КОДИРАНЕ

И

КРИПТОГРАФИЯ

НИКОЛАЙ Л. МАНЕВ

С Ъ Д Ъ Р Ж А Н И Е

Въведение

Най-характерната черта на съвременното общество, заради която често е наричано "Информационно общество", е интензивния и глобален обмен на информация, превръщаш субекти, разположени на хиляди километри в събеседници или в единен работен екип. Налагайки се все повече като жизнено необходим, този информационен обмен обуславя бурното, лавинообразно развитие и разпространение на компютърните и комуникационни технологии, които навлизат дори в най-традиционните области на човешкото битие. Разбира се, това е един непрекъснат процес, който постоянно поражда нови научни и технологични задачи.

Един основен проблем на съвременните информационни технологии е огромния обем от данни, които трябва да се предават по канали (или съхраняват в среди) с най-разнообразни физически характеристики. Най-често те са твърде чувствителни към шум, което обуславя и нуждата от средства за контрол и отстраняване по възможност на възникващите грешки. Освен това информацията трябва да бъде в компактен и удобен за предаване и съхраняване вид. Решаването на тези задачи е предмет на *теория на кодирането*. Без нейната помощ е немислима разработката на съвременните цифрови комуникационни системи.

Необходимостта и стремежът дадена информация да бъде достъпна само за определен кръг от хора са стари колкото цивилизацията, но дори до преди 30-40 години това бяха проблеми основно на военните и дипломатически служби. Днес нещата са твърде различни. Развитието на компютърните технологии доведе до изграждане на компютърни мрежи обхващащи цели страни и континенти, които осигуряват достъп до най-разнообразни бази от данни със скъпоструваща информация за хора, продукти, услуги и процеси със съществено значение за значителен брой граждани. Разкриването и/или неправомерното използване на тази информация може да доведе до тежки последици засягащи огромни маси от хора. Всичко това превръща защитата на информацията в една от основните задачи не само за правителствените, а така също и за редица обществени организации, както

и за частните икономически субекти. За успешното решаване на тези добре познати, но поставени в нова светлина от Информационното общество проблеми, както и на редица нововъзникнали, все по-широко се привлича съвременната *криптография*.

Осигуряване отказоустойчивост на всички системи участващи в информационния обмен е друга съществена задача, към която двете научни дисциплини също имат принос.

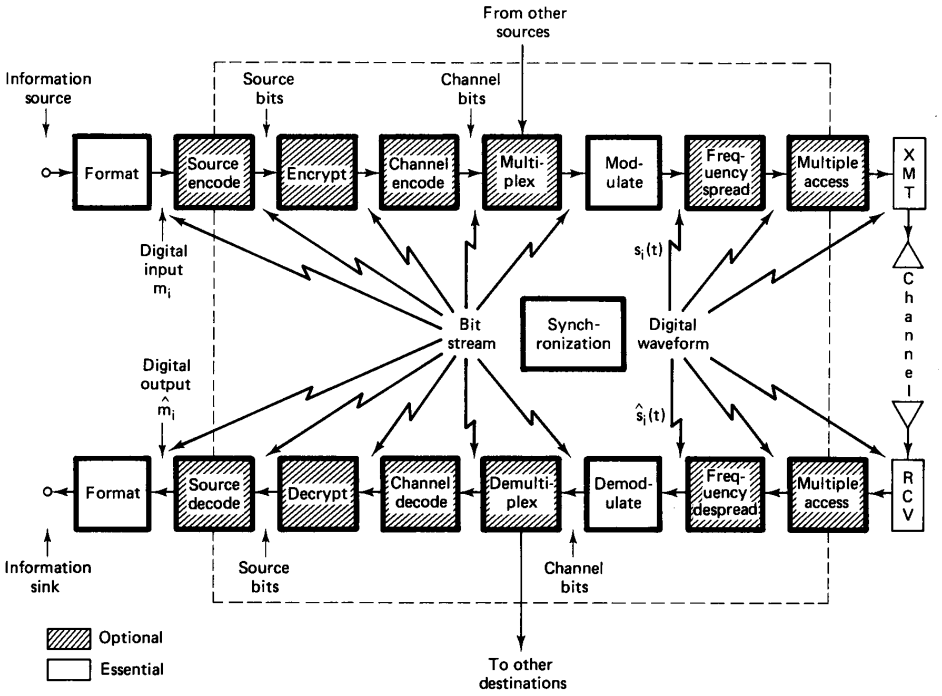
Теория на кодирането и съвременната криптография са пример за бурно развиващи се теоретични дисциплини, пряко свързани с практиката, но същевременно използващи твърде абстрактни резултати. Алгебрични теории, разработвани дори преди столетия, изведнъж намериха съществено приложение, което доведе до нов интерес към тях. В съчетание с възможностите на съвременните компютърни технологии те се превърнаха в качествено нови мощни инструменти за научни изследвания и приложни разработки. Ето и някои области на приложението им:

