare coloană. Construcţia unei astfel de reţele va necesita

$$N_s = 2^{min\{k_1, k_2\}}$$

stări (reamintim, aici $k_i = n_i - 1$. $i = 1, 2$). Procedura de generare ([8]) a reţelei este:

1. Fie $n_1 = min\{n_1, n_2\}$ (pentru cazul invers se procedează similar);

2. Se determină numărul N_s de stări şi numărul N_c de coloane prin relaţiile

$$N_s = 2^{n_1-1}, \qquad N_c = n_2 + 1;$$

3. Pentru fiecare coloană V_p $(0 \leq p \leq N_c - 1)$ se notează elementele (stările) sale prin $(A)_p = (a_1 a_2 \ldots a_{k_1})_p$, unde $a_i \in Z_2$ $(1 \leq i \leq k_1)$;

4. Rădăcina reţelei este $(00\ldots0)_0$, iar ţinta este $(00\ldots0)_{N_c-1}$;

5. Pentru orice pereche de stări $(A, B) \in V_p \times V_{p+1}$ $(0 \leq p < N_c - 2)$, se marchează arcul corespunzător cu $\delta_p(A,B)/\lambda_p(A,B)$ definite $\delta_p(A,B) = (A)_p + (B)_{p+1}$, $\lambda_p(A,B) = \delta_p(A,B)c_1$, unde c_1 este bitul de paritate al secvenţei $\delta_p(A,B)$, iar suma se realizează în Z_2 (operaţia XOR);

6. Arcele care leagă stările $A \in V_{N_c-2}$ de ţintă au $\delta_{N_c-2}(A,B) = \epsilon, \lambda_{N_c-2}(A,B) = (A)_{N_c-2}c_1$ unde c_1 este bitul de paritate al secvenţei $(A)_{N_c-2}$ care defineşte starea A.

Există $2^{k_1 \cdot k_2}$ drumuri distincte prin reţea, fiecare corespunzând unui cuvânt - cod unic. Procedura poate fi extinsă uşor la codurile produs cu mai multe simboluri de control.

Exemplul 13.6 *Să construim reţeaua asociată codului tablou* $(3,2) \cdot (3,2)$ *(deci* $n_1 = n_2 = 3$, $k_1 = k_2 = 2$, $d = 4$*).*

Ea are $N_s = 2^{n_1-1} = 2^{3-1} = 4$ *stări distincte şi* $N_c = n_2 + 1 = 3 + 1 = 4$ *coloane.*

Fiecare stare este de forma $(a_1 a_2)_p$ *cu* $p = 0, 1, 2, 3$, *şi* $a_1, a_2 \in Z_2$.

$(00)_0$ *şi* $(00)_3$ *sunt fi rădăcina respectiv ţinta reţelei.*

Marcajele arcelor sunt de forma $\delta_p(A,B)/\lambda_p(A,B)$, *obţinute prin toate combinaţiile posibile ale lui* $(a_1 a_2)_p$ *şi* $(a_1 a_2)_{p+1}$.

Astfel, pentru $p = 0$ *avem:*

$\delta_0(00,00) = (00)_0 + (00)_1 = 00$, $\quad \delta_0(00,01) = (00)_0 + (01)_1 = 01$,

$\delta_0(00,01) = (00)_0 + (10)_1 = 10$, $\quad \delta_0(00,11) = (00)_0 + (11)_1 = 11$,

iar fiecare arc va fi marcat cu o pereche de forma

$$\delta_0/\lambda_0 = a_1a_2/a_1a_2c_1,$$

unde $c_1 = a_1 + a_2$.

Pe nivelul $p = 1$, *arcele sunt marcate prin perechi din tabelul următor:*

$\delta_1(A,B)$	00	01	10	11
00	00/000	01/011	10/101	11/110
01	01/011	00/000	11/110	10/101
10	10/101	11/110	00/000	01/011
11	11/110	10/101	01/011	00/000

La ultimul nivel, pentru $p = N_c - 2 = 2$ *vom avea* $\delta_2(A,00) = \epsilon$ *şi* $\lambda_2(00,00) = 000$, $\lambda_2(01,00) = 011$, $\lambda_2(10,00) = 101$, $\lambda_2(11,00) = 110$ *(pe ultimul arc sunt numai simboluri de control).*

Reţeaua se poate reprezenta grafic ca în Figura 13.1:

Exemplul 13.7 *În mod similar se construiesc codurile tablou obţinute prin produsul a două* $(3,2)\cdot(n_2,k_2)$ *- coduri bloc. Toate au acelaşi număr de stări* $N_s = 2^2 = 4$, *doar adâncimea (numărul de coloane) variind în funcţie de* n_2 *(*$N_c = n_2 + 1$*). Reprezentarea grafică a reţelei este dată în Figura 13.2, marcajul nodurilor fiind realizat similar cu cel din Exemplul 13.6.*

13.2.3 Decodificarea codurilor tablou

Operaţia de codificare se realizează foarte simplu: fiecare mesaj de informaţie de $k_1 \cdot k_2$ caractere, se scrie ca o secvenţă de $k_2 + 1$ grupuri, primele k_2 grupuri având câte k_1 simboluri, iar ultimul ϵ. Această secvenţă marchează prin reţea un drum unic $(00\ldots0) - A_1 - \ldots - A_{N_c-2} - (00\ldots0)$ de la rădăcina $00\ldots0$ la ţinta $00\ldots0$. Ceea ce se obţine prin concatenarea ieşirilor

$$\lambda_0(00\ldots0, A_1)\lambda_1(A_1, A_2)\ldots\lambda_{N_c-2}(A_{N_c-2}, 00\ldots0)$$

va fi un cuvânt - cod al $(n_1 \cdot n_2, k_1 \cdot k_2)$ - codului tablou reprezentat de reţea.

Pentru decodificare se poate folosi o variantă a algoritmului Viterbi, cu mici modificări, datorate trecerii de la codurile convoluţionale la coduri - bloc (în particular, modului de definire a stărilor).

Figura 13.1:

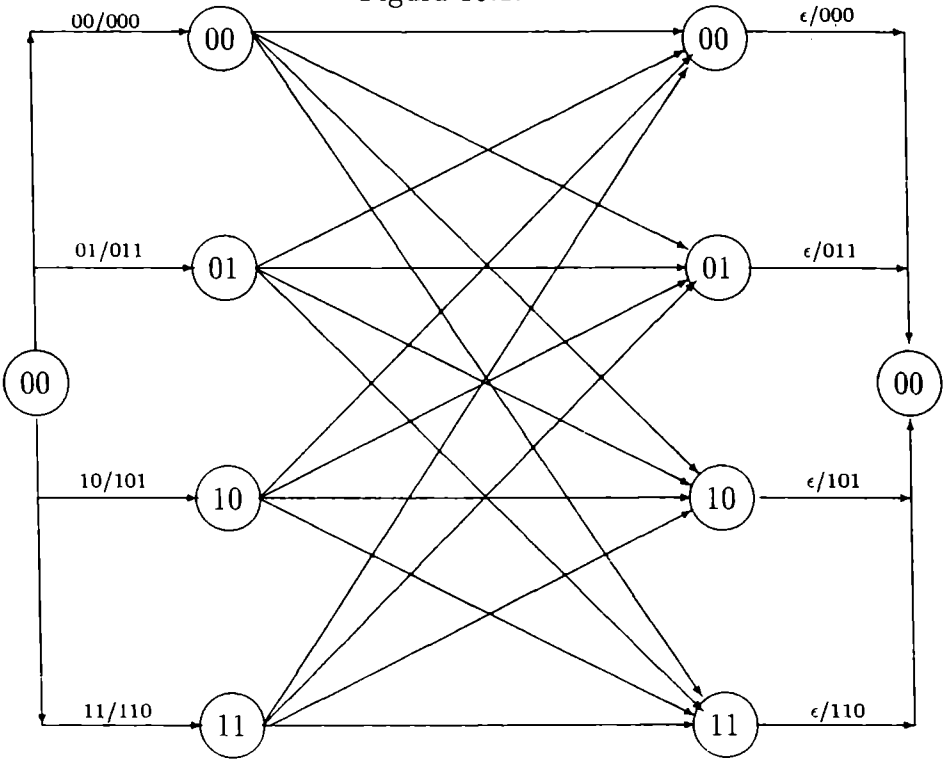
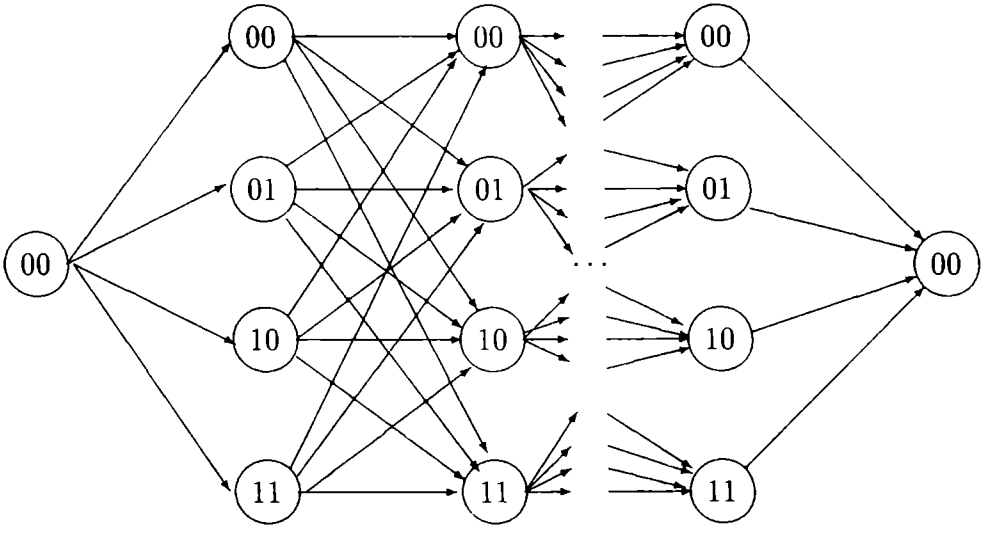


Figura 13.2:



1. Cuvântul recepţionat este împărţit în n_2 subcuvinte, fiecare având n_1 valori digitizate (vezi Exemplul 13.8) ale alfabetului - cod.

2. La fiecare nivel p, pentru fiecare arc (A_p, B_{p+1}) se calculează *metrica arcului*, care este distanţa euclidiană dintre al $p+1$-lea subcuvânt recepţionat şi $\lambda_p(A_p, B_{p+1})$. Distanţa euclidiană dintre $\mathbf{x} = x_1x_2\ldots x_n$ şi $\mathbf{y} = y_1y_2\ldots y_n$ este
$d_E(\mathbf{x},\mathbf{y}) = (x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2$.

3. Pentru fiecare nod B al reţelei se determină *metrica nodului* B, astfel:

   (a) Metrica nodului rădăcină este 0;

   (b) Pentru un nod $B_{p+1} \in V_{p+1}$ $(p \geq 0)$, metrica lui B este cea mai mică valoare a sumei dintre metrica unui nod $A_p \in V_p$ şi $d_E((A)_p, (B)_{p+1})$, minimul fiind luat după toate nodurile din V_p.

4. Arcul selectat în calculul metricii nodului se introduce într-un *drum activ*, iar toate celelalte arce de pe nivelul respectiv sunt eliminate.

5. După ce s-a calculat metrica nodului ţintă, se consideră drumul activ corespunzător şi decodificarea se face prin concatenarea sub-mesajelor de informaţie care marchează arcele acestui drum.

Exemplul 13.8 *Să reluăm codul - tablou definit în Exemplul 13.6. Presupunem că s-a transmis cuvântul cod* $\mathbf{a} = 011101110$ *şi s-a recepţionat mesajul digitizat*

$$\mathbf{b}' = 0{,}2\ \ 0{,}4\ \ 0{,}4\ \ 0{,}3\ \ 0{,}1\ \ 0{,}9\ \ 1{,}0\ \ 0{,}9\ \ 0{,}6.$$

Presupunând că orice simbol digitizat din intervalul $[0{,}0; 0{,}5)$ *reprezintă caracterul binar* 0, *iar* $[0.5; 1{,}0]$ *reprezintă* 1, *putem considera că s-a primit secvenţa* $\mathbf{b} = 000001111$ *deci au apărut* 4 *erori. Într-o decodificare obişnuită a codurilor liniare, deoarece* $d = 4$, *erorile nu pot fi corectate. Folosind structura de reţea însă, acest lucru este realizabil. Să detaliem paşii utilizaţi în decodificarea cuvântului* $\mathbf{b}'$.

– *Metrica nodului rădăcină* $(00)_0$ *este* 0.

- *Prima subsecvenţă de* $n_1 = 3$ *caractere primite este* 0,2 0,4 0,4.
- *Se calculează metricile celor patru arce care pleacă din rădăcină:*

$$d_E((00)_0,(00)_1) = (0,2-0,0)^2+(0,4-0,0)^2+(0,4-0,0)^2 = 0,36;$$
$$d_E((00)_0,(01)_1) = (0,2-0,0)^2+(0,4-1,0)^2+(0,4-1,0)^2 = 0,76;$$
$$d_E((00)_0,(10)_1) = (0,2-1,0)^2+(0,4-0,0)^2+(0,4-1,0)^2 = 1,16;$$
$$d_E((00)_0,(11)_1) = (0,2-1,0)^2+(0,4-1,0)^2+(0,4-0,0)^2 = 1,16.$$

- *Metricile celor patru noduri de pe nivelul* 1 *sunt chiar aceste valori.*

Următorul submesaj primit este 0,3 0,1 0,9. *Pentru nodurile de pe nivelul* 2, *valorile calculate ale metricilor (metrica nodului predecesor plus metrica arcului) sunt:*

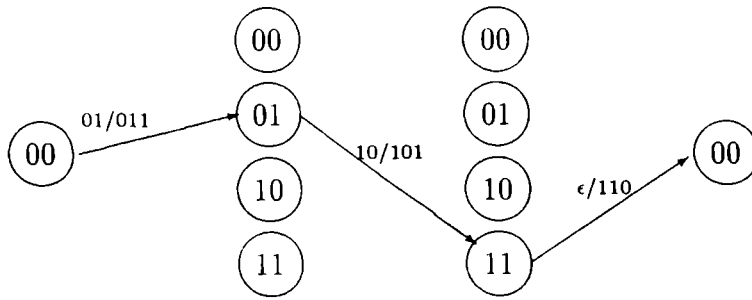
	$(00)_2$	$(01)_2$	$(10)_2$	$(11)_2$
$(00)_1$	**1,27**	**1,27**	**0,87**	2,42
$(01)_1$	1,67	1,67	2,87	**1,37**
$(10)_1$	1,67	3,27	2,07	2,07
$(11)_1$	3,27	1,67	2,07	2,07

- *Minimul pe fiecare coloană ale acestor valori (scris îngroşat) reprezintă metrica nodului respectiv. Deci nodurile* $(00)_2$ *şi* $(01)_2$ *au metrica* 1,27, $(10)_2$ *are metrica* 0,87, *iar* $(11)_2$ *are metrica* 1,37.

Ultima subsecvenţă de trei caractere digitizate este 1,0 0,9 0,6. *Valo-rile calculate ale metricilor (metrica nodului predecesor plus metrica arcului) drumurilor care ajung în nodul ţintă sunt:*

	$(00)_3$
$(00)_2$	3,44
$(01)_2$	2,44
$(10)_2$	1,84
$(11)_2$	**1,74**

Deci metrica nodului ţintă este 1,74. *Refăcând acum traiectoria în sens invers, se obţine drumul activ* $(00)_0-(01)_1-(11)_2-(00)_3$, *reprezentat în figura*



Prin concatenarea marcajelor arcelor de pe acest drum, rezultă cuvântul - cod 011101110 *(care coincide cu mesajul transmis* **a***) şi decodificarea lui - mesajul de informaţie* 0110.

13.2.4 Coduri tablou generalizate (GAC)

Definiţia codurilor - tablou se poate extinde pentru a cuprinde şi alte clase de coduri liniare (Hamming, Reed - Muller) sau ciclice (BCH, Reed - Solomon); ceea ce se obţine este un *cod - tablou generalizat* (GAC - Generalized Array Codes).