- Научно-изследователски компютърни мрежи за изчисления и съвместно ползване на бази данни;
- Компютъризация на административната дейност и обслужване: е-правителство;
- Компютърни системи за съхранение и обработка на здравна, правна, социално-икономическа и др. информация;
- Национална сигурност;
- Електронна търговия;
- Финансови разплащания;
- Електронно гласуване;
- Дистанционно обучение;
- Съвременни безжични комуникации;
- Физически устройства за съхранение на информация като лентови, CD и DVD устройства, полупроводникови памети.

Като начало на теорията на кодирането и съвременната криптография се посочва 1948 и 1949 години, когато се появяват двете основополагащи

статии [10, 11] на Клод Шенон (Claude Shannon). Но те разглеждат и изследват предаването на информацията в най-общ план и всъщност поставят началото на *теорията на информацията*. Теорията на кодирането и криптографията имат да решават по-специфични задачи и те се отделят по-късно като самостоятелни дисциплини.

Подробна схема на основните блокове, от които е изградена всяка съвременна комуникационна система е дадена на фигура 1 (препечатана от [12]). Белите блокове изобразяват модулите необходими за всяка комуникационна система, а потъмнените тези, които не са задължителни. Но твърде рядко се срещат системи, в които да липсват всички незадължителни блокове.



Фигура 1: Блок-диаграма на съвременна комуникационна система. [12]

Основните компоненти са:

- ***Форматиране***: Има за цел да преобразува получаваната от източника информация в удобна за предаване и съхраняване форма - най-често електрически сигнали представящи нули и единици.

- ***Кодиране на източника***: Компресира и преобразува, ако е необходимо, форматираните данни.

- ***Модулация и демодулация***: Преобразува цифровите сигнали в подходящи за предаване на големи разстояния по меден или оптически кабел, или чрез радиовълни. Устройството за модулация и демодулация се нарича *модем*.

- ***Синхронизация***: Основна е сихронизацията по време, която трябва да осигури разграничаването на символите и фреймовите. Други видове синхронизация са честотната и мрежовата.

- ***Шумозащитно кодиране (channel coding)***: Има за цел да намали вероятността за приемане на грешни данни при дадено съотношение сигнал/шум. Разбира се, това се заплаща с по-голяма сложност на системата и намаляване обема данни предавани за единица време.

- ***Криптиране и декриптиране***: Осигурява защита на информацията от подправяне или разкриване от неоторизирани субекти (т.е. нямащи съответните права за достъп).

- ***Мултиплексиране***: Служи за комбиниране на сигналите от няколко източника с цел да се уплътни във времето и честотната лента канала.

- ***Spread-spectrum технологии***: Разработени първоначално за военни цели тези технологии позволяват маскиране на излъчването и го правят по-слабо чувствително към естествени или преднамерени интерференции.

В настоящия курс ще се спрем по-подробно на шумозащитното кодиране и криптирането. Бегло ще разгледаме кодирането на източника и модулацията. Реализацията на останалите компоненти няма да бъде дискутирана.

Курсът има уводен характер и е съобразен за хорариум 30-40 часа. По тази причина редица важни раздели на теория на кодирането и на криптографията въобще не са засегнати или се разглеждат твърде бегло. Но като цяло дава представа за целите, които се преследват и задачите, които трябва да се решават при създаване на съвременни комуникационни системи.

Глава 1

Теория на информацията

1.1 Ентропия и информация.

Дефиниция 1.1.1 *Нека X е случайна променлива приемаща две стойности (например 0 и 1) с вероятности p и $1-p$. Функцията*

$$H(X) = -p\log_2 p - (1-p)\log_2(1-p)$$

се нарича ***ентропия (неопределеност)*** *на X.*

$H(X)$ приема стойности в интервала [0,1] и естествено максимумът ѝ (т.е. пълната неопределеност) се достига при $p = 1/2$. Графиката ѝ е дадена на фигура 1.1.

Търсим функция $H(p_1, \ldots, p_n)$ на аргументите $p_i \geq 0,\ p_1 + \cdots + p_n = 1$, която удовлетворява следните условия:

A1. $H(p_1, \ldots, p_n)$ достига максимум при $p_1 = \cdots = p_n = 1/n$.

A2. $H(p_1, \ldots, p_n)$ е симетрична функция на аргументите си $p_1, \ldots, p_n$.

A3. $H(p_1, \ldots, p_n)$ е непрекъсната функция на аргументите си $p_1, \ldots, p_n$.

A4. $H(p_1, \ldots, p_n) \geq 0$

A5. $H(p_1, \ldots, p_n, 0) = H(p_1, \ldots, p_n)$.

A6. $H(\frac{1}{n}, \ldots, \frac{1}{n}) \leq H(\frac{1}{n+1}, \ldots, \frac{1}{n+1})$.

A7. $H(\frac{1}{mn}, \ldots, \frac{1}{mn}) = H(\frac{1}{m}, \ldots, \frac{1}{m}) + H(\frac{1}{n}, \ldots, \frac{1}{n})$.

Фигура 1.1: Графика на функцията $H(X)$.

A8. Нека $p = p_1 + \cdots + p_n$ и $q = q_1 + \cdots + q_m$, където p_i и q_j са неотрицателни, $p, q > 0$ и $p + q = 1$. Тогава

$$H(p_1, \ldots, p_n; q_1, \ldots, q_m) = H(p, q) + pH(\frac{p_1}{p}, \ldots, \frac{p_n}{p}) + qH(\frac{q_1}{q}, \ldots, \frac{q_m}{q}).$$

Теорема 1.1.2 *(*За единственост*) Нека $H(p_1, \ldots, p_n)$ е функция дефинирана за всяко n и всяко множество от стойности удовлетворяващи $p_i \geq 0$, $p_1 + \cdots + p_n = 1$. Ако H удовлетворява* A1-A8*, то*

$$H(X) = -\lambda \sum_k p_k \log_2 p_k,$$

където λ е произволна положителна константа и сумирането се извършва по всички k, за които $p_k > 0$.

Дефиниция 1.1.3 *Нека X е случайна величина приемаща n стойности с вероятности $p_1, p_2, \ldots, p_n$, където $\sum_{i=1}^n p_i = 1$. **Ентропия (неопределеност)** на X се нарича функцията*

$$H(X) = -\sum_k p_k \log_2 p_k,$$

където сумирането се извършва по всички k, за които $p_k > 0$.

Теорема 1.1.4 *За всяко n е в сила*

$$H(p_1, \ldots, p_n) \leq \log_2 n,$$

като равенство се достига тогава и само тогава, когато $p_1 = \cdots = p_n = 1/n$.

Дефиниция 1.1.5 *Нека $X_1, X_2, \ldots, X_n$ са множество от случайни величини приемащи само краен брой стойности. Под* ***взаимна ентропия*** *на X_i се разбира функцията*

$$H(X_1, \ldots, X_n) = -\sum p(x_1, \ldots, x_n) \log p(x_1, \ldots, x_n),$$

където $p(x_1, \ldots, x_n) = \Pr(X_1 = x_1, \ldots, X_n = x_n)$ и сумирането се извършва по всички множества от стойности $\{x_1, \ldots, x_n\}$.

Теорема 1.1.6 *Ако X и Y са две случайни величини приемащи само краен брой стойности, то*

$$H(X, Y) \leq H(X) + H(Y),$$

като равенство се достига тогава и само тогава, когато X и Y са независими.