Un GAC este o sumă binară de două sau trei coduri - tablou, completate eventual cu linii nule. Procedura de construcţie a unui astfel de (n, k) - cod C este:

- Fie C_1 un cod - produs prin înmulţirea Kronecker $G_{A_1} \times G_{A_2}$, unde:
  - G_{A_1} este o matrice $k_1 \times (k_1 + 1)$ de tipul (1);
  - $G_{A_2} = (I_{k_2} | J_{k_2, n_2 - k_2})$ unde I_{k_2} este matricea unitate iar J este o matrice cu toate elementele 1;
  - $n = (k_1 + 1) \cdot n_2$, $k_A = k_1 \cdot k_2$.
- Fie C_2 un cod - produs prin înmulţirea Kronecker $G_{B_1} \times G_{B_2}$, unde:
  - G_{B_1} este o matrice $1 \times n_{B_1}$ (vector linie) cu toate elementele 1;
  - $G_{B_2} = (G_3 | G_4)$ este o matrice $k_B \times n_{B_2}$ unde G_3 este o matrice cu coloane 0 sau 1, iar G_4 este o matrice de tipul (1) cu k_B linii. Coloanele lui G_{B_2} pot fi eventual permutate.
  - $n = n_{B_1} \cdot n_{B_2}$.
- Fie C_3 un cod - produs prin înmulţirea Kronecker $G_{C_1} \times G_{C_2}$, unde:
  - $G_{C_1} = (00 \ldots 01)$ are dimensiunea $1 \times n_{C_1}$;
  - $G_{C_2} = (11 \ldots 1)$ are dimensiunea $1 \times n_{C_2}$;
  - $n = n_{C_1} \cdot n_{C_2}$.
- Codul C este $C_1 + C_2$ sau $C_1 + C_2 + C_3$ unde suma se efectuează în Z_2; el are lungimea n, $k = k_A + k_B$ respectiv $k = k_A + k_B + 1$ simboluri de informaţie şi este generat de matricea

$$\begin{pmatrix} G_{A_1} \times G_{A_2} \\ G_{B_1} \times G_{B_2} \end{pmatrix} \qquad \text{sau} \qquad \begin{pmatrix} G_{A_1} \times G_{A_2} \\ G_{B_1} \times G_{B_2} \\ G_{C_1} \times G_{C_2} \end{pmatrix}$$

În exemplele următoare vom prezenta sub formă de coduri GAC unele din cele mai cunoscute coduri - bloc (liniare sau ciclice).

Exemplul 13.9 *Să considerăm*

$$G_{A_1} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \quad G_{A_2} = (1\ 1),$$
$$G_{B_1} = (1\ 1\ 1\ 1), \quad G_{B_2} = (0\ 1).$$

Dimensiunile codului sunt $n = 4 \cdot 2 = 8$, $k = 3 + 1 = 4$. *Matricea generatoare va fi*

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

Aceasta corespunde unui cod Reed - Muller $\mathcal{RM}(1,3)$. *Dacă se ignoră ultima coloană (de control), se obţine un* $(7,4)$ *- cod Hamming binar.*

Aranjarea sub formă de tablou a cuvintelor - cod din cele două coduri - produs componente este următoarea:

Fie $\mathbf{a} = a_1a_2a_3a_4$ *un mesaj de informaţie. Atunci cuvântul - cod* $\mathbf{x}$ *este dat prin:*

$$\mathbf{x_1} = \begin{matrix} a_1 & p_1 \\ a_2 & p_2 \\ a_3 & p_3 \\ p_4 & p_5 \end{matrix}, \qquad \mathbf{x_2} = \begin{matrix} 0 & a_4 \\ 0 & a_4 \\ 0 & a_4 \\ 0 & a_4 \end{matrix},$$

$$\mathbf{x} = \mathbf{x_1} + \mathbf{x_2} = \begin{matrix} a_1 & p_1 + a_4 \\ a_2 & p_2 + a_4 \\ a_3 & p_3 + a_4 \\ p_4 & p_4 + a_4 \end{matrix} = (a_1, a_4 + p_1, a_2, p_2 + a_4, a_3, p_3 + a_4, p_4, p_4 + a_4),$$

unde $p_i = a_i$ $(1 \leq i \leq 3)$ *şi* $p_4 = a_1 + a_2 + a_3$ *sunt simboluri de control.*

Exemplul 13.10 *Codul* $\mathcal{RM}(1,4)$ *poate fi generat cu un cod GAC astfel:*

$$G_{A_1} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \qquad G_{A_2} = (1\ 1\ 1\ 1),$$

$$G_{B_1} = (1\ 1\ 1\ 1), \qquad G_{B_2} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Deci $n = 4 \cdot 4$, $k = 3 + 2 = 5$ *şi*

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

Eliminând ultimul simbol de control (ultima coloană din matricea G*) se ajunge la un* $(15, 5)$ *- cod binar cu* $d = 7$*, care este un cod* BCH*.*

Fie $\mathbf{a} = a_1a_2a_3a_4a_5$ *un mesaj de informaţie. Reprezentarea tabelară a cuvintelor - cod este:*

$$\mathbf{x_1} = \begin{matrix} a_1 & p_1 & p_1 & p_1 \\ a_2 & p_2 & p_2 & p_2 \\ a_3 & p_3 & p_3 & p_3 \\ p_4 & p_4 & p_4 & p_4 \end{matrix}, \qquad \mathbf{x_2} = \begin{matrix} 0 & a_4 & a_5 & a_4 + a_5 \\ 0 & a_4 & a_5 & a_4 + a_5 \\ 0 & a_4 & a_5 & a_4 + a_5 \\ 0 & a_4 & a_5 & a_4 + a_5 \end{matrix},$$

$$\mathbf{x} = \mathbf{x_1} + \mathbf{x_2} = \begin{matrix} a_1 & p_1 + a_4 & p_1 + a_5 & p_1 + a_4 + a_5 \\ a_2 & p_2 + a_4 & p_2 + a_5 & p_2 + a_4 + a_5 \\ a_3 & p_3 + a_4 & p_3 + a_5 & p_3 + a_4 + a_5 \\ p_4 & p_4 + a_4 & p_4 + a_5 & p_4 + a_4 + a_5 \end{matrix},$$

unde $p_i = a_i$ $(1 \leq i \leq 3)$, $p_4 = a_1 + a_2 + a_3$ *sunt simboluri de control.*

Exemplul 13.11 *Să plecăm de la următoarele matrici de bază:*

$$G_{A_1} = G_{A_2} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix},$$

$$G_{B_1} = G_{C_2} = (1\ 1\ 1\ 1), \quad G_{B_2} = G_{C_1} = (0\ 0\ 0\ 1).$$

Se obţine un cod $C = C_1 + C_2 + C_3$ *cu* $n = 4 \cdot 4 = 16$, $k = 3 \cdot 3 + 1 + 1 = 11$ *şi matrice generatoare*

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

El are $d = 4$ şi este echivalent cu un cod $\mathcal{RM}(2,4)$. Dacă se elimină ultimul simbol de control, se obţine un $(15,11)$ - cod Hamming binar.

Fie $\mathbf{a} = a_1a_2\dots a_{11}$ un mesaj de informaţie. Cuvântul - cod $\mathbf{x}$ este format din:

$$\mathbf{x_1} = \begin{matrix} a_1 & a_2 & a_3 & p_1 \\ a_4 & a_5 & a_6 & p_2 \\ a_7 & a_8 & a_9 & p_3 \\ p_4 & p_5 & p_6 & p_7 \end{matrix}, \quad \mathbf{x_2} = \begin{matrix} 0 & 0 & 0 & a_{10} \\ 0 & 0 & 0 & a_{10} \\ 0 & 0 & 0 & a_{10} \\ 0 & 0 & 0 & a_{10} \end{matrix}, \quad \mathbf{x_3} = \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ a_{11} & a_{11} & a_{11} & a_{11} \end{matrix},$$

$$\mathbf{x} = \mathbf{x_1} + \mathbf{x_2} + \mathbf{x_3} = \begin{matrix} a_1 & a_2 & a_3 & p_1 + a_{10} \\ a_4 & a_5 & a_6 & p_2 + a_{10} \\ a_7 & a_8 & a_9 & p_3 + a_{10} \\ p_4 + a_{11} & p_5 + a_{11} & p_6 + a_{11} & p_7 + a_{10} + a_{11} \end{matrix}$$

Exemplul 13.12 *Pentru $(24,12)$ - codul Golay binar extins se poate da următoarea reprezentare GAC:*

$$G_{A_1} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad G_{A_2} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

$$G_{B_1} = (1\ 1\ 1), \quad G_{B_2} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$G_{C_1} = (0\ 0\ 1) \quad G_{C_2} = (1\ 1\ 1\ 1\ 1\ 1\ 1\ 1)$.

Fie $\mathbf{a} = a_1a_2\dots a_{12}$ un mesaj de informaţie.

Pentru obţinerea cuvântului - cod, avem:

$$\mathbf{x_1} = \begin{matrix} a_1 & a_2 & a_3 & a_4 & p_1 & p_2 & p_3 & p_4 \\ a_5 & a_6 & a_7 & a_8 & p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} & p_{13} & p_{14} & p_{15} & p_{16} \end{matrix},$$

$$\mathbf{x_2} = \begin{matrix} c_1 & a_9 & c_2 & a_{10} & c_3 & c_4 & a_{11} & c_5 \\ c_1 & a_9 & c_2 & a_{10} & c_3 & c_4 & a_{11} & c_5 \\ c_1 & a_9 & c_2 & a_{10} & c_3 & c_4 & a_{11} & c_5 \end{matrix},$$

$$\mathbf{x_3} = \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{12} & a_{12} & a_{12} & a_{12} & a_{12} & a_{12} & a_{12} & a_{12} \end{matrix},$$

deci

$$\mathbf{x} = \mathbf{x_1} + \mathbf{x_2} + \mathbf{x_3} =$$

$$\begin{matrix} a_1 + c_1 & a_2 + a_9 & a_{11} + p_3 & c_5 + p_4 \\ a_5 + c_1 & a_6 + a_9 & a_{11} + p_7 & c_5 + p_8 \\ a_{12} + p_9 + c_1 & a_{12} + p_{10} + a_9 & a_{12} + a_{11} + p_{15} & a_{12} + p_{16} + c_5 \end{matrix}$$

De remarcat maniera mult mai simplă de codificare, ea înlocuind înmulțirile cu adunări în Z_q (pentru simbolurile de control).

13.2.5 Decodificarea codurilor *GAC*

Pentru decodificare se poate folosi tot algoritmul lui Viterbi, aplicat unei structuri de rețea sub care se pot reprezenta aceste coduri. Structura de rețea se construiește similar cu cea a codurilor - tablou, cu unele mici modificări datorate definiției cuvintelor - cod, ca sume de două sau trei alte secvențe. Algoritmul este dat în Figura 13.3

Vom considera reprezentarea cuvintelor unui cod *GAC* sub formă de tablou cu n_1 linii și n_2 coloane. Fie k_i numărul de simboluri noi de informație care pot apare pe linia i a cuvintelor - cod și $t = \max\limits_{1 \le i \le n_1} \{k_i\}$.

Figura 13.3:

1. Numărul de coloane (adâncimea rețelei) este $N_c = n_1 + 1$, iar numărul de stări distincte este $N_s = 2^t$.
2. Fiecare stare de pe coloana p se notează $(a_1 a_2 \dots a_t)_p$ unde $a_i \in Z_2$ $(1 \le i \le t)$.
3. Rădăcina rețelei este $(00\dots0)_0$ iar ținta $(00\dots0)_{n_1}$.
4. Fiecare arc (A_p, B_{p+1}) de pe nivelul p este marcat cu o pereche $\delta_p(A,B)/\lambda_p(A,B)$ unde $\delta_p(A,B)$ este secvența de simboluri noi de informație de pe linia p (eventual ϵ), iar $\lambda_p(A,B)$ este linia p a cuvântului - cod.
5. Dacă A_p și B_{p+1} sunt legate printr-un arc și $|\delta_p(A,B)| = s$, atunci $B_{p+1} = A_p + \delta_p(A,B)0^{t-s}$.
6. Dacă construcția codului *GAC* a folosit și codul C_3, atunci fiecare nod de pe coloana n_2 este legat de țintă prin două arce, marcajul celui de-al doilea arc fiind complementara marcajului primului arc (pe acest nivel $\delta_{n_2}(A, 00\dots0) = \epsilon$).

În această construcţie de reţea sunt $\prod_{i=1}^{n_1} 2^{k_i}$ drumuri distincte, fiecare din ele corespunzând unui cuvânt - cod.

Operaţiile de codificare/decodificare (inclusiv aplicarea algoritmului Viterbi adaptat) sunt identice cu cele definite la codurile - tablou.

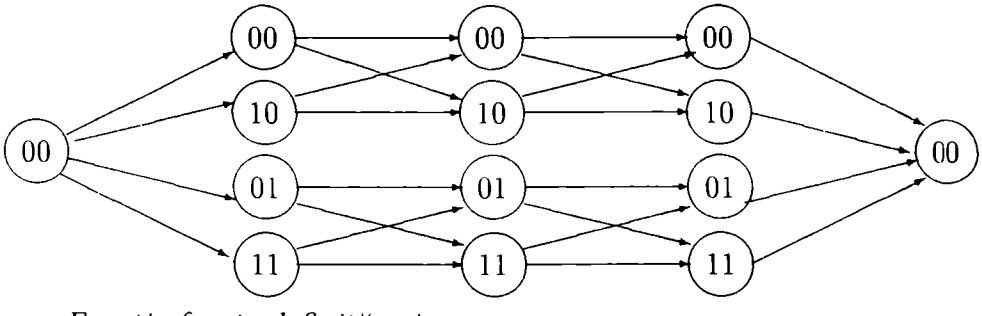
Exemplul 13.13 *Reluăm codul definit în Exemplul 13.9. Pentru el se poate construi o reţea în felul următor:*

Sunt necesare $N_c = n_1 + 1 = 4 + 1 = 5$ *coloane. Pentru numărul de stări, să studiem forma de tablou a cuvintelor cod:*

$$\mathbf{x} = \begin{matrix} a_1 & p_1 + a_4 \\ a_2 & p_2 + a_4 \\ a_3 & p_3 + a_4 \\ p_4 & p_4 + a_4 \end{matrix}$$

În această reprezentare, pe prima linie sunt două simboluri de informaţie (a_1 *şi* a_4)*, pe a doua şi a treia câte unul* (a_2 *respectiv* a_3, a_4 *nefiind simbol nou), iar pe a patra linie, nici unul. Deci numărul de stări al reţelei este* $N_s = 2^2 = 4$*, fiecare stare fiind reprezentată printr-o secvenţă de* $t = 2$ *caractere binare* (00, 01, 10, 11).

Reprezentarea grafică a reţelei este:



Funcţia δ *este definită prin*

$\delta_0(00, B) = a_1a_4, \quad \delta_1(A, B) = a_2, \quad \delta_2(A, B) = a_3, \quad \delta_3(A, 00) = \epsilon.$

Nodurile şi arcele sunt marcate cu valorile date de tabelele următoare:

$\delta_0(A,B)/\lambda_0(A,B)$	00	10	01	11
00	00/00	10/11	01/01	11/10

$\delta_1(A,B)/\lambda_1(A,B)$	00	10	01	11
00	0/00	1/11	–	–
10	1/11	0/00	–	–
01	–	–	0/01	1/10
11	–	–	1/10	0/01

$\delta_2(A,B)/\lambda_2(A,B)$	00	10	01	11
00	0/00	1/11	–	–
10	1/11	0/00	–	–
01	–	–	0/01	1/10
11	–	–	1/10	0/01

$\delta_3(A,B)/\lambda_3(A,B)$	00
00	ϵ/00
10	ϵ/11
01	ϵ/01
11	ϵ/10

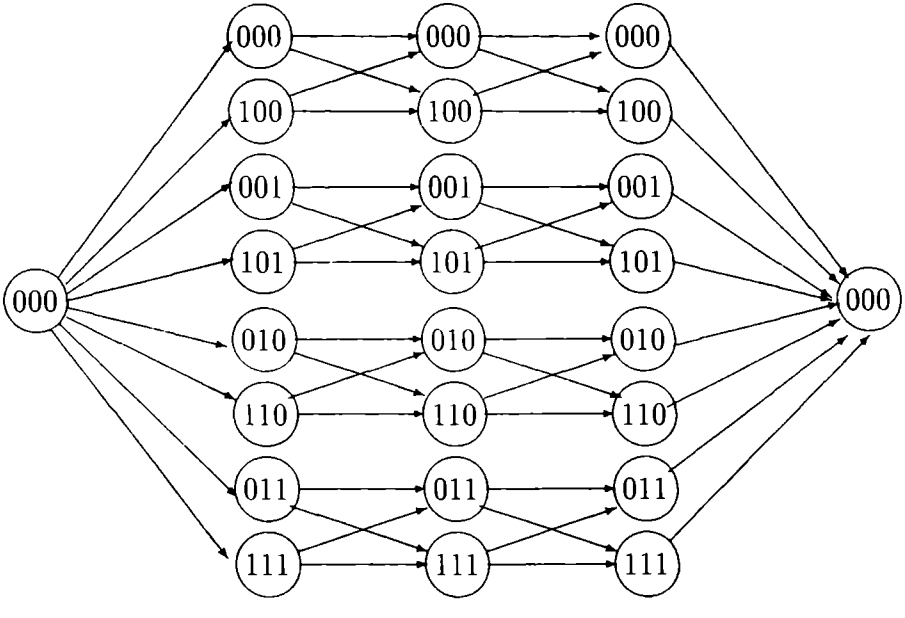
pentru codul $\mathcal{RM}(1,3)$,

$\delta_3(A,B)/\lambda_3(A,B)$	00
00	ϵ/0
10	ϵ/1
01	ϵ/0
11	ϵ/1

pentru $(7,4)$ *- codul Hamming*

Exemplul 13.14 *Codurile* $\mathcal{RM}(1,4)$ *şi* $(15,5)$ *– BCH construite sub formă de GAC în Exemplul 13.10, sunt reprezentate ca reţea conform Figurii 13.4*

Figura 13.4:



Adâncimea reţelei (numărul de coloane) este $N_c = 4+1 = 5$. *Numărul*

maxim de simboluri de informaţie noi se află pe prima linie a cuvintelor - cod (a_1, a_4, a_5); deci reţeaua are $N_s = 2^3 = 8$ stări distincte.