Доказателство. Нека X и Y приемат стойности $\{a_i,\ 1 \leq i \leq n\}$ и $\{b_j,\ 1 \leq j \leq m\}$, съответно. Нека означим $p_i = \Pr(X = a_i)$, $q_j = \Pr(Y = b_j)$ и $r_{ij} = \Pr(X = a_i \mid Y = b_j)$, $1 \leq i \leq n$, $1 \leq j \leq m$. Тъй като $p_i = \sum_j r_{ij}$ и $q_j = \sum_i r_{ij}$, то

$$\begin{aligned} H(X) + H(Y) &= -\left(\sum_i p_i \log p_i + \sum_j q_j \log q_j \right) \\ &= \left(\sum_i \sum_j r_{ij} \log p_i + \sum_j \sum_i r_{ij} \log q_j \right). \end{aligned}$$

$\diamond$

Нека X и Y са две случайни величини приемащи стойности $\{a_i,\ 1 \leq i \leq n\}$ и $\{b_j,\ 1 \leq j \leq m\}$, съответно. Да означим с XY случайната величина приемаща стойности $\{a_i b_j\}$. Съгласно дадените дефиниции $H(XY) = H(X, Y)$.

Дефиниция 1.1.7 ***Условна ентропия*** *на Y при условие събитието S_i : $X = a_i$, наричаме*

$$H(Y|S_i) = -\sum_{j=1}^{m} \Pr(Y = b_j|X = a_i) \log \Pr(Y = b_j|X = a_i).$$

Теорема 1.1.8

$$H(X,Y) = H(X) + \Pr(X = a_1)H(Y|S_1) + \cdots + \Pr(X = a_n)H(Y|S_n). \quad (1.1)$$

Дефиниция 1.1.9 ***Условна ентропия*** *на Y при условие X наричаме*

$$H(Y|X) = \sum_{i=1}^{m} \Pr(X = a_i)H(Y|X = a_i).$$

При така дадената дефиниция равенство (1.1) можем да запишем във вида

$$H(X,Y) = H(X) + H(Y|X). \quad (1.2)$$

Дефиниция 1.1.10 *Под* ***информация*** *на (асоциирана със) събитието E разбираме*

$$I(E) \stackrel{\text{def}}{=} -\log_2 \Pr(E).$$

Например, ако E е събитието източник, който излъчва с равна вероятност 0 и 1 да произведе n последователни 0, то

$$I(E) = -\log_2 \left(\frac{1}{2}\right)^n = n.$$

Нека X приема стойности $\{a_i,\ 1 \leq i \leq n\}$ с вероятности $p_1, p_2, \ldots, p_n$. Съгласно дадените дефиниции можем да напишем

$$H(X) = \sum_{i=1}^{n} p_i I(X = a_i),$$

което интерпретира ентропията като усреднена стоойност за информацията, асоциирана с елементарните събития съответсващи на X.

В този дух ще дадем следното определение:

Дефиниция 1.1.11 ***Информация за Y съдържаща се (пренасяна от) X*** *се нарича*

$$I(Y|X) = H(Y) - H(Y|X).$$

С други думи $I(Y|X)$ е мярка за неопределеността на Y, когато X е определено. Очевидно в сила са:

$I(Y|Y) = H(Y)$

$I(Y|X) = 0$ тогава и само тогава, когато X и Y са независими.

1.2 Кодиране на дискретен източник без памет.

Под **дискретен източник** ще разбираме случаен процес, който поражда редица от символи $x_1, x_2, x_3, \ldots$ принадлежащи на крайно множество $A = \{a_1, a_2, \ldots, a_n\}$, наричано **азбука на източника**. Азбуката се характеризира с вероятностното си разпределение $(p_1, p_2, \ldots, p_n)$, т.е. символът a_i се появява на изхода на източника с вероятност p_i и $\sum_{i=1}^{n} p_i = 1$.

Дефиниция 1.2.1 *Ако вероятността символът a_j да се появява на изхода на източника като i-ти символ x_i не зависи от i и породените до този момент символи x_k, т.е. ако* $\Pr(x_i = a_j) = p_j$, *то източникът се нарича* ***дискретен източник без памет.*** *Числото*

$$H(\mathfrak{A}) = -\sum_{k=1}^{n} p_k \log_2 p_k,$$

се нарича ***ентропия на източника*** $\mathfrak{A}$.