Funcţia δ este definită

$\delta_0(000, B) = a_1a_4a_5$, $\delta_1(A, B) = a_2$, $\delta_2(A, B) = a_3$, $\delta_3(A, 000) = \epsilon$.

iar funcţia λ:

$\lambda_0(000, B) = a_1(a_1 + a_4)(a_1 + a_5)(a_1 + a_4 + a_5)$,

$\lambda_1(A, B) = a_2(a_2 + a_4)(a_2 + a_5)(a_2 + a_4 + a_5)$,

$\lambda_2(A, B) = a_3(a_3 + a_4)(a_3 + a_5)(a_3 + a_4 + a_5)$,

$\lambda_3(A, 000) = (a_1 + a_2 + a_3)(a_1 + a_2 + a_3 + a_4)(a_1 + a_2 + a_3 + a_5)(a_1 + a_2 + a_3 + a_4 + a_5)$ *pentru codul $\mathcal{RM}(1,4)$. Pentru $(15,5)$ - codul BCH se elimină ultimul bit de control.*

Nodurile şi arcele sunt etichetate conform tabelelor:

δ_0/λ_0	000	100	001	101	010	110	011	111
000	000/0000	100/1111	001/0011	101/1100	010/0101	110/1010	011/0110	111/1001

δ_i/λ_i	000	100	001	101	010	110	011	111
000	0/0000	1/1111	—	—	—	—	—	—
100	1/1111	0/0000	—	—	—	—	—	—
001	—	—	0/0011	1/1100	—	—	—	—
101	—	—	1/1100	0/0011	—	—	—	—
010	—	—	—	—	0/0101	1/1010	—	—
110	—	—	—	—	1/1010	0/0101	—	—
011	—	—	—	—	—	—	0/0110	1/1001
111	—	—	—	—	—	—	1/1001	0/0110

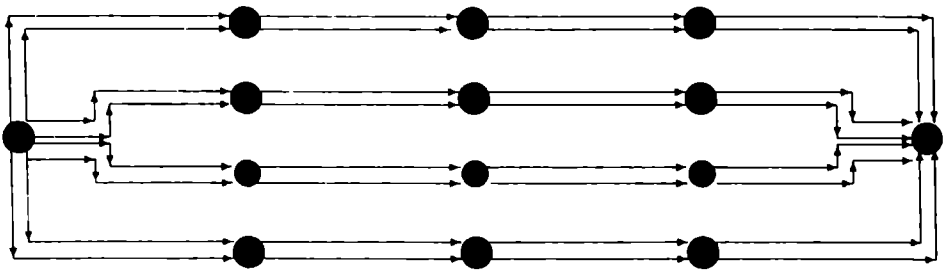
$(i=1,2)$

δ_3/λ_3	0000
000	ϵ/0000
100	ϵ/1111
001	ϵ/0011
101	ϵ/1100
010	ϵ/0101
110	ϵ/1010
011	ϵ/0110
111	ϵ/1001

respectiv

δ_3/λ_3	0000
000	ϵ/000
100	ϵ/111
001	ϵ/001
101	ϵ/110
010	ϵ/010
110	ϵ/101
011	ϵ/011
111	ϵ/100

De remarcat că această rețea se poate reprezenta folosind numai 4 *stări în loc de* 8*:*



Cele două arce care leagă două noduri sunt marcate cu etichete complementare. Excepție fac arcele de pe primul nivel, unde în definiția funcției δ_0 *este complementat numai primul bit* (a_1)*; ceilalți doi biți de informație* a_4a_5 *sunt identici pentru fiecare pereche de arce și formează marcajele nodurilor succesoare de pe nivelul* 1*:* $00, 01, 10, 11$.

13.3 *RLL* - coduri

Sistemele de comunicare actuale (telefonice, de înregistrare etc) - datorită vitezei mari de lucru diferă substanțial de canalele clasice (binar simetrice). Pentru acestea se folosesc filtre de bandă, capabile să elimine fenomenul de interferență care apare. Numite *canale cu răspuns parțial*, ele corectează aceste interferențe folosind algoritmi de tip Viterbi, incorporați prin construcție. Reușita decodificării corecte a cuvintelor recepționate este asigurată de supracodificări ale mesajului sursă. În prezent sunt folosite trei tipuri de supracodificări, care asigură transmiterea fără interferențe (în anumite limite) a mesajului: acestea sunt codurile cu lungime limitată și codurile balansate pentru codurile - bloc, turbo-codurile pentru codurile convoluționale.

Definiția 13.2 *Un cod cu lungime limitată (RLL - Run Length Limited Codes), sau* (d_0, k_0) *- cod, este un cod - bloc binar, cu proprietatea că orice două simboluri* 1 *consecutive sunt separate prin p zerouri, unde* $d_0 \leq p \leq k_0$.

Parametrul d_0 este folosit pentru controlul interferenței dintre simboluri (fenomen care apare la viteze de transmisie mari, apropiate de capacitatea de saturație fizică a canalului); k_0 impune numărul maxim

de zerouri care pot apare, pentru păstrarea unei rate de informaţie cât mai convenabile.

Fie C un (n,k) cod - bloc binar care poate fi reprezentat sub formă de $(n_1 \cdot n_2, k_1 \cdot k_2)$ - cod - tablou, şi $\mathbf{c} \in Z_2^n$ un cuvânt binar (numit *secvenţă de comutaţie*. Codul $C + \mathbf{c} = \{\mathbf{a} + \mathbf{c} | \mathbf{a} \in C\}$ va fi tot un (n,k) - cod, cu restricţii *RLL*.

Dificultatea acestei construcţii rezidă în modul de alegere a secvenţei $\mathbf{c}$. De aceea, vom detalia procedura de supra - codificare pentru obţinerea de coduri *RLL* cu $d_0 = 0$:

1. Fie C un $(n_1 \cdot n_2, k_1 \cdot k_2)$ cod - tablou binar.
2. Se defineşte cuvântul de comutaţie $\mathbf{c} = \mathbf{c_1 c_2} \ldots \mathbf{c_{n_2}}$, unde $\mathbf{c_i} = c_{i1} c_{i2} \ldots c_{in_1}$ $(1 \leq i \leq n_2)$ este o secvenţă binară de comutaţie pentru linia i, definită astfel:

   dacă n_1 este par, atunci $\mathbf{c_i}$ are un număr impar de 1,

   altfel, $\mathbf{c_i}$ are un număr par de 1.
3. Dacă $\mathbf{a} \in C$, $\mathbf{a} = (a_{ij})_{ij}$ este un cuvânt - cod, supra-codificarea sa este $\mathbf{a'} = (a_{ij} + c_{ij})_{ij}$, $1 \leq i \leq n_2$, $1 \leq j \leq n_1$.

Exemplul 13.15 *Fie* $(16,9)$ *- codul tablou binar* C_1 *construit în Exemplul 13.11. El are cuvintele - cod de forma*

$$\mathbf{a} = \begin{matrix} a_1 & a_2 & a_3 & p_1 \\ a_4 & a_5 & a_6 & p_2 \\ a_7 & a_8 & a_9 & p_3 \\ p_4 & p_5 & p_6 & p_7 \end{matrix}$$

Deoarece $n_1 = 4$, *secvenţa de comutaţie a fiecărei linii conţine un număr impar de* 1. *Vor exista opt variante posibile pentru* $\mathbf{c_i}$*:*

0001 0010 0100 1000 1110 1101 1011 0111,

deci se pot defini 8^4 *secvenţe de comutaţie distincte* $\mathbf{c}$.

Dacă se ia de exemplu $\mathbf{c} = 0100110110001110$, *codul RLL definit pe baza lui va conţine cuvintele - cod*

$$\mathbf{a} = \begin{matrix} a_1 & \overline{a_2} & a_3 & p_1 \\ \overline{a_4} & \overline{a_5} & a_6 & \overline{p_2} \\ \overline{a_7} & a_8 & a_9 & p_3 \\ \overline{p_4} & \overline{p_5} & \overline{p_6} & p_7 \end{matrix}$$

unde s-a notat $\overline{x} = 1 - x$ *(reamintim, calculele se efectuează în binar).*

Tehnica descrisă mai sus se poate extinde la coduri GAC, conducând la supracodificări cu proprietăţi RLL optime ([8]). Generalizarea este următoarea:

1. Dacă n_1 este par atunci $\mathbf{c_i}$ $(1 \leq i \leq n_2)$ conţine zero sau un număr impar de 1, altfel $\mathbf{c_i}$ $(1 \leq i \leq n_2)$ conţine un număr par nenul de 1;
2. Cuvântul de comutaţie $\mathbf{c}$ va conţine un număr par nenul de 1;
3. În general, $\mathbf{c_i}$ nu este cuvânt - cod al codului care formează liniile; dacă aceasta nu se poate evita, atunci $\mathbf{c_{i+1}}$ se alege de asemenea dintre cuvintele aceluiaşi cod.

Exemplul 13.16 *Fie un* $(8, 4)$ *- cod binar care codifică fiecare mesaj de informaţie* $a_1a_2a_3a_4$ *în*

$$\mathbf{a} = \begin{matrix} a_1 & a_1 + a_4 \\ a_2 & a_2 + a_4 \\ a_3 & a_3 + a_4 \\ p & p + a_4 \end{matrix}$$

unde $p = a_1 + a_2 + a_3$ *(vezi şi Exemplul 13.9).*

Deoarece n_1 *este par, alegem secvenţele de comutaţie* $\mathbf{c_i}$ $(1 \leq i \leq 4)$ *cu număr impar de* 1, *astfel încât numărul total de* 1 *din* $\mathbf{c}$ *să fie par. Putem lua de exemplu*

$$\mathbf{c_1} = 10, \quad \mathbf{c_2} = 00, \quad \mathbf{c_3} = 01, \quad \mathbf{c_4} = 00$$

Deci cuvântul de comutaţie este $\mathbf{c} = 10000100$ *şi cuvintele - cod cu restricţia RLL sunt*

$$\mathbf{a} = \begin{matrix} \overline{a_1} & a_1 + a_4 \\ a_2 & a_2 + a_4 \\ a_3 & \overline{a_3 + a_4} \\ p & p + a_4 \end{matrix} = (\overline{a_1}, a_1 + a_4, a_2, a_2 + a_4, a_3, \overline{a_3 + a_4}, p, p + a_4).$$

Codificarea celor 16 *mesaje de informaţie este dată de tabelul următor:*

0000	$\longrightarrow$	10000100	1000	$\longrightarrow$	01000111
0001	$\longrightarrow$	11010001	1001	$\longrightarrow$	00010010
0010	$\longrightarrow$	10001011	1010	$\longrightarrow$	01001000
0011	$\longrightarrow$	11011110	1011	$\longrightarrow$	00011101
0100	$\longrightarrow$	10110111	1100	$\longrightarrow$	01110100
0101	$\longrightarrow$	11100010	1101	$\longrightarrow$	00100001
0110	$\longrightarrow$	10111000	1110	$\longrightarrow$	01111011
0111	$\longrightarrow$	11101101	1111	$\longrightarrow$	00101110

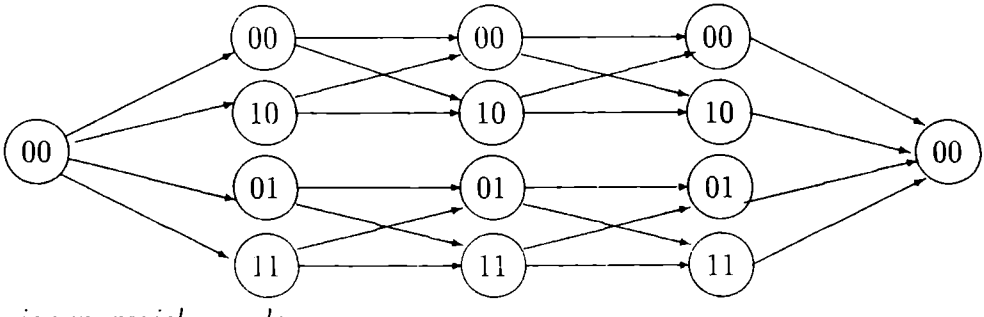
Am obţinut un $(8,4)$ *- cod binar care este un* $(0,4)$ *- cod RLL; se poate demonstra că aceasta este valoarea minimă care se poate obţine pentru* k_0 *în cazul* $(8,4)$ *- codurilor binare.*

Dacă se relaxează codul prin eliminarea ultimului simbol de control, se ajunge la un $(7,4)$ *- cod Hamming binar cu* $k_0 = 4$.

Pentru cuvântul de comutaţie $\mathbf{c} = 10\ 00\ 00\ 01$ *se obţine un* $(0,6)$ *- cod RLL, lucru care se verifică imediat.*

Pentru decodificarea codurilor *RLL* se poate folosi structura de reţea definită în cazul codurilor *GAC*, cu o singură modificare: la adnotarea arcelor, se înlocuieşte $\lambda_p(A,B)$ cu $\lambda_p(A,B) + \mathbf{c_p}$.

Exemplul 13.17 *Să reluăm* $(8,4)$ *- codul GAC de mai sus, a cărui reţea este descrisă în Exemplul 13.13. Folosind cuvântul de comutaţie* $\mathbf{c} = 00010001$, *se obţine un cod RLL cu aceeaşi structură de reţea:*



iar marcajele arcelor:

δ_0/λ_0	00	10	01	11
00	00/00	10/11	01/01	11/10

δ_1/λ_1	00	10	01	11
00	0/01	1/10	–	–
10	1/10	0/01	–	–
01	–	–	0/00	1/11
11	–	–	1/11	0/00

δ_3/λ_3	00
00	ϵ/01
10	ϵ/10
01	ϵ/00
11	ϵ/11

δ_2/λ_2	00	10	01	11
00	0/00	1/11	–	–
10	1/11	0/00	–	–
01	–	–	0/01	1/10
11	–	–	1/10	0/01

(pentru (7,4) - codul Hamming binar se ignoră ultimul bit din λ_3).

De remarcat că față de Exemplul 13.13 cuvântul de comutație a modificat numai marcajele nodurilor de pe nivelele 1 *și* 3*).*

13.4 Coduri balansate

Codurile balansate reprezintă o clasă specială de coduri *RLL*, în care fiecare cuvânt - cod are un număr egal de 0 și 1. Avantajele pe care le oferă această condiție (siguranță la transmiterea cu viteze mari a datelor, capacitatea de auto-control al tacților, posibilități de detectare și corectare a erorilor comparabile cu codurile - bloc) o recomandă ca o măsură eficientă de supracodificare.

Sunt multe modalități de transformare a unui cod - bloc binar într-un cod balansat; de exemplu dublarea fiecărui cuvânt - cod **a** cu o secvență identică (**aa**) sau cu caractere complementate (**a**$\overline{\mathbf{a}}$) conduc la coduri balansate. Construcția unui cod balansat prezentată în Figura 13.5 asigură posibilitatea construcției unei rețele de decodificare în maniera celei din paragrafele anterioare.