Дефиниция 1.2.2 *Нека $\mathfrak{A}$ е източник с азбука $A = \{a_1, a_2, \ldots, a_n\}$.* ***Кодиране*** *(или* ***код****) на $\mathfrak{A}$ се нарича всяко изображение f на A в множеството $\mathcal{W}$ от крайните редици от символи на някаква азбука $W = \{\mathfrak{w}_1, \mathfrak{w}_2, \ldots, \mathfrak{w}_m\}$.*

За така дефинирания тип кодиране често се употребява термина *побуквено кодиране*. Например, елементите на $\mathcal{W}$ могат да бъдат двоични редици, в общия случай с различна дължина (т.е. $W = \{0, 1\}$). С $|f(a_i)|$ ще бележим *дължината* на редицата $f(a_i)$.

Дефиниция 1.2.3 ***Средна дължина (цена)*** *на кода f се дефинира като числото*

$$L(f) \stackrel{\text{def}}{=} \sum_{i=1}^{n} p_i |f(a_i)|.$$

Дефиниция 1.2.4 *Казваме, че кодът (кодирането) f е* ***разделим****, ако всяка крайна редица от елементи на $\mathcal{W}$ е образ най-много на едно съобщение $x_1 x_2 \ldots x_k$, $x_i \in A$.*

Очевидно необходимо условие кодът да е разделим е f да бъде инективно изображение, т.е. $a_i \neq a_j$ да влече $f(a_i) \neq f(a_j)$.

Дефиниция 1.2.5 *Един код f се нарича* ***префиксен****, ако не съществуват $a_i \neq a_j$, такива че $f(a_i)$ да е начало (префикс) на $f(a_j)$.*

Пример 1.2.1 Нека $A = \{a_1, a_2, a_3, a_4\}$. Кодът

$$f(a_1) = 0,\ f(a_2) = 10,\ f(a_3) = 110,\ f(a_4) = 1110$$

е разделим и префиксен. (0 изпълнява ролята на запетая, която разделя кодовите думи.) Докато кодът

$$g(a_1) = 0,\ g(a_2) = 01,\ g(a_3) = 11,\ g(a_4) = 0110,$$

очевидно не е разделим.

Теорема 1.2.6 *(Теорема ва Шенон) Съществува разделим код f на източника $\mathfrak{A}$, такъв че е изпълнено*

$$\frac{H(\mathfrak{A})}{\log_2 |W|} \leq L(f) \leq 1 + \frac{H(\mathfrak{A})}{\log_2 |W|},$$

като долната граница се удовлетворява от всеки разделим код.

Дефиниция 1.2.7 *Нека f е разделим код. Казваме, че f е* ***оптимален*** *за $\mathfrak{A}$, ако за всеки друг разделим код g на $\mathfrak{A}$ е изпълнено $L(f) \leq L(g)$.*

Теорема 1.2.8 *За всеки източник съществува оптимален префиксен код.*

Както отбелязахме, изучаването на кодиране на източника не попада сред основните цели на настоящия курс. Затова ще се ограничим с гореказаното, а за доказателствата и по-подробна информация по темата препоръчваме като подходящо четиво на български език, учебниците по дискретна математика [1] и [2].

1.3 Предаване на данни при наличие на шум.

Най-общо, всеки реален канал за предаване на данни може да се разглежда като черна кутия, която получавайки на входа x, дава на изхода стойност y, която се явява случайна величина по отношение на x. Тъй като на входа постъпват стойностите на случайна величина (както отбелязахме източника се разглежда като случаен процес), то ползвайки математическата терминология може да кажем, че каналът представлява трансформация на една случайна величина X в друга случайна величина Y. Ако разглеждаме комуникационния канал между модулатора и демодулатора, на входа постъпват непрекъснати реални функции, а на изхода се получават техни "зашумени" версии. Но ако разглеждаме канала заедно с модулатора и демодулатора, то и на входа, и на изхода се появяват нули и единици, т.е. дискретни величини. Ние ще отделим внимание главно на модели на втория вид комуникационни канали.