În construcția algoritmului **Balansat** mai trebuie rezolvată problema alegerii lui n_1. Aceasta se realizează folosind următoarea teoremă:

Teorema 13.4 *Fiind dat k, n_1 este de forma $n_1 = 2s$, unde s este cel mai mic număr pozitiv care verifică inegalitatea*

$$2^{k+2-4s} \leq 2 + C_{2s}^{s}. \qquad (2)$$

Demonstrație: Condiția ca n_1 să fie par rezultă din cerința ca în **u** jumătate din cele n_1 funcții g să fie f_2 și jumătate să fie $\overline{f_2}$. Fie deci $n_1 = 2s$.

Din condiția $2n_1 - 2 \leq k$ rezultă $k + 2 - 4s \geq 0$.

Deoarece **x** este format din $k + 2 - 4s$ caractere binare, sunt 2^{k+2-4s} secvențe **x** posibile. Din acestea, $2^{k+2-4s} - 2$ sunt construite folosind numai funcția f_2, deci trebuie să existe pentru ele cel puțin tot atâtea secvențe α. Deoarece numărul secvențelor binare α este C_{2s}^{s}, rezultă inegalitatea din enunț.

Figura 13.5:

Algoritm **Balansat**:

Fie $\mathbf{a} = a_1 a_2 \dots a_k \in Z_2^k$ $(k \geq 2)$ un mesaj de informaţie. Definim următorul cuvânt - cod tablou balansat asociat lui **a**:

1. Se alege n_1 par nenul astfel ca $2n_1 - 2 \leq k$; se notează $k_1 = 2n_1 - 2$, $n = 4 \cdot n_1$.

2. Se construieşte următorul cuvânt - cod tablou cu primii k_1 biţi de informaţie:

$$\mathbf{v} = \begin{array}{cccccc} a_1 & a_3 & a_5 & \dots & a_{k_1-1} & p_1 \\ \overline{a_1} & \overline{a_3} & \overline{a_5} & \dots & \overline{a_{k_1-1}} & \overline{p_1} \\ a_2 & a_4 & a_6 & \dots & a_{k_1} & p_2 \\ \overline{a_2} & \overline{a_4} & \overline{a_6} & \dots & \overline{a_{k_1}} & \overline{p_2} \end{array}$$

unde $p_1 = \sum_{i=1}^{n_1-1} a_{2i-1}$, $\sum_{i=1}^{n_1-1} a_{2i}$.

Cuvintele de această formă, citite pe coloană, formează un (n, k_1) - cod tablou balansat binar C_b. C_b este un cod - cadru, folosit pentru definirea caracterelor de control p_1 şi p_2.

3. Se definesc funcţiile $f_1, f_2 : Z_2 \to Z_2^4$ prin

$$f_1(x, y) = (x, \overline{x}, y, \overline{y}),$$
$$f_2(x, y) = (x \wedge y, x \wedge \overline{y}, \overline{x} \wedge y, \overline{x} \wedge \overline{y}),$$

unde $\wedge$ este operatorul boolean de conjuncţie. De remarcat că

$$\mathbf{v} = f_1(a_1, a_2) f_1(a_3, a_4) \dots f_1(a_{k_1-1}, a_{k_1}) f_1(p_1, p_2).$$

4. Se construieşte cuvântul - cod **u** folosind ultimele $k - k_1$ caractere de informaţie $\mathbf{x} = a_{k_1+1} \dots a_k$ astfel:

   1. Dacă $\mathbf{x} = 0 \dots 00$, atunci $\mathbf{u} = f_1(a_1, a_2) f_1(a_3, a_4) \dots f_1(p_1, p_2)$;

   2. Dacă $\mathbf{x} = 0 \dots 01$, atunci $\mathbf{u} = f_1(a_1, a_2) f_1(a_3, a_4) \dots \overline{f_1(p_1, p_2)}$;

   3. În rest, $\mathbf{u} = g(a_1, a_2) g(a_3, a_4) \dots g(a_{k_1-1}, a_{k_1}) g(p_1, p_2)$, unde

      (i) g este f_2 sau $\overline{f_2}$; fiecare din ele apare de un număr egal de ori (câte $n_1/2$ apariţii);

      (ii) Dacă se notează f_2 cu 0 şi $\overline{f_2}$ cu 1, lui **u** i se asociază o secvenţă α de n_1 caractere binare, cu proprietatea că relativ la primele k_1 caractere de informaţie, aplicaţia $\mathbf{x} \mapsto \alpha$ este izotonă faţă de relaţia de ordine lexicografică.

5. Cuvântul - cod final se obţine prin scrierea pe fiecare linie a unei valori $f_i(x, y)$ din **u**. El aparţine astfel unui cod GAC C de dimensiuni $n_1 \times 4$.

Condiţia de minim asigură unicitatea codificării. □

De remarcat că prin această metodă, operaţia de balansare nu păstrează proprietatea de cod liniar sau ciclic.

Exemplul 13.18 *Să construim un cod GAC balansat care să codifice* $k = 3$ *simboluri de informaţie. Folosind inegalitatea* (2), *se obţine* $s = 1$, *deci* $n_1 = 2$, $k_1 = 2$, $n = 8$.

Codul - cadru, care defineşte simbolurile de control, este un $(8, 2)$ *cod - tablou binar, având cuvintele - cod de forma (reamintim, reprezentarea desfăşurată se face pe coloane):*

$$\mathbf{v} = \begin{matrix} a_1 & p_1 \\ \overline{a_1} & \overline{p_1} \\ a_2 & p_2 \\ \overline{a_2} & \overline{p_2} \end{matrix} = a_1\ \overline{a_1}\ a_2\ \overline{a_2}\ p_1\ \overline{p_1}\ p_2\ \overline{p_2},$$

unde $p_1 = a_1$, $p_2 = a_2$. *Operaţia de codificare este reprezentată de tabloul:*

$000 \rightarrow 0101\ 0101$	$001 \rightarrow 0101\ 1010$
$010 \rightarrow 0110\ 0110$	$011 \rightarrow 0110\ 1001$
$100 \rightarrow 1001\ 1001$	$101 \rightarrow 1001\ 0110$
$110 \rightarrow 1010\ 1010$	$111 \rightarrow 1010\ 0101$

Exemplul 13.19 *Pentru* $k = 4$ *simboluri de informaţie se obţine de asemenea* $s = 1$ *şi acelaşi* $(8, 2)$ *- cod cadru, cu două simboluri de control. Cele două caractere de informaţie rămase,* a_3a_4 *asigură codificarea în felul următor:*

$$\begin{aligned} \alpha = 00 &\Longrightarrow \mathbf{u} = f_1(a_1, a_2)\underline{f_1(p_1, p_2)}; \\ \alpha = 01 &\Longrightarrow \mathbf{u} = f_1(a_1, a_2)\underline{f_1(p_1, p_2)}; \\ \alpha = 10 &\Longrightarrow \mathbf{u} = \underline{f_2(a_1, a_2)}f_2(p_1, p_2); \\ \alpha = 11 &\Longrightarrow \mathbf{u} = \underline{f_2(a_1, a_2)}f_2(p_1, p_2). \end{aligned}$$

Tabelul următor conţine cele 16 *mesaje de informaţie împreună cu cuvintele - cod generate de Algoritmul* **Balansat**.

0000	$\longrightarrow$	01010101	1000	$\longrightarrow$	10011001
0001	$\longrightarrow$	01011010	1001	$\longrightarrow$	10010110
0010	$\longrightarrow$	00011110	1010	$\longrightarrow$	01001011
0011	$\longrightarrow$	11100001	1011	$\longrightarrow$	10110100
0100	$\longrightarrow$	01100110	1100	$\longrightarrow$	10101010
0101	$\longrightarrow$	01101101	1101	$\longrightarrow$	10100101
0110	$\longrightarrow$	00101101	1110	$\longrightarrow$	10000111
0111	$\longrightarrow$	11010010	1111	$\longrightarrow$	01111000

S-a obţinut un $(8,4)$ *- cod balansat (care nu este cod liniar). Din studiul acestei tabele rezultă că am obţinut un cod cu distanţa minimă* $d = 4$.

Exemplul 13.20 *Următorul cod balansat a fost studiat de Blaum ([3]), odată cu introducerea metodei de balansare prezentată mai sus. Pentru* $k = 9$, *singura valoare posibilă este* $s = 2$, *deci* $n_1 = 4$, $k_1 = 6$, $n = 16$. *Cuvintele - cod cadru au forma*

$$\mathbf{v} = \begin{matrix} a_1 & a_3 & a_5 & p_1 \\ \overline{a_1} & \overline{a_3} & \overline{a_5} & \overline{p_1} \\ a_2 & a_4 & a_6 & p_2 \\ \overline{a_2} & \overline{a_4} & \overline{a_6} & \overline{p_2} \end{matrix},$$

unde $p_1 = a_1 + a_3 + a_5$, $p_2 = a_2 + a_4 + a_6$.

Procedura de codificare este dată de tabelul următor:

$a_7a_8a_9$	$u_1u_2u_3u_4$	$u_5u_6u_7u_8$	$u_9u_{10}u_{11}u_{12}$	$u_{13}u_{14}u_{15}u_{16}$
000	$f_1(a_1,a_2)$	$f_1(a_3,a_4)$	$f_1(a_5,a_6)$	$\overline{f_1(p_1,p_2)}$
001	$f_1(a_1,a_2)$	$f_1(a_3,a_4)$	$\overline{f_1(a_5,a_6)}$	$\overline{f_1(p_1,p_2)}$
010	$f_2(a_1,a_2)$	$\overline{f_2(a_3,a_4)}$	$\overline{f_2(a_5,a_6)}$	$\overline{f_2(p_1,p_2)}$
011	$f_2(a_1,a_2)$	$\overline{f_2(a_3,a_4)}$	$\overline{f_2(a_5,a_6)}$	$\overline{f_2(p_1,p_2)}$
100	$\overline{f_2(a_1,a_2)}$	$\overline{f_2(a_3,a_4)}$	$\overline{f_2(a_5,a_6)}$	$\overline{f_2(p_1,p_2)}$
101	$\overline{f_2(a_1,a_2)}$	$f_2(a_3,a_4)$	$\overline{f_2(a_5,a_6)}$	$\overline{f_2(p_1,p_2)}$
110	$\overline{f_2(a_1,a_2)}$	$\overline{f_2(a_3,a_4)}$	$\overline{f_2(a_5,a_6)}$	$f_2(p_1,p_2)$
111	$\overline{f_2(a_1,a_2)}$	$\overline{f_2(a_3,a_4)}$	$f_2(a_5,a_6)$	$f_2(p_1,p_2)$

S-a obţinut astfel un $(16,9)$ *- cod balansat cu distanţa minimă* $d = 8$.

Să codificăm de exemplu mesajul de informaţie $\mathbf{a} = 001101010$. *Pentru determinarea biţilor de control se construieşte cuvântul*

$$\mathbf{v} = \begin{matrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{matrix}$$

Deoarece $a_7 = 0$, $a_8 = 1$, $a_9 = 0$, *codul va fi generat folosind a treia linie din tabelul de sus:*

$$\mathbf{u} = 0001\ 1000\ 1101\ 1011.$$

Alte exemple de mesaje de informaţie şi codificările lor:

$$\begin{aligned} 000000000 &\Longrightarrow 0101010101010101 \\ 111111001 &\Longrightarrow 1010101010100101 \\ 000000101 &\Longrightarrow 1110000100011110 \\ 111111011 &\Longrightarrow 1000011110000111 \end{aligned}$$

Pentru realizarea unei structuri de reţea a codurilor balansate, dificultatea constă în faptul că aceste coduri nu sunt liniare, deci algoritmii utilizaţi până acum nu funcţionează. Se va utiliza următorul algoritm (Figura 13.6), specific codurilor balansate definite în acest paragraf:

Figura 13.6:

1. Pe baza algoritmilor anteriori, se construieşte structura de reţea R_b a codului - bloc cadru C_b; fie k_0 numărul ei de coloane (egal cu numărul de coloane din reprezentarea în tablou a cuvintelor - codului C_b.

2. Se copiază reţeaua R_b de $2^p - 1$ ori, unde $p = k - k_1$ este dat de pasul 3 al algoritmului **Balansat**; fiecare copie corespunde unei generări posibile a unui cuvânt - cod balansat.

   Toate subreţelele au o singură rădăcină şi o singură ţintă.

3. În fiecare din cele 2^p subreţele:

   (a) Arcul $((00\ldots0)_0, B_1)$ de pe nivelul 0 se marchează cu δ_0/λ_0 unde $\delta_0(00\ldots0, B) = a_{2n_1-1}\ldots a_k$, iar $\lambda_0(00\ldots0, B) = f_1(a_1, a_2)$.

   (b) Arcul spre ţintă are $\delta_{k_0}(A, 00\ldots0) = \epsilon$ şi $\lambda_{k_0}(A, 00\ldots0) = g(p_1, p_2)$ (notaţiile sunt cele din algoritmul **Balansat**).

4. În fiecare subreţea se defineşte valoarea δ_j/λ_j $(1 \leq j < k_0)$ a arcului (A_j, B_{j+1}) astfel: $\delta_j(A, B)$ este valoarea nemodificată din R_b, iar $\lambda_j(A, B) = g(a_{2j+1}, a_{2j+2})$.

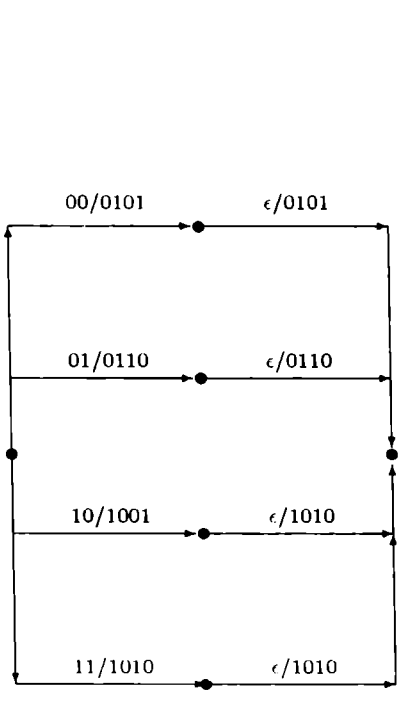
Exemplul 13.21 *Să reluăm* $(8, 4)$ *- codul balansat din Exemplul 13.18. După ce se construieşte reţeaua* $(8, 2)$ *- codului cadru - deoarece există doar un bit de informaţie suplimentar - avem* $p = 1$*, deci reţeaua se mai copiază de* $2^p - 1 = 1$ *ori.*

Sunt posibile două maniere de abordare:

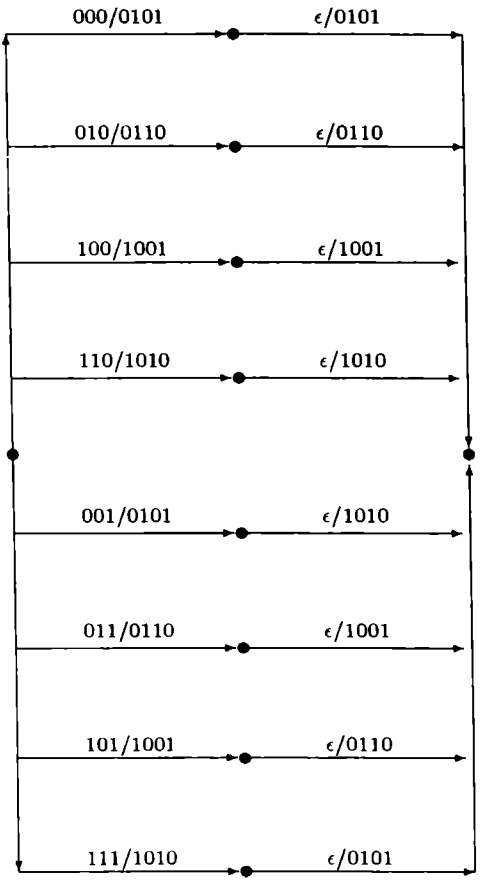
A. Dacă transmiterea datelor se face pe linii (specific codurilor balansate), reţeaua codului cadru este cea prezentată în Figura 13.7, poziţia

(a); din ea se obţine reţeaua (b) a codului balansat.

Figura 13.7:

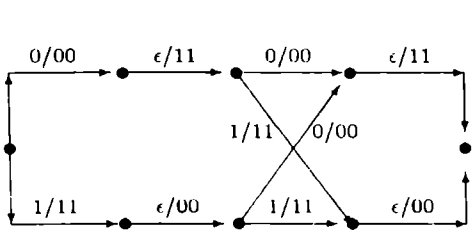


(a)

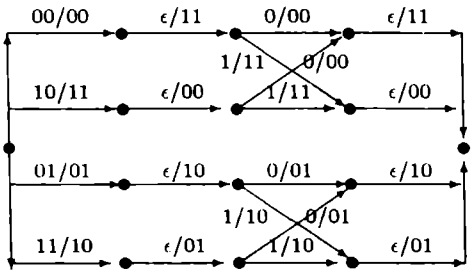


(b)

B. Dacă transmiterea se face pe coloane, apare o construcţie conformă cu cea a reţelelor anterioare (definite pentru codurile GAC) şi vom avea situaţiile (c) respectiv (d).



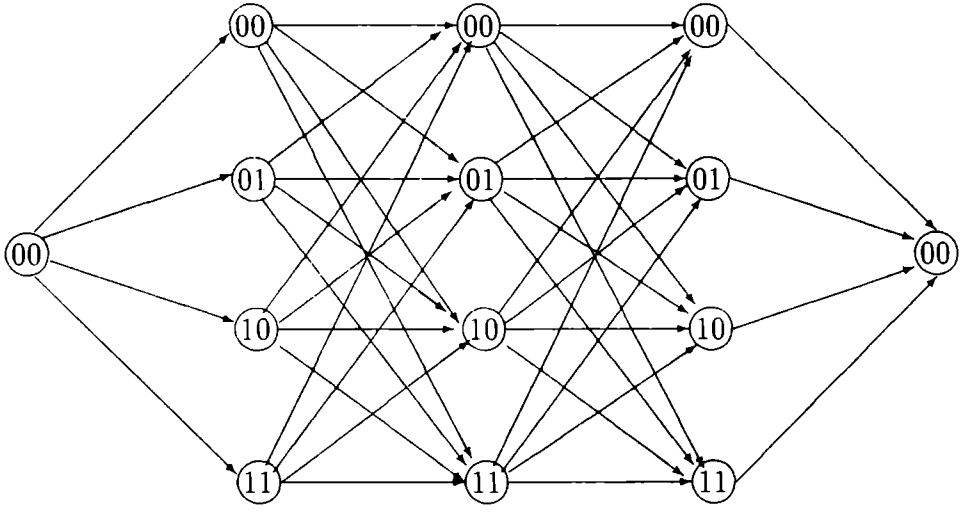
(c)



(d)

Exemplul 13.22 *Să construim reţeaua de (de)codificare pentru* $(16, 9)$ *- codul balansat definit în Exemplul 13.20. Conform procedurii, sunt necesare* 8 *subreţele, câte una pentru fiecare linie din tabelul de codificare. Deoarece cuvintele - cod sunt citite pe linii, fiecare subreţea are câte* 5 *coloane şi* 4 *stări.*

Pentru $a_7 = a_8 = a_9 = 0$ *(prima linie din tabel) subreţeaua construită plecînd de la* $(16, 6)$ *- codul cadru din Exemplul 13.20 are structura*



unde marcajul arcelor este dat de tabela:

p	δ_p/λ_p	00	01	10	11
0	00	00/0101	01/0110	10/1001	11/1010
1	00	00/0101	01/0110	10/1001	11/1010
	01	01/0110	00/0101	11/1010	10/1001
	10	10/1001	11/1010	00/0101	01/0110
	11	11/1010	10/1001	00/0101	00/0101
2	00	00/0101	01/0110	10/1001	11/1010
	01	01/0110	00/0101	11/1010	10/1001
	10	10/1001	11/1010	00/0101	01/0110
	11	11/1010	10/1001	01/0110	00/0101
3	00	ϵ/0101	–	–	–
	01	ϵ/0110	–	–	–
	10	ϵ/1001	–	–	–
	11	ϵ/1010	–	–	–

Pentru $a_7 = a_8 = 0$, $a_9 = 1$ *(a doua linie din tabel) se construieşte aceeaşi subreţea, în care valorile funcţiei* λ_3 *din marcajul arcelor ultimului nivel se complementează. Prin compunerea acestor două subreţele, se obţine un* $(16, 7)$ *- codul balansat cu distanţa minimă* $d = 8$.

Celelalte 6 *subreţele se definesc similar. Reţeaua finală a* $(16, 9)$ *- codului balansat cu* $d = 8$ *obţinută din compunerea lor, va avea* 32 *stări şi* 5 *nivele.*

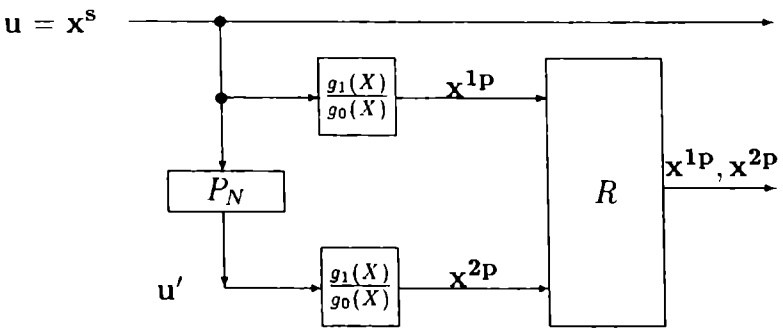
13.5 Turbo - coduri

În ultimii ani, viteza tot mai mare de transmisie a datelor - aproape de capacitatea maximă a canalelor de comunicaţie - a condus la modalităţi noi de codificare. Turbo codurile sunt prezentate prima dată în 1993 de Berrou, Glavier şi Thitimajshima şi combină sub formă de reţea (minim) două coduri convoluţionale. Modalitatea de decodificare este total deosebită de algoritmii cunoscuţi până acum (se folosesc capacităţi statistice de performanţă ale canalelor de transmisie). Ele asigură o rată de corectare a erorilor mult mai ridicată decât la codurile clasice; de aceea turbo codurile sunt folosite în transmisiile de pe staţiile cosmice automate lansate după anul 1993, precum şi canalele de satelit.

13.5.1 Structura unui turbo - cod

Un turbo - cod standard este reprezentat de Figura 13.8:

Figura 13.8: Turbo - codificator standard



El foloseşte două (2, 2) - coduri convoluţionale (care – fără a micşora generalitatea – au fost considerate identice), separate printr-un bloc P_N de permutare a celor N caractere de informaţie (N fiind o constantă fixată, dependentă de structura canalului de transmisie). Mesajul de informaţie este spart în blocuri $\mathbf{u} \in Z_q^N$; dacă se ignoră mecanismul de relaxare R, rata codificatorului este $1/3$ (N simboluri de informaţie se transformă în cuvinte - cod de lungime $3N$). Cele 3 ieşiri $(\mathbf{x^s}, \mathbf{x^{1p}}, \mathbf{x^{2p}})$ sunt apoi transmise pe coloană, ca un cod GAC.

Codificatorul

Pentru un (2, 2) - cod convoluţional, matricea generatoare poate fi considerată (într-o variantă simplificată, bazată pe structura de reţea a mesajelor) $G = (g_0(X),\ g_1(X))$. Codificatorul unui turbo - cod va folosi ca matrice generatoare o formă echivalentă recursivă:

$$G_R^T = \begin{pmatrix} \mathbf{1} & \dfrac{g_1(X)}{g_0(X)} \end{pmatrix}.$$

Din acest motiv, un codificator convoluţional pentru turbo - coduri este numit *Codificator Sistematic Recursiv* (RSE).

Un mesaj de informaţie $u(X)$ este codificat de codul convoluţional în $u(X)G = (u(X)g_0(X),\ u(X)g_1((X))$. RSE va realiza aceeaşi ieşire pentru mesajul $u'(X) = u(X)g_0(X)$, deoarece se verifică imediat relaţia $u(X)g_0(X)G_R = u(X)G$. Vom numi totuşi "cuvânt - cod" perechea de polinoame $u(X)G$ (deşi se mai efectuează o operaţie de înmulţire pentru obţinerea mesajului $u'(X)$).

Se observă că pentru un RSE, cuvântul cod are pondere finită dacă şi numai dacă mesajul de intrare se divide cu $g_0(X)$.

Corolarul 13.1 *Un mesaj sursă* $\mathbf{u}'$ *cu* $w(\mathbf{u}') = 1$ *se codifică într-un cuvânt - cod de pondere infinită.*

Demonstraţie: Evident, deoarece $u'(X) = X^p$ nu se divide cu $g_0(X)$. □

Corolarul 13.2 *Pentru orice polinom netrivial* $g_0(X) \in Z_q[X]$ *există o infinitate de mesaje sursă de pondere* 2 *care se codifică în cuvinte - cod de pondere finită.*

Demonstraţie: Pentru $g_0(X) \in Z_q[X]$, $g_0(X) \neq X^p$, există un n minim cu proprietatea $g_0(X) | X^n - 1$ (n este lungimea secvenţei pseudo-aleatoare generată de $g_0(X)$ - a se vedea paragraful 5.1, Capitolul 5).

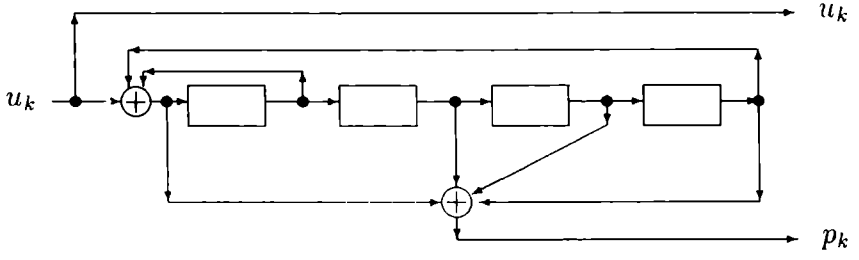
Orice secvenţă $\mathbf{u}'$ de forma $u'(X) = aX^i(X^n - 1)$, $a \in Z_q \setminus \{0\}$ are pondere 2 şi este divizibilă cu $g_0(X)$, deci codificarea prin RSE va genera un polinom cu un număr finit de termeni. □

Exemplul 13.23 *Fie* $g_0(X) = 1 + X + X^4$, $g_1(X) = 1 + X^2 + X^3 + X^4$ *polinoame din* $Z_2[X]$. *În mod uzual se foloseşte notaţia în octal; deci, cum* $g_0 = 11001_2 = 31_8$, $g_1 = 10111_2 = 27_8$, *vom avea* $(g_0\ g_1) = (31, 27)$.

Matricea generatoare este

$$G_R = \begin{pmatrix} 1 & \dfrac{1 + X^2 + X^3 + X^4}{1 + X + X^4} \end{pmatrix},$$

iar un circuit liniar care realizează acest RSE are forma:



(s-a notat cu u_k *simbolul de informaţie curent, iar cu* p_k *simbolul de control corespunzător).*

Cum $g_0(X)$ *este primitiv, lungimea secvenţelor este* $2^4 - 1 = 15$. *De exemplu, mesajul sursă* $u(X) = 1 + X^{15}$ *se codifică în* $(1 + X^{15}, 1 + X + X^2 + X^3 + X^5 + X^7 + X^8 + X^{11}) = (1000000000000001, 1111010110010000)$. $u(X) = X^7(1 + X^{15})$ *va genera acelaşi cuvânt - cod, cu o întârziere de şapte tacţi.*

Permutatorul

P_N este un bloc de permutare. Cele N simboluri de informaţie care constituie intrarea în primul codificator RSE sunt rearanjate înainte de a intra în al doilea codificator. Din considerente practice este preferabil ca permutarea folosită să nu păstreze nici o ordine anterioară a simbolurilor (deşi acest lucru este uneori dificil, mai ales pentru cuvinte de pondere mică). De asemenea, N trebuie să fie suficient de mare (în practică se foloseşte $N \geq 1000$). Aceste două cerinţe – uzuale în criptografie – sunt necesare în obţinerea de performanţe ridicate la decodificare.

Mecanismul de relaxare

Dacă pentru transmiterea de imagini din spaţiu sunt folosite coduri cu rate mici de informaţie (fiecărui bit îi corespund cel puţin 3 caractere - cod), în alte situaţii (comunicări prin satelit de exemplu) sunt preferabile rate mari (cel puţin 1/2). Rolul mecanismului de relaxare (vezi paragraful 2.7, Capitolul 2) este de a reduce periodic anumite caractere, pentru a scurta lungimea cuvintelor - cod. De obicei se elimină biţi de control; astfel, pentru a obţine o rată de informaţie 1/2 se pot elimina toţi biţii de control pari de la începutul cuvintelor - cod şi toţi biţii de control impari de la sfârşitul lor.

13.5.2 Decodificarea turbo - codurilor

Din construcţie rezultă că un turbo codificator este liniar, deoarece toate componentele sale sunt liniare. Codificările RSE sunt implementate prin circuite liniare, permutatorul este liniar deoarece poate fi modelat printr-o matrice de permutare. La fel, mecanismul de relaxare nu afectează liniaritatea, deoarece din toate cuvintele - cod se şterg simbolurile de pe aceleaşi poziţii. Importanţa liniarităţii constă în faptul că se poate lua ca referinţă cuvântul - cod nul. De asemenea, toate construcţiile sunt prezentate pentru cazul binar, cu simbolurile ± 1 (în loc de $\{0, 1\}$). Decodificatorul va lucra după principiul obişnuit al decodificării cele mai probabile.

Din păcate, utilizarea algoritmului Viterbi nu este posibilă aici, din cauza operaţiei de permutare folosită în decodificare. Totuşi, pentru subsecvenţe stricte, un astfel de algoritm poate da rezultate.

Primul algoritm de decodificare pentru turbo - coduri a fost propus de Berrou în 1993 ([13]), bazat pe ideile din [2]. Numit $BCJR$, el foloseşte

o decodificare caracter-cu-caracter (spre deosebire de algoritmul Viterbi care decodifică secvenţe - bloc de câte n caractere).

Vom prezenta acest algoritm, deşi - conceptual - el este complet diferit de ceilalţi algoritmi de decodificare construiţi până în prezent. Volumul enorm de date transmise conduce la o nouă strategie de decodificare, anume estimarea caracterelor corecte pe criterii statistice, folosind elemente de teoria probabilităţilor.

În cele ce urmează, se vor folosi următoarele notaţii:

- E_i - notarea codificatorului RSE i $(1 \leq i \leq 2)$;
- D_i - notarea decodificatorului i $(1 \leq i \leq 2)$;
- m - capacitatea de memorie (buffer) a codificatorului;
- S - mulţimea celor 2^m stări ale codificatorului;
- $\mathbf{x^s} = x_1^s x_2^s \dots x_N^s = u_1 u_2 \dots u_N$ secvenţa de informaţie care se codifică;
- $\mathbf{x^p} = x_1^p x_2^p \dots x_N^p$ cuvântul de control generat de codificator;
- $y_k = y_k^s y_k^p$ o recepţie (posibil perturbată) a lui $x_k^s x_k^p$;
- $\mathbf{y_a^b} = y_a y_{a+1} \dots y_b$;
- $\mathbf{y_1^N} = y_1 y_2 \dots y_N$ cuvântul recepţionat.

Algoritmul $BCJR$ (iniţial şi modificat)

O primă versiune a algoritmului se bazează pe decodificarea cea mai probabilă *aposteriori* (MAP - maximul aposteriori). Se realizează decodificarea

$$u_k = \begin{cases} +1 & \text{dacă } P(u_k = +1|\mathbf{y}) > P(u_k = -1|\mathbf{y}), \\ -1 & \text{altfel} \end{cases}.$$

Formal, $\hat{u}_k = sign[L(u_k)]$, unde $L(u_k)$ este logaritmul raportului probabilităţilor de potrivire aposteriori, definit prin relaţia

$$L(u_k) = log\left(\frac{P(u_k = +1|\mathbf{y})}{P(u_k = -1|\mathbf{y})}\right).$$

Dacă se ţine cont de faptul că se codifică în reţea, aceasta se scrie:

$$L(u_k) = log\left(\frac{\sum_{S^+} p(s_{k-1}=s', s_k=s, \mathbf{y})/p(\mathbf{y})}{\sum_{S^-} p(s_{k-1}=s', s_k=s, \mathbf{y})/p(\mathbf{y})}\right), \tag{3}$$

unde

- $s_k \in S$ este starea codificatorului la momentul k;
- S^+ este mulţimea perechilor (ordonate) (s', s) corespunzătoare tuturor stărilor de tranziţie $(s_{k-1} = s') \rightarrow (s_k = s)$ generate de $u_k = +1$;
- S^- este definită similar pentru $u_k = -1$.