Дефиниция 1.3.1 ***Дискретен комуникационен канал*** *наричаме статистически модел с вход случайна величина U, приемаща стойности от крайната азбука $A = \{a_1, a_2, \ldots, a_m\}$ и изход случайна величина V, приемаща стойности от крайната азбука $B = \{b_1, \ldots, b_n\}$ и описващ се със стохастичната матрица $\mathbf{P}=(p_{ij})$, където*

$$p_{ij} \;=\; \Pr(V = b_j \,|\, U = a_i),\;\; 1 \le i \le m,\, 1 \le j \le n, \qquad \textit{и} \qquad \sum_{j=1}^{N} p_{ij} = 1.$$

A и B се наричат съответно ***входна и изходна азбука*** *на канала.*

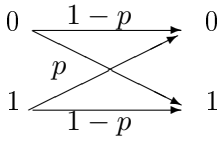
Обикновено $A = B$, но по принцип те могат и да са различни. Също така най-често за азбука се избира крайно поле $\mathbb{F} = GF(q)$ с q елемента. Най-простия случай $q = 2$, т.е. $\mathbb{F} = \{0, 1\}$ е и най-използван в практиката.

Нека $u = u_1, u_2, \ldots$ и $v = v_1, v_2, \ldots$ са входна и съответната ѝ изходна редица, т.е. U приема последователно стойности $u_1, u_2, \ldots$ в резултат на което V приема вторите стойности. Ако вероятностите $\Pr(v_k = b_j \,|\, u_k = a_i)$ не зависят от това какви са стойностите на $u_i,\ i < k$, (т.е. вероятностите p_{ij} зависят само от постъпващия в дадения момент символ, а не от предходните), то каналът се нарича **дискретен канал без памет (DMLC-discrete memoryless channel)**.

Ето някои примери на дискретни канали без памет:

1. ***Двоичен симетричен канал*** (BSC)

Нека $A = B = GF(2)$ и p е вероятността 0 и 1 да се трансформират в 1 и 0, съответно. В такъв случай BSC се описва чрез диаграмата и стохастичната матрица дадени на фигура 1.2.



$$P = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

Фигура 1.2: Двоичен симетричен канал.

2. ***Двоичен симетричен канал с изтриване*** (Binary erasure channel - BEC)

При цифровите апаратури, за всеки получен бит, приемащото устройство трябва да реши дали той е 0 или 1. Системи, в които такова решение винаги се взема се наричат *системи с твърдо решение (твърдо декодиране)* (*hard decision systems*). Понякога, обаче, е по-добре да се постави "?" вместо да се решава на всяка цена дали е получено 0 или 1. Символът "?" се нарича *изтриване*, а системите реализиращи тази идея - *системи с меко решение (меко декодиране)* (*soft decision system.*) За BEC входната азбука е $A = GF(2) = \{0, 1\}$, а изходната е $B = \{0, 1, ?\}$. Разглеждат се два модела на двоичен симетричен канал с изтриване, които се описват със следните стохастични матрици:

$$P = \begin{pmatrix} 1-p & 0 & p \\ 0 & 1-p & p \end{pmatrix},$$

където p е вероятността 0 (или 1) да бъде заместена с "?" и

$$P = \begin{pmatrix} 1-p-q & q & p \\ q & 1-p-q & p \end{pmatrix},$$

където p е вероятността 0 (или 1) да бъде заменена с "?" , а q вероятноста 0 (или 1) да се преобразува в 1 (съответно 0).

Капацитет (пропускателна способност) на канал.

Нека е даден дискретен канал без памет с входна и изходна азбуки A и B и стохастична матрица $\mathbf{P} = (p_{ij})$. Нека постъпващите на входа елементи

на азбуката A се генерират от дискретен източник без памет $\mathfrak{A}$, зададен с вероятностното разпределение

$$p_k = \Pr(U = a_k),\ k = 1, 2, \ldots, m.$$

Тогава изходът на канала може да се разглежда като дискретен източник без памет $\mathfrak{B}$, с вероятностно разпределение $(q_1, q_2, \ldots, q_n)$, където

$$q_j = \sum_{i=1}^{m} p_i \Pr(V = b_j \,|\, U = a_i) = \sum_{i=1}^{m} p_i p_{ij}.$$

Съгласно Дефиниция 1.1.11 и (1.2) информацията

$$I(\mathfrak{A}|\mathfrak{B}) = H(\mathfrak{A}) - H(\mathfrak{A}|\mathfrak{B}) = H(\mathfrak{A}) + H(\mathfrak{B}) - H(\mathfrak{A}, \mathfrak{B})$$

зависи само от разпределението $(p_1, p_2, \ldots, p_n)$ и матрицата на канала $\mathbf{P}$. Наистина за необходимите за изчисляване на $H(\mathfrak{A}, \mathfrak{B})$ вероятности имаме

$$\Pr(U = a_i, V = b_j) = \Pr(V = b_j \,|\, U = a_i) \Pr(U = a_i) = p_i p_{ij}. \tag{1.3}$$

Дефиниция 1.3.2 ***Капацитет (пропускателна способност) на канал*** *се нарича стойността C зададена с*

$$C = \sup I(\mathfrak{A}|\mathfrak{B}),$$

където супремумът е взет по всички разпределения $(p_1, p_2, \ldots, p_n)$. (Тъй като това множество е компакт и функцията е непрекъсната, всъщност се взема максимумът.)

Теорема 1.3.3 *Пропускателната способност на двоичен симетричен канал с вероятност за грешка p се задава с*

$$C = 1 - H(p) = 1 + p \log_2 p + (1 - p) \log_2 (1 - p).$$

Доказателство. Нека положим $q = 1 - p$ и ад означим с α и $\beta = 1 - \alpha$ вероятностите източника $\mathfrak{A}$ да дава 0 и 1, съответно. Тогава на изхода на канала получаваме 0 и 1, съответно с вероятности

$$q_1 = \alpha q + \beta p \quad \text{и } q_2 = \alpha p + \beta q.$$

Уравнения (1.3) ни дават

$$\Pr(U = 0, V = 0) = \alpha p, \quad \Pr(U = 0, V = 1) = \alpha q,$$

$$\Pr(U=1,V=0)=\beta p, \quad \Pr(U=1,V=1)=\beta q.$$

Следователно

$$H(\mathfrak{A},\mathfrak{B})=-\alpha p\log_2(\alpha p)-\alpha q\log_2(\alpha q)-\beta p\log_2(\beta p)-\beta q\log_2(\beta q)$$

$$H(\mathfrak{B})=-(\alpha q+\beta p)\log_2(\alpha q+\beta p)-(\alpha p+\beta q)\log_2(\alpha p+\beta q)$$

$$H(\mathfrak{A})=-\alpha\log_2\alpha-\beta\log_2\beta,$$

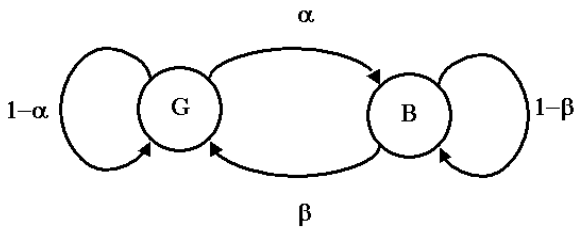
Тогава

$$\begin{aligned} I(\mathfrak{A}|\mathfrak{B}) = & \\ - \quad & \alpha\log_2\alpha-\beta\log_2\beta+\alpha p\log_2\alpha+\alpha p\log_2 p+\alpha q\log_2\alpha \\ + \quad & \alpha q\log_2 q+\beta p\log_2\beta+\beta p\log_2 p+\beta q\log_2\beta+\beta q\log_2 q+H(\mathfrak{B}) \\ = \quad & \alpha p\log_2 p+\alpha q\log_2 q+\beta p log_2 p+\beta q\log_2 q+H(\mathfrak{B}) \\ = \quad & p\log_2 p+q\log_2 q+H(\mathfrak{B}). \end{aligned}$$

Но $H(\mathfrak{B})$ достига максимум при $\alpha q+\beta p=\alpha p+\beta q=1/2$, което е еквивалентно с $\alpha=\beta=p=q=1/2$.

Дискретен канал с памет (**Discrete channel with memory.**)

В реалните канали грешките не са независими, а най-често се групират в пакети. Един прост модел на тази ситуация се явява *Gilbert 2-state* модел. Той се характеризира с две състояния: G - "добро" (gap state) и B - "лошо" (burst state) и вероятности $\alpha:\ G\to B$ и $\beta:\ B\to G$. Във всяко състояние каналът може да се разглежда като двоичен симетричен канал с $p=p_1\approx 0$ при състояние G и $p=p_2\approx 1/2$ при B. Например, ако $\alpha=10^{-6}$ и $\beta=0.056$, то средната дължина на пакета е $1/\beta\approx 18$ бита и състоянието G трае $1/\alpha=10^6$ бита



Фигура 1.3: Модел на Джилберт-Елиот.