Este posibil să simplificăm cu $p(\mathbf{y})$ în (3); deci este necesară doar o formulă pentru calculul lui $p(s', s, \mathbf{y})$. În [2], este construită o variantă sub forma

$$p(s', s, \mathbf{y}) = \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s), \tag{4}$$

unde:

- $\gamma_k(s', s) = p(s_k = s, y_k | s_{k-1} = s')$;
- $\alpha_k(s) = p(s_k = s, y_1^k)$ este calculat recursiv cu formula
$$\alpha_k(s) = \sum_{s' \in S} \alpha_{k-1}(s') \cdot \gamma_k(s', s)$$
cu condiţiile iniţiale $\alpha_0(0) = 1,\ \alpha_0(s \neq 0) = 0$
(codificatorul pleacă din starea 0).
- $\beta_k(s) = p(y_{k+1}^N | s_k = s)$ are formula recursivă de calcul
$$\beta_{k-1}(s') = \sum_{s \in S} \beta_k(s) \cdot \gamma_k(s', s)$$
şi condiţiile $\beta_N(0) = 1,\ \beta_N(s \neq 0) = 0$
(după N biţi de intrare codificatorul trebuie să ajungă la starea 0; restricţia se realizează alegând corespunzător ultimii m biţi, numiţi *biţi de încheiere*).

Aplicarea acestei variante de decodificare la turbo - coduri (algoritmul $BCJR$ iniţial a fost definit în 1974) are un neajuns: simplificarea cu $p(\mathbf{y})$ conduce la algoritmi numerici instabili. De aceea, Berrou ([13]) face o modificare a algoritmului, în felul următor:

Se definesc probabilităţile (modificate)

$$\tilde{\alpha}_k(s) = \alpha_k(s)/p(y_1^k), \quad \tilde{\beta}_k(s) = \beta_k(s)/p(y_{k+1}^N|y_1^N).$$

Prin împărţirea relaţiei (4) cu $p(\mathbf{y})/p(y_k) = p(y_1^{k-1}) \cdot p(y_{k+1}^N|y_1^k)$, se ajunge la

$$p(s', s|\mathbf{y}) \cdot p(y_k) = \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s).$$

Pentru că $p(y_1^k) = \sum_{s \in S} \alpha_k(s)$, valorile $\tilde{\alpha}_k(s)$ se pot determina din $\alpha_k(s)$ pe baza formulei

$$\tilde{\alpha}_k(s) = \frac{\alpha_k(s)}{\sum_{s \in S} \alpha_k(s)},$$

sau - folosind definiţia recursivă a lui $\alpha_k(s)$:

$$\tilde{\alpha}_k(s) = \frac{\sum_{s' \in S} \alpha_{k-1}(s') \cdot \gamma_k(s', s)}{\sum_{s \in S} \sum_{s' \in S} \alpha_{k-1}(s') \cdot \gamma_k(s', s)} = \frac{\sum_{s' \in S} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s)}{\sum_{s \in S} \sum_{s' \in S} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s)},$$

ultimul rezultat fiind obţinut prin împărţirea numărătorului şi numitorului cu $p(y_1^{k-1})$.

Definirea recursivă a lui $\tilde{\beta}_k(s)$ se obţine plecând de la

$$p(y_k^N|y_1^{k-1}) = p(y_1^k) \cdot \frac{p(y_{k+1}^N|y_1^k)}{p(y_1^{k-1}} = \sum_{s \in S} \sum_{s' \in S} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \frac{p(y_{k+1}^N|y_1^k)}{p(y_1^{k-1})} =$$

$$\sum_{s \in S} \sum_{s' \in S} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot p(y_{k+1}^N|y_1^k)$$

şi folosind definiţia recursivă a lui $\beta_{k-1}(s)$, prin împărţire se ajunge la relaţia

$$\tilde{\beta}_{k-1}(s') = \frac{\sum_{s \in S} \tilde{\beta}_k(s) \cdot \gamma_k(s', s)}{\sum_{s \in S} \sum_{s' \in S} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s)}.$$

În final, algoritmul $BCJR$ modificat va folosi valoarea $L(u_k)$ dată de relaţia

$$L(u_k) = log \left(\frac{\sum_{S^+} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{S^-} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)} \right). \tag{5}$$

Condiţiile la limită pentru $\tilde{\alpha}_k(s)$ şi $\tilde{\beta}_k(s)$ sunt aceleaşi cu cele de la $\alpha_k(s)$ respectiv $\beta_k(s)$.

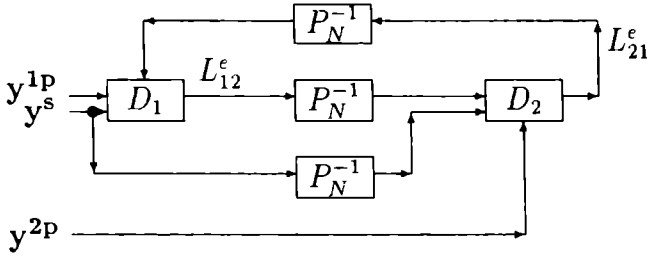
O altă versiune a algoritmului $BCJR$ foloseşte informaţia *apriori*.

Pentru acesta, avem

$$L(u_k) = log\left(\frac{P(\mathbf{y}|u_k = +1)}{P(\mathbf{y}|u_k = -1)}\right) + log\left(\frac{P(u_k = +1)}{P(u_k = -1)}\right).$$

Deoarece în mod normal $P(u_k = +1) = P(u_k = -1)$ (probabilitatea ca simbolul recepţionat să fie $+1$ este egală cu probabilitatea să fie -1), al doilea termen al sumei este zero în decodificatoarele uzuale. Pentru un turbo - decodor care lucrează recursiv, D_1 primeşte informaţie suplimentară de la D_2, care serveşte ca informaţie apriori. Similar, D_2 primeşte de la D_1 informaţie suplimentară ş.a.m.d. Ideea constă în faptul că D_2 poate da lui D_1 informaţii despre u_k la care acesta nu are acces (de exemplu caracterele de control generate de E_2); acelaşi lucru îl realizează D_1 pentru D_2.

Un circuit de decodificare bazat pe algoritmul $BCJR$ este:



S-a notat cu P_N^{-1} inversa matricii de permutare P_N din circuitul de codificare. L_{12}^e este informaţia suplimentară transmisă de la D_1 la D_2, iar L_{21}^e este cea transmisă de la D_2 la D_1. Deciziile finale de decodificare pot veni fie de la D_1, fie de la D_2.

Mai rămâne de văzut cum se poate obţine această informaţie suplimentară care circulă între cei doi decodificatori recursivi.

Definiţia lui $\gamma_k(s', s)$ se poate rescrie

$$\gamma_k(s', s) = P(s|s') \cdot p(y_k|s', s) = P(u_k) \cdot p(y_k|u_k),$$

unde evenimentul u_k corespunde tranziţiei $s' \to s$. Dacă se notează:

$$L^e(u_k) = log\left(\frac{P(u_k = +1)}{P(u_k = -1)}\right),$$

$P_+ = P(u_k = +1)$, $P_- = P(u_k = -1)$, avem

$$\left(\frac{\sqrt{P_-/P_+}}{1 + P_-/P_+}\right) \cdot \sqrt{P_+/P_-} = P_+ \text{ dacă } u_k = +1,$$

$$\left(\frac{\sqrt{P_-/P_+}}{1 + P_-/P_+}\right) \cdot \sqrt{P_-/P_+} = P_- \text{ dacă } u_k = -1.$$

În această situaţie se obţine

$$P(u_k) = \left(\frac{exp[-L^e(u_k)/2]}{1+exp[-L^e(u_k)]}\right)\cdot exp[u_k\cdot L^e(u_k)/2] = A_k\cdot exp[u_k\cdot L^e(u_k)/2],$$

şi (reamintim, $y_k = y_k^s y_k^p$, $x_k = x_k^s x_k^p = u_k x_k^p$)

$$p(y_k|u_k) \propto exp\left[-\frac{(y_k^s-u_k)^2}{2\sigma^2} - \frac{(y_k^p-x_k^p)^2}{2\sigma^2}\right] =$$
$$= exp\left[-\frac{(y_k^s)^2+u_k^2+(y_k^p)^2+(x_k^p)^2}{2\sigma^2}\right]\cdot exp\left[\frac{u_k\cdot y_k^s + x_k^p\cdot y_k^p}{\sigma^2}\right] =$$
$$= B_k\cdot exp\left[\frac{y_k^s\cdot u_k + y_k^p\cdot x_k^p}{\sigma^2}\right]$$

deci

$$\gamma_k(s',s) \propto A_k\cdot B_k\cdot exp[u_k\cdot L^e(u_k)/2]\cdot exp\left[\frac{u_k\cdot y_k^s + x_k^p\cdot y_k^p}{\sigma^2}\right] \quad (6)$$

Deoarece $\gamma_k(s',s)$ apare în (5) la numărător (unde $u_k=+1$) şi numitor (unde $u_k=-1$), factorul $A_k\cdot B_k$ se va reduce fiind independent de u_k. De asemenea, particularităţile de canal la transmisia lui ± 1 dau relaţia $\sigma^2 = N_0/(2E_c)$ unde E_c este energia de canal per bit. Din (6) se obţine

$$\gamma_k(s',s) \sim exp[u_k\cdot(L^e(u_k)+L_c\cdot y_k^s)/2 + L_c\cdot y_k^p\cdot x_k^p/2] =$$
$$= exp[u_k\cdot(L^e(u_k)+L_c\cdot y_k^s)/2]\cdot\gamma_k^e(s',s),$$

unde $L_c = \dfrac{4E_c}{N_0}$ şi $\gamma_k^e(s',s) = exp[L_c\cdot y_k^p\cdot x_k^p/2]$.

Combinând această relaţie cu (5) se ajunge la

$$L(u_k) = log\left(\frac{\sum\limits_{S^+}\tilde\alpha_{k-1}(s')\cdot\gamma_k^e(s',s)\cdot\tilde\beta_k(s)\cdot C_k}{\sum\limits_{S^-}\tilde\alpha_{k-1}(s')\cdot\gamma_k^e(s',s)\cdot\tilde\beta_k(s)\cdot C_k}\right) =$$
$$= L_c\cdot y_k^s + L^e(u_k) + log\left(\frac{\sum\limits_{S^+}\tilde\alpha_{k-1}(s')\cdot\gamma_k^e(s',s)\cdot\tilde\beta_k(s)}{\sum\limits_{S^-}\tilde\alpha_{k-1}(s')\cdot\gamma_k^e(s',s)\cdot\tilde\beta_k(s)}\right) \quad (7)$$

unde s-a notat $C_k = exp[u_k\cdot(L^e(u_k)+L_c\cdot y_k^s)/2]$.

A doua egalitate rezultă deoarece $C_k(u_k=+1)$ şi $C_k(u_k=-1)$ pot fi scoase ca factor. Primul termen al sumei (7) este numit *valoare de canal*, al doilea reprezintă informaţia apriori despre u_k (furnizată de un decodificator anterior), iar al treilea termen conţine informaţia suplimentară care se trimite la decodificatorul următor. Deci - de exemplu - la orice

iteraţie, D_1 calculează

$$L_1(u_k) = L_c \cdot y_k^s + L_{21}^e(u_k) + L_{12}^e(u_k),$$

unde $L_{21}^e(u_k)$ este informaţia suplimentară venită de la D_2 şi $L_{12}^e(u_k)$ este al treilea termen din (7), folosit ca o informaţie suplimentară trecută de la D_1 spre D_2.

Formalizat, această variantă a algoritmului se poate scrie astfel:

Algoritmul *BCJR*:

1. (Iniţializare):

$\mathbf{D_1}$: $\tilde{\alpha}_0^{(1)}(s) = \tilde{\beta}_N^{(1)}(s) := \begin{cases} 1 & \text{pentru } s = 0 \\ 0 & \text{pentru } s \neq 0 \end{cases}$

$L_{21}^e(u_k) := 0, \quad (k = 1, 2, \ldots, N).$

$\mathbf{D_2}$: $\tilde{\alpha}_0^{(2)}(s) := \begin{cases} 1 & \text{pentru } s = 0 \\ 0 & \text{pentru } s \neq 0 \end{cases}$, $\tilde{\beta}_N^{(2)}(s) := \tilde{\alpha}_N^{(2)}(s), \ \forall s.$

$L_{12}^e(u_k)$ se determină din D_1 după prima trecere, deci nu se iniţializează.

2. (A n-a iteraţie):

$\mathbf{D_1}$: pentru $k = 1, 2, \ldots, N$

- Se determină $y_k := y_k^s y_k^{1p}$ unde y_k^{1p} sunt biţii de control recepţionaţi pentru E_1 (posibil perturbaţi de canal);
- Se determină $\gamma_k(s', s)$ pentru toate tranziţiile posibile $s' \to s$;
- Se determină $\tilde{\alpha}_k^{(1)}(s), \ \forall s.$

pentru $k = N, N-1, \ldots, 2$ se determină $\tilde{\beta}_{k-1}^{(1)}(s), \ \forall s$;

pentru $k = 1, 2, \ldots, N$ se determină $L_{12}^e(u_k)$ cu valorile de probabilităţi asociate lui D_1.

$\mathbf{D_2}$: pentru $k = 1, 2, \ldots, N$

- Se determină $y_k := y_{P_N[k]}^s y_k^{2p}$;
- Se determină $\gamma_k(s', s)$ pentru toate tranziţiile posibile $s' \to s$;
- Se determină $\tilde{\alpha}_k^{(2)}(s), \ \forall s$:

pentru $k = N, N-1, \ldots, 2$ se determină $\tilde{\beta}_{k-1}^{(2)}(s), \ \forall s$;

pentru $k = 1, 2, \ldots N$ se determină $L_{21}^e(u_k)$ cu valorile de probabilităţi asociate lui D_2;

3. (După ultima iteraţie):

Pentru $k = 1, 2, \ldots, N$

- Se determină $L_1(u_k) := L_c \cdot y_k^s + L_{21}^e(u_{P_N^{-1}[k]}) + L_{12}^e(u_k)$;
- Dacă $L_1(u_k) > 0$ atunci $u_k := +1$ altfel $u_k := -1$;

4. Stop.

Scrierea algoritmului a ţinut cont de unele ipoteze implicite. Astfel:

- se presupune că cei doi codificatori lucrează corect, adică ultimii m biţi din mesajul de informaţie de lungime N se codifică astfel ca la sfârşit E_1 să ajungă la starea zero.

- decodificatorii deţin toată informaţia referitoare la reţeaua de codificare; astfel, ei au tabele complete cu simbolurile de informaţie şi de control pentru toate tranziţiile de stări $s' \to s$, matricile de permutare şi inversele lor.

13.6 Exerciţii

13.1 *Fie codul generat de polinomul* $g(X) = 1 + X + X^2 + X^3 + X^6$. *Codificaţi mesajele de informaţie*

$$m_1(X) = 1,\ m_2(X) = X^2,\ m_3(X) = 1 + X,$$
$$m_4(X) = 1 + X^2,\ m_5(X) = X^3,\ m_6(X) = 1.$$

Determinaţi şirul de biţi transmişi dacă se foloseşte o transpunere la adâncimea s *unde*

$$(a)\ s = 1 \qquad (b)\ s = 2 \qquad (c)\ s = 3.$$

13.2 *Demonstraţi Teorema 13.2*

13.3 *Folosind codurile definite în Exemplul 13.4, să se codifice următoarele mesaje de informaţie prin transpunerea încrucişată a lui* A_1 *cu* A_2*:*

(a) $\mathbf{m_1} = 0110,\ \mathbf{m_2} = 1011,\ \mathbf{m_3} = 1111,\quad s = 2;$
(b) $\mathbf{m_1} = 0110,\ \mathbf{m_2} = 1011,\ \mathbf{m_3} = 1111,\quad s = 3;$
(c) $\mathbf{m_1} = 0010,\ \mathbf{m_2} = 1111,\ \mathbf{m_3} = 1010,\quad s = 3;$
(d) $\mathbf{m_1} = 1000,\ \mathbf{m_2} = 0100,\ \mathbf{m_3} = 0010,\ \mathbf{m_4} = 0001,\ \mathbf{m_5} = 0011,$
$\mathbf{m_6} = 0100 \quad s = 3.$

13.4 *Găsiţi un rezultat analog Teoremei 13.3 dacă pentru* A_2 *se foloseşte* s *- cadrul de transpunere întârziată în loc de* s *- transpunere.*

13.5 *Construiţi codurile* $(11, 6)$ *şi* $(6, 3)$ *GAC.*

13.6 *Dacă mesajul de informaţie este* $\mathbf{a} = a_1a_2a_3a_4a_5 = 01001$, *găsiţi codificarea sa folosind un* $(15, 5)$ *- cod GAC.*

13.7 *Dacă mesajul de informaţie este* $\mathbf{a} = a_1a_2a_3a_4a_5 = 11011$, *găsiţi codificarea sa folosind un* $(15,5)$ - *cod GAC.*

13.8 *Justificaţi de ce codul Golay binar extins nu poate fi reprezentat folosind o descompunere tablou de tipul* 6×4.

13.9 *Construiţi reţeaua pentru* $(6,3)$ - *codul GAC cu* $d = 3$.

13.10 *Determinaţi nuărul de stări şi de coloane pentru următoarele coduri GAC:*

(a) $(16,10)$, $d = 4$; (b) $(11,6)$, $d = 3$; (c) $(63,42)$, $d = 3$.

13.11 *În diagrama - reţea a* $(20,14)$ - *codului GAC cu* $d = 4$, *determinaţi marcajul arcului dintre codurile* $(00011)_2$ *şi* $(10100)_3$.

13.12 *Arătaţi că diagrama reţea a* $(8,4)$ - *codului GAC* $(d = 4)$ *poate fi dedusă din reţeaua* $(6,3)$ - *codului GAC cu* $d = 3$.

13.13 *Mesajul de informaţie* $\mathbf{a} = 0001$ *este codificat cu* $(8,4)$ - *codul GAC având distanţa* $d = 4$. *La recepţie s-a primit secvenţa*

$$0,6 \;\; 0,95 \;\; 0,1 \;\; 0,1 \;\; 0,05 \;\; 0,9 \;\; 1,0$$

Arătaţi că structura de reţea poate decodifica şi corecta cele două erori care au apărut în transmisie.

13.14 *Construiţi un* $(3,2)(3,2)$ - *RLL cod tablou şi calculaţi parametrii săi* $(d_0,\ k_0)$.

13.15 *Construiţi un* $(12,8)$ - *cod GAC cu restricţii RLL şi calculaţi pentru el* $(d_0,\ k_0)$.

13.16 *Construiţi un GAC cod balansat pentru*

(a) $k = 10$; (b) $k = 13$..

13.17 *Să se determine numărul minim de stări ale unui* $(4,2)(4,3)$ - *cod tablou.*

13.18 *Să se determine numărul minim de stări şi numărul de coloane pentru codurile tablou*

(a) $(3,2)(6,5)$ (b) $(3,2)(100,99)$ (c) $(4,3)(7,6)$

13.19 *Mesajul de informaţie* 001101 *este codificat folosind un* $(3,2)(4,3)$ *- cod - tablou. Dacă se recepţionează*

$$\mathbf{x} = 0{,}6\ \ 0{,}55\ \ 0{,}1\ \ 0{,}9\ \ 0{,}95\ \ 0{,}0\ \ 0{,}1\ \ 1{,}0\ \ 0{,}9\ \ 1{,}0\ \ 0{,}1\ \ 0{,}9$$

arătaţi că algoritmul Viterbi poate efectua o decodificare corectă.

13.20 *Definiţi o secvenţă de comutaţie care transformă codul tablou* $(3,2)(3,2)$ *într-un cod RLL. Construiţi reţeaua acestui cod.*

13.21 *Aceeaşi problemă pentru* $(12,8)$ *- codul GAC.*

13.22 *Construiţi un* $(12,5)$ *cod balansat cu* $d = 6$. *Trasaţi reţeaua acestui cod.*

13.23 *Fie* $g_0(X) = 1 + X + X^3, \quad g_1(X) = 1 + X^2 + X^4 + X^5$ *din* $Z_2[X]$. *Construiţi circuitul liniar pentru codificatorul RSE.*

Codificaţi mesajele de informaţie $1 + X^2 + X^3$, $1 + X^7$, $X + X^4 + X^5$.

13.24 *Aceeaşi problemă pentru polinoamele*

$$g_0(X) = 1 + X^3 + X^4, \quad g_1(X) = X + X^3 + X^6.$$

13.25 *Să se construiască un turbo-codificator folosind codificatorii RSE din Exemplul 13.23,* $N = 3$, *matricea de comutaţie* $P_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$ *şi fără mecanism de relaxare.*

Să se codifice mesajul de informaţie 100 011 101.

Capitolul 14

Coduri aritmetice

14.1 Principii generale

Construcţiile oferite în cadrul teoriei codurilor detectoare şi corectoare de erori pot fi utilizate şi în alte domenii, dinafara celor clasice (de transmitere şi recepţie corectă a mesajelor). O aplicaţie legată de operaţiile aritmetice pe calculator este prezentată de van Lindt în [11].

Definiţia 14.1 *Fie $r \geq 2$ un număr întreg fixat. r - ponderea aritmetică a unui număr întreg x este definită prin*

$$w_{(r)}(x) = \begin{cases} 0 & \text{dacă } x = 0 \\ t & \text{altfel} \end{cases}$$

$$\text{unde } t = min \left\{ p \left| \sum_{i=1}^{p} a_i \cdot r^{n(i)} = x,\ |a_i| < r,\ n(i) \geq 0 \right. \right\}.$$

De obicei, r este asimilată noţiunii de *bază de numeraţie* . În practică se folosesc des cazurile $r = 2, 8, 10, 16$.

Exemplul 14.1
$w_{(10)}(5) = 1$,
$w_{(10)}(199) = 2$, *(deoarece* $199 = 200 - 1 = 2 \cdot 10^2 - 1 \cdot 10^0$*)*,
$w_{(2)}(-9) = 2$, *(deoarece* $-9 = -2^3 - 2^0$*)*,
$w_{(2)}(246) = 3$ *(deoarece* $246 = 256 - 8 - 2 = 2^8 - 2^3 - 2^1$*)*.

Propoziţia 14.1
1. $w_{(r)}(-x) = w_{(r)}(x)$;
2. $w_{(r)}(x + y) \leq w_{(r)}(x) + w_{(r)}(y)$.

Demonstraţie: Este lăsată ca exerciţiu. □

În cele ce urmează vom considera r ca o valoare fixată. Se numeşte *distanţă aritmetică* valoarea

$$d_A(x,y) = w_{(r)}(x-y).$$

Propoziţia 14.2
1. *d_A este o distanţă;*
2. *d_A este invariantă la translatare;*
3. *$d_A(x,y) \leq d_H(x,y)$, unde d_H este distanţa Hamming dintre două secvenţe numerice reprezentate în baza r.*

Demonstraţie: (1) Faptul că d_A este o distanţă se verifică imediat din Definiţia 14.1 şi Propoziţia 14.1.

(2) Deoarece $d_A(x+z,y+z) = w_{(r)}(x+z-y-z) = w_{(r)}(x-y) = d_A(x,y)$, rezultă că distanţa aritmetică este invariantă la translatare (proprietate pe care distanţa Hamming nu o are).

(3) Este lăsat ca exerciţiu. Precizăm că cele două numere reprezentate în baza r pot fi aduse la un număr egal de cifre completând eventual numărul mai scurt cu 0-uri (nesemnificative) în faţă. □

Definiţia 14.2 *Fie a, b numere întregi pozitive. Se numeşte AN - cod, mulţimea finită $C_{a,b} = \{a \cdot n | 0 \leq n < b\}$.*

Ideea folosirii AN - codurilor în aritmetica calculatorului este următoarea: să presupunem că trebuie calculată suma $n_1 + n_2$ (n_1, n_2 numere pozitive, mici în comparaţie cu b); fie S suma lor. Cele două numere se codifică în $a \cdot n_1, a \cdot n_2 \in C_{a,b}$. Dacă $S \equiv 0 \ (mod\ a)$ atunci suma a fost efectuată corect şi ea este $S\ div\ a$. Dacă nu, înseamnă că au apărut erori de calcul şi se ia ca rezultat acel număr n_3 cu proprietatea că $d_A(S, a \cdot n_3)$ este minimă.

Pentru a corecta orice combinaţie de maxim t erori, este necesar ca $C_{a,b}$ să aibă distanţa aritmetică $\geq 2t+1$, deci orice număr din $C_{a,b}$ are r - ponderea aritmetică minim $2t+1$.

Teorema 14.1 *Fie a un număr întreg fixat şi $s = min\{w_{(r)}(a \cdot n) | n \neq 0\}$. Atunci $s \leq 2$.*

Demonstraţie: Cazurile $a = 0$ şi $a = 1$ sunt banale. Pentru $a < 0$ se va lucra cu $-a$. Deci rămâne de studiat numai cazul $a > 1$. Vom folosi Teorema lui Fermat:

Dacă $(a, r) = 1$ atunci $r^{\phi(a)} \equiv 1 \ (mod\ a)$.

(reamintim, $\phi(a)$ este simbolul Euler).

Deci $a \mid r^{\phi(a)} - 1$. Cum $w_{(r)}(r^{\phi(a)} - 1) = 2$, afirmaţia este demonstrată pentru cazul când a şi r sunt prime între ele.

Să presupunem acum că $(a, r) = p > 1$. Vom avea $a = a_1 \cdot p$, $r = r_1 \cdot p$ cu $(a_1, r_1) = 1$. Atunci (notând $\phi(a_1) = n$), putem scrie $r^n - 1 =$
$= p^n \cdot r_1^n - 1 = p^n(r_1^n - 1) + (p^n - 1)$. Se ştie că $a_1 \mid r_1^n - 1$.

Dacă $(a_1, p) = 1$ atunci $a_1 \mid p^n - 1$, deci $a_1 \mid r^n - 1$, sau $a \mid p \cdot r^n - p$ şi cum $p < r$, teorema este demonstrată.

Dacă $(a_1, p) > 1$, raţionamentul se reia şi el se va termina după un număr finit de paşi (deoarece $a_1 < a$, $p < r$). □

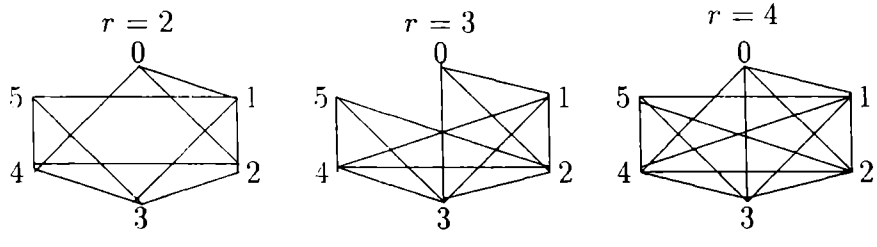
14.2 *AN* - coduri ciclice

Rezultatul menţionat în Teorema 14.1 conduce la o dificultate de alegere. O aritmetică a calculatorului eficientă recomandă alegerea unei valori mari pentru b. Pe de-altă parte Teorema 14.1 arată că în acest caz riscul de a avea r - ponderea aritmetică minimă cel mult 2 (deci de a nu putea corecta nici o eroare) este mare.

Problema se elimină dacă vom considera codurile *AN modulare*. Fie a, b numere prime şi $m = a \cdot b$. Definim $C_{a,b}$ ca subgrup al lui Z_m. Aceasta conduce la o altă definiţie a distanţei dintre două numere. Pentru determinarea ei, să luăm elementele lui Z_m ca vârfuri într-un graf Γ_m; $x, y \in Z_m$ sunt legate printr-un arc dacă şi numai dacă

$$\exists\, c, j \ (0 < c < r,\ j \geq 0), \quad x - y \equiv \pm c \cdot r^j \ (mod\ m).$$

Exemplul 14.2 *Să considerăm $a = 2$, $b = 3$, deci $m = 6$. În funcţie de baza r, graful Γ_6 are una din formele:*



Definiţia 14.3 *Distanţa modulară $d_m(x, y)$ dintre două numere $x, y \in Z_m$ este lungimea drumului minim dintre x şi y în graful Γ_m.*

Ponderea modulară a lui $x \in Z_m$ este $w_m(x) = d_m(x, 0)$.

Exemplul 14.3 *Pentru grafurile* Γ_6 *construite în Exemplul 14.2, cele trei tabele cu distanţe sunt:*

$r = 2$

d_m	0	1	2	3	4	5
0	0	1	1	2	1	2
1	1	0	1	1	2	1
2	1	1	0	1	1	2
3	2	1	1	0	1	1
4	1	2	1	1	0	1
5	2	1	2	1	1	0

$r = 3$

d_m	0	1	2	3	4	5
0	0	1	1	1	2	2
1	1	0	1	1	1	2
2	1	1	0	1	1	1
3	2	1	1	1	0	1
4	2	1	1	1	0	1
5	2	2	1	1	1	0

$r = 4$

d_m	0	1	2	3	4	5
0	0	1	1	1	1	2
1	1	0	1	1	1	1
2	1	1	0	1	1	1
3	1	1	1	0	1	1
4	1	1	1	1	0	1
5	2	1	1	1	1	0

De remarcat că alegerea lui m trebuie făcută cu grijă. De exemplu, dacă se ia $r = 3$, $m = 35$ vom avea $d_{35}(0, 12) = 1$ deoarece $12 = 3^{11} \ (mod\ 35)$. Practic însă, lucrând cu numere din Z_{35}, nu vom putea corecta erori pe poziţia corespunzătoare lui 3^{11}.

De aceea, aritmetica pe calculator consideră doar situaţia $m = r^n - 1$ $(n \geq 2)$, care elimină astfel de situaţii.

Orice număr întreg nenul x admite o reprezentare unică

$$x \equiv \sum_{i=0}^{n-1} c_i \cdot r^i \quad mod\ (r^n - 1),$$

cu $0 \leq c_i < r$ nu toţi nuli.

Deci Z_{r^n-1} poate fi interpretat ca fiind mulţimea $GF(r^n) \setminus \{0\}$ a tuturor secvenţelor nenule de lungime n, peste alfabetul $\{0, 1, \ldots, r-1\}$.

Pentru $a \cdot b = m = r^n - 1$, distanţa modulară devine distanţa obişnuită în Z_m şi $C_{a,b}$ are un comportament similar unui cod liniar.

Definiţia 14.4 *Un AN - cod ciclic de lungime* n *şi bază* r *este un subgrup multiplicativ* $C \subseteq Z_{r^n-1}$.

Evident, un astfel de subgrup este ideal principal în inelul Z_{r^n-1}, deci există $a, b \in \mathcal{Z}$ astfel încât $a \cdot b = r^n - 1$ şi $C = C_a$, adică:

$$C_a = \{a \cdot k | k \in \mathcal{Z},\ 0 \leq k < b\}.$$

Dacă $x \in C_a$, atunci $rx \ (mod\ r^n - 1)$ este tot în C_a, deoarece acesta grupul multiplilor lui a; pe de-altă parte noul număr este o permutare ciclică a lui x (ambele fiind scrise în baza r). Numărul $b = (r^n - 1)/a$ poate fi asimilat polinomului de control al unui cod ciclic.

Exemplul 14.4 *Fie* $r = 2$, $n = 11$. *Atunci* $m = 2^{11} - 1 = 2047$. *Să considerăm* $a = 23$, $b = 89$. *Se obţine AN - codul ciclic format din* 89

multipli ai lui 23 (până la 2047). Există 22 modalităţi de a semnala o eroare, fiecare din ele corespunzând unui număr de forma $\pm 2^j$ $(0 \leq j \leq 11)$. Acestea sunt exact toate numerele din $Z_{23} \setminus \{0\}$. Deci orice număr întreg din intervalul $[0, 2047]$ are distanţa modulară 0 sau 1 de exact un cuvânt - cod. Acest AN - cod ciclic este perfect şi poate fi considerat o generalizare a codurilor Hamming.

În [11] se face afirmaţia că nu există AN - coduri perfecte corectoare de o eroare pentru $r = 10$ sau $r = 2^k$ $(k > 1)$.

14.3 *AN* - coduri corectoare de mai multe erori

Pentru a construi *AN* - coduri capabile să corecteze erori multiple, va trebui definită o modalitate mai simplă de determinare a ponderii aritmetice sau modulare a numerelor întregi.

Conform Definiţiei 14.1, orice număr întreg x se poate scrie sub forma

$$x = \sum_{i=1}^{w_{(r)}} a_i \cdot r^{n(i)},$$

cu a_i, $n(i)$ numere întregi, $|a_i| < r$, $n(i) \geq 0$ $(0 \leq i \leq w_{(r)})$. Această reprezentare are defectul că nu este unică. De exemplu, pentru $r = 10$, numărul 99 se poate reprezenta în două moduri diferite:

$$99 = 9 \cdot 10 + 9 \cdot 10^0, \qquad 99 = 1 \cdot 10^2 - 1 \cdot 10^0.$$

Se poate obţine – prin impunerea de restricţii asupra coeficienţilor – o reprezentare unică a numerelor întregi.

Definiţia 14.5 *Fie $b, c \in \mathcal{Z}$, $|b| < r$, $|c| < r$. Perechea (b, c) se numeşte admisibilă dacă este adevărată una din relaţiile:*

(1) $b \cdot c = 0$;
(2) $b \cdot c > 0$ *şi* $|b + c| < r$;
(3) $b \cdot c < 0$ *şi* $|b| > |c|$.

De remarcat că ambele perechi (b, c), (c, b) sunt admisibile numai în cazurile (1) sau (2). Cazul (3) nu permite comutativitatea relaţiei de admisibilitate.

Exemplul 14.5 *Pentru $r = 2$ este posibil numai cazul (1). Deci o reprezentare $x = \sum_{i=0}^{\infty} c_i \cdot 2^i$ în care toate perechile (c_{i+1}, c_i) sunt admisibile, nu are doi coeficienţi consecutivi nenuli.*

Definiţia 14.6 *O reprezentare* $x = \sum_{i=0}^{\infty} c_i \cdot r^i$ *cu* $c_i \in \mathcal{Z}$, $|c_i| < r$ *şi* $\exists n_x$ *cu* $c_i = 0$ $\forall i > n_x$, *se numeşte NAF (non-adiacent form) dacă pentru orice* $i \geq 0$, *perechea* (c_{i+1}, c_i) *este admisibilă.*

Exemplul 14.6 *Pentru* $r = 10$ *putem scrie:*

$96 = -4 \cdot 10^0 + 0 \cdot 10^1 + 1 \cdot 10^2$ *(cazul (1)),*

$11 = 1 \cdot 10^0 + 1 \cdot 10^1$ *(cazul (2)),*

$38 = -2 \cdot 10^0 + 4 \cdot 10^1$ *(cazul (3)).*

De remarcat că reprezentarea lui $96 = 6 \cdot 10^0 + 9 \cdot 10^1$ *nu este în forma NAF deoarece perechea* $(9,6)$ *nu este admisibilă. La fel pentru reprezentările celorlaltor numere.*

Teorema 14.2 *Orice număr întreg* x *are o reprezentare NAF unică în baza* r. *Dacă aceasta este*

$$x = \sum_{i=0}^{\infty} c_i \cdot r^i,$$

atunci $w_{(r)}(x) = card(\{i | i \geq 0,\ c_i \neq 0\})$.

Demonstraţie: Fie $\sum_{i=0}^{\infty} b_i \cdot r^i$, $(|b_i| < r)$ o reprezentare a lui x în baza r, şi i cel mai mic număr cu proprietatea că perechea (b_{i+1}, b_i) nu este admisibilă. Putem presupune că $b_i > 0$ (altfel se va lucra cu $-x$). Vom înlocui b_i cu $b'_i = b_i - r$ şi b_{i+1} cu $b'_{i+1} = b_{i+1} + 1$ (dacă $b_{i+1} + 1 = r$, atunci $b'_{i+1} = 0$ şi facem deplasarea obişnuită de la adunare).

Dacă $b_{i+1} > 0$, atunci avem sau $b'_{i+1} = 0$, sau $b'_i \cdot b'_{i+1} < 0$ şi $b'_{i+1} = b_{i+1} + 1 > r - b_i = |b'_i|$ (deoarece perechea (b_{i+1}, b_i) nu era admisibilă).

Dacă $b_{i+1} < 0$, atunci sau $b'_{i+1} = 0$, sau $b'_i \cdot b'_{i+1} > 0$ şi $|b'_i + b'_{i+}| = r - b_i - b'_{i+1} < r$ deoarece $-b_{i+1} \leq b_i$ (perechea (b_{i+1}, b_i) nu era admisibilă).

Deci, (b'_{i+1}, b'_i) este admisibilă, şi se verifică similar dacă (b'_i, b_{i-1}) este admisibilă. Procedeul continuă până se ajunge la $i = 0$.

Să arătăm acum că reprezentarea NAF este unică. Presupunem că există $x \in \mathcal{Z}$ cu două astfel de reprezentări:

$$x = \sum_{i=0}^{\infty} c_i \cdot r^i = \sum_{i=0}^{\infty} c'_i \cdot r^i.$$

Considerăm – fără a micşora generalitatea – că $c_0 \neq c'_0$ şi $c_0 > 0$; deci $c'_0 = c_0 - r$. Atunci pentru c'_1 sunt posibile trei valori: $c_1 + 1$, $c_1 + 1 \pm r$. Dacă $c'_1 = c_1 + 1 - r$, atunci $c_1 \geq 0$, deci $c_0 + c_1 \leq r - 1$. Deoarece

$c'_0 \cdot c'_1 > 0$, avem $-c'_0 - c'_1 < r$, de unde $r - c_0 + r - c_1 - 1 < r$, deci $c_0 + c_1 > r - 1$, contradicţie.

Celelalte două cazuri se tratează similar. □

Reprezentarea NAF a unui număr x se poate afla direct şi pe baza teoremei:

Teorema 14.3 *Fie $x \in \mathcal{Z}, x \geq 0$. Considerăm reprezentările în baza r a numerelor*

$$(r+1) \cdot x = \sum_{i=0}^{\infty} a_i \cdot r^i, \quad x = \sum_{i=0}^{\infty} b_i \cdot r^i,$$

unde $a_i, b_i \in \{0, 1, \ldots, r-1\}$. Atunci reprezentarea NAF pentru x este

$$x = \sum_{i=0}^{\infty} (a_{i+1} - b_{i+1}) \cdot r^i.$$

Demonstraţie: Ştim că pentru două numere naturale nenule a, r, $[a/r]$ reprezintă câtul împărţirii celor două numere, iar $a - [a/r] \cdot r$ - restul.

Din reprezentarea din enunţ, rezultă că fiecare coeficient a_i se determină prin adunarea coeficienţilor corespunzători ai numerelor x şi $r \cdot x$, scrise în baza r. Să definim secvenţa numerică α_i, $i \geq 0$ astfel:

$$\alpha_0 = 0, \quad \alpha_i = \left[\frac{\alpha_{i-1} + b_{i-1} + b_i}{r} \right].$$

Atunci, conform observaţiei de la începutul demonstraţiei, $a_i = \alpha_{i-1} + b_{i-1} + b_i - \alpha_i \cdot r$. Dacă notăm $c_i = a_i - b_i$, avem $c_i = \alpha_{i-1} + b_{i-1} - \alpha_i \cdot r$.

Mai rămâne de verificat faptul că (c_{i+1}, c_i) este o pereche admisibilă. Relaţia $|c_{i+1} + c_i| < r$ rezultă imediat din definiţia lui α_i. Să presupunem $c_i > 0$, $c_{i+1} < 0$; atunci $\alpha_i = 0$. Vom avea $c_i = \alpha_{i-1} + b_{i-1}$, $c_{i+1} = b_i - r$ şi condiţia $|c_{i+1}| > |c_i|$ este echivalentă cu $\alpha_{i-1} + b_{i-1} + b_i < r$, adică $\alpha_i = 0$. Celălalt caz se arată analog. □

Exemplul 14.7 *Pentru a găsi reprezentarea NAF a numărului 98 în baza $r = 10$, avem*

$$\begin{aligned} 98 &= 8 \cdot 10^0 + 9 \cdot 10^1 + 0 \cdot 10^2 + 0 \cdot 10^3 + \ldots \\ 980 + 98 = 1078 &= 8 \cdot 10^0 + 7 \cdot 10^1 + 0 \cdot 10^2 + 1 \cdot 10^3 \end{aligned}$$

Deci, $98 = (7-9) \cdot 10^0 + (0-0) \cdot 10^1 + (1-0) \cdot 10^2 = -2 + 1 \cdot 10^2$.

Similar situaţiei din paragraful anterior, să considerăm acum cazul reprezentării modulare. Vom lua deci $m = r^n - 1$, $(n \geq 2)$.

Definiţia 14.7 *O reprezentare*

$$x \equiv \sum_{i=0}^{n-1} c_i \cdot r^i \ (mod\ m)$$

cu $c_i \in \mathcal{Z}, |c_i| < r$ *se numeşte* $CNAF$ *(cyclic* NAF*) pentru* x *dacă* $\forall i\ (0 \leq i \leq n-1)$, (c_{i+1}, c_i) *este admisibilă (se consideră* $c_n = c_0$*).*

Din Teoremele 14.2 şi 14.3 rezultă un rezultat similar pentru reprezentările $CNAF$:

Teorema 14.4 *Orice număr întreg* x *admite o reprezentare* $CNAF$ *modulo* m*. Această reprezentare este unică, exceptând cazul*

$$x \not\equiv 0\ (mod\ m), \qquad (r+1) \cdot x \equiv 0\ (mod\ m),$$

când sunt posibile două reprezentări.

Dacă $x \equiv \sum_{i=0}^{n-1} c_i \cdot r^i\ (mod\ m)$*, atunci*

$$w_m(x) = card(\{i | 0 \leq i < n,\ c_i \neq 0\}).$$

Demonstraţie: Construcţia este identică cu cea din demonstraţia Teoremei 14.3. Unicitatea se arată similar cu cea din demonstraţia Teoremei 14.2. Singura excepţie este cazul $b_{i+1} \equiv b_i\ (mod\ m)$. În acest caz sunt posibile două reprezentări:

$x \equiv b_0 + b_1 \cdot r + \ldots b_{n-1} \cdot r^{n-1}$ şi $x \equiv -b_1 - b_2 \cdot r - \ldots - b_{n-1} \cdot r^{n-1}$, ambele modulo m. □

14.4 Coduri Mandelbaum - Barrows

O clasă specială de AN coduri a fost definită de Mandelbaum şi Barrows, generalizată ulterior de van Lindt ([11]).

Iniţial este necesar un rezultat referitor la ponderea modulară în AN coduri ciclice, a cărui demonstrare se află în [11], pag. 127 – 128:

Teorema 14.5 *Fie* $C \subset \mathcal{Z}/(r^n - 1)$ *un* AN *- cod ciclic cu generator* a *şi* $b = (r^n - 1)/a = card(C)$*, cu proprietatea că* $\forall x \in C$ *are o reprezentare* $CNAF$ *unică. Atunci*

$$\sum_{x \in C} w_m(x) = n \left(\left[\frac{r \cdot b}{r+1} \right] - \left[\frac{b}{r+1} \right] \right).$$

Teorema 14.6 *Fie* b *un număr prim care nu divide* r*, cu proprietatea că grupul multiplilor* Z_b *este generat de* r *şi* -1*. Fie* n *un număr întreg pozitiv astfel ca* $r^n \equiv 1\ (mod\ b)$ *şi* $a = (r^n - 1)/b$*. Atunci codul* $C_a \subset \mathcal{Z}/(r^n - 1)$ *al multiplilor lui* a *este un cod echidistant cu distanţa*

$$\frac{n}{b-1} \left(\left[\frac{r \cdot b}{r+1} \right] - \left[\frac{b}{r+1} \right] \right).$$

Demonstraţie: Fie $x \in C_a \setminus \{0\}$. Atunci $x = a \cdot n \pmod{r^n - 1}$ cu $n \neq 0 \pmod{b}$. Din ipoteză rezultă că există j întreg cu $n \pm r^j \pmod{b}$. Deci $w_m(x) = w_m(\pm r^j a) = w_m(a)$. Aceasta arată că C_a este echidistant. Valoarea distanţei rezultă din Teorema 14.5. □

Exemplul 14.8 *Să considerăm* $r = 3$, $n = 4$. *Atunci* $r^n - 1 = 63 = 7 \cdot 9$. *Vom alege* $a = 9$, $b = 7$. *Aceste numere verifică condiţiile Teoremelor 14.5 şi 14.6. Codul* C_9 *este*

$$C_9 = \{09, 18, 27, 36, 45, 54\}.$$

El este un cod echidistant, distanţa dintre două cuvinte cod fiind

$$\frac{3}{6}\left(\left[\frac{28}{5}\right] - \left[\frac{7}{5}\right]\right) = 2$$

14.5 Exerciţii

14.1 *Demonstraţi Propoziţia 14.1.*

14.2 *Demonstraţi Propoziţia 14.2.* (3).

14.3 *Calculaţi* $w_{(2)}$, $w_{(10)}$ *şi* $w_{(16)}$ *pentru numerele*
(*a*) 100, (*b*) 32412, (*c*) 999, (*d*) 1024.

14.4 *Fie* $x \in \mathcal{Z}$. *Un cod Booth este o reprezentare* $x = \sum_{i=0}^{\infty} c_i \cdot 3^i$ *unde* $c_i \in \{-1, 0, 1\}$.

1. *Să se reprezinte în codul Booth numerele* 23, 455, 81, −6493;
2. *Să se arate că pentru orice număr întreg, codul Booth este unic.*

14.5 *Determinaţi AN - codurile ciclice din* Z_{2^3-1} *şi* Z_{3^3-1}.
Stabiliţi valorile a şi b pentru fiecare din ele.

14.6 *Generalizaţi Exemplul 14.4. Găsiţi un AN - cod ciclic perfect corector de o eroare pentru* $r = 3$.

14.7 *Scrieţi în forma NAF pentru* $r = 2$, $r = 10$ *şi* $r = 7$ *numerele*
(*a*) − 15, (*b*) 32075, (*c*) 5665, (*d*) − 992.

14.8 *Completaţi demonstraţia Teoremei 14.2, verificând unicitatea reprezentării NAF în cazurile* $c_1' = c_1 + 1$ *şi* $c_1' = c_1 + 1 + r$.

14.9 *În definiţia reprezentării NAF a numărului întreg x (cu completarea $n_0 = -1$), să se arate că*

$$n_x \leq k \quad \Longleftrightarrow \quad |x| < \frac{r^{k+2}}{r+1}.$$

14.10 *Considerăm reprezentarea ternară modulo $3^6 - 1$. Să se determine forma $CNAF$ pentru numărul 455.*

14.11 *Determinaţi cuvintele - cod din codul Mandelbaum - Barrows cu $b = 11$, $r = 3$, $n = 5$.*

Bibliografie

[1] J. Adamek - *Foundations of Coding*, Wiley - Interscience, 1991;

[2] L. Bahl, J. Cocke, F. Jelinek, J. Raviv - *Optimal decoding of linear codes for minimizing symbol error rate*, IEEE Inf. Theory, pp. 284-287, Martie 1974;

[3] M. Blaum - *A* $(16, 9, 6, 5, 4)$ *error correcting dc-free block code*, IEEE Transactions on Information Theory, vol. 34, 1988, pp. 38-141;

[4] C. Carlet - *Codes de Reed - Muller; Codes de Kerdok et de Preparata* (teză de doctorat), PARIS VI, 1990;

[5] G. Cullmann - *Coduri detectoare şi corectoare de erori*, Editura Tehnică, 1972;

[6] S. Guiaşu - *Teoria Codurilor*, Tipografia Universităţii Bucureşti, 1976;

[7] D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger, J.R. Wall - *Coding Theory; The Essentials*, Marcel Dekker, Inc, 1991;

[8] B. Honary, G. Markarian - *Trellis Decoding of Block Codes, A Practical Approach*, Kluwer Academic Publ., 1997;

[9] A. M. Kerdok - *A class of low-rate non linear codes*, Information and Control, 20 (1972), 182-187;

[10] J.H. van Lindt - *Coding Theory*, Springer Verlag, 1971;

[11] J.H. van Lindt - *Introduction to Coding Theory*, Springer Verlag, 1982:

[12] F.J.Mac Williams, N.J.A. Sloane - *The Theory of Error - Correcting Codes*, North Holland Mathematical Library, 1977;

[13] W.E.Ryan - *A Turbo Code Tutorial*, IEEE Trans. comm. pp. 1261-1271, Oct. 1996;

Index

Tiparul s-a executat sub cda 849/2001
la Tipografia Editurii Universității din București

Lucrarea are o construcție în mai multe dimensiuni. Una este cea a șirului de teme expuse și este pusă în evidență de împărțirea cărții în capitole și paragrafe. A doua dimensiune poate fi sesizată doar prin parcurgerea lucrării și se referă la nivelul de dificultate al tratării. Este vorba de modelele matematice folosite în codificare. Prezentarea este riguroasă, justificările metodelor prezentate sunt bine concepute, iar exemplele sunt sugestive. Problemele discutate la acest nivel vor interesa cu siguranță specialiștii sau studenții care se pregătesc să lucreze în domeniul transmiterii informației, dar și doctoranzii care lucrează la teze conexe acestui domeniu. În final, mai există un nivel practic al lucrării, evidențiat prin numeroasele exemple și probleme propuse și care conduce la o mai bună înțelegere a unor metode de codificare, unele foarte complexe. Ele vor interesa, desigur, tehnicienii și proiectanții care doresc să transmită informația prin mijloace cât mai fiabile.

ISBN 973-575-589-0