По-сложните модели се базират на Марковски процеси от ранг ≥ 2.

Непрекъснати канали

Нека $X = X(s)$ е случайна величина приемаща реални стойности, когато s описва пространството от вероятностните събития (представляващи също реални величини). Под ***функция на разпределението*** на случайната величина X по дефиниция се разбира

$$F_X(x) \stackrel{\text{def}}{=} \Pr(X \leq x).$$

Непрекъснати случайни величини са тези, които притежават непрекъсната функция на разпределение. Тя притежава следните свойства:

1. $0 \leq F_X(x) \leq 1$
2. $F_X(x_1) \leq F_X(x_2)$, за $x_1 \leq x_2$
3. $F_X(-\infty) = 0$
4. $F_X(\infty) = 1$.

Плътност (на разпределението) (pdf) на случайната величина X се нарича функцията

$$p_X(x) \stackrel{\text{def}}{=} \frac{dF_X(x)}{dx}.$$

Името "плътност" се обуславя от следните свойства на $p_X(x)$:

1. $\Pr(x_1 < X \leq x_2) = \int_{x_1}^{x_2} p_X(x)\,dx$
2. $p_X(x) \geq 0$
3. $\int_{-\infty}^{+\infty} p_X(x)\,dx = 1.$

Дефиниция 1.3.4 *Казваме, че случайната величина X има* ***гаусово разпределение****, ако има плътност*

$$p_X(x) \stackrel{\text{def}}{=} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2\sigma^2}},$$

където m и $\sigma > 0$ са реални параметри, наричани съответно, математическо очакване и средно квадратично отклонение на X. σ^2 се нарича дисперсия.

Плътността $p_X(x)$ е симетрична по отношение на точката $x = m$, където достига и максималната си стойност:

$$\max_x p_X(x) = p_X(m) = \frac{1}{\sigma\sqrt{2\pi}}.$$

Да напомним, че математическото очакване m_X на произволна непрекъсната случайна величина X се дефинира с

$$m_X \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} x p_X(x)\,dx,$$

а дисперсията с

$$\sigma_X^2 \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} (x - m_X)^2 p_X(x)\, dx = \mathbf{E}\{(X - m_X)^2\}.$$

Произволен електрически сигнал се описва като функция $s(t)$ на времето и в много случаи тя има случаен характер - в смисъл, че за всяко t стойността ѝ $s(t)$ може да се разглежда като случайна величина зададена с някаква функция на разпределение. Такъв характер имат сигналите породени от природни източници, както и от човешката дейност (като нейни странични последици).

Дефиниция 1.3.5 *Казваме, че сигналът $n(t)$ е* ***гаусов шум****, ако има гаусово розпределение с нулево очакване, т.е. ако плътността му $p(n)$ се задава с*

$$p(n) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{n^2}{2\sigma^2}}.$$

При $\sigma = 1$ се нарича нормален (стандартен) гаусов шум.

Типичен пример за такъв шум е термалния шум, т.е. породения от топлинното движение на електроните във всички компоненти на устройствата. Този шум се характеризира и с това, че в много широк спектър от нула до 10^{12} Hz интензивността му е приблизително еднаква. По аналогия със светлината такъв шум се нарича ***бял гаусов*** и математически се характеризира с една и съща спектрална плътност $N_0/2$ (мерена в W/Hz) по всички честоти и е в сила $\sigma^2 = N_0/2$.

Ако даден канал трансформира предавания сигнал $s(t)$ в

$$r(t) = s(t) + n(t),$$

където $n(t)$ е бял гаусов шум казваме, че каналът е с ***адитивен бял гаусов шум (AWGN)***.

Понякога е удобно да се разглеждат и модели, при които входът на канала е дискретен, а изходът непрекъснат.

Глава 2

Теория на кодирането

2.1 Модулация.

Както отбелязаме във Въведението, модулацията и демодулацията са задължителни елементи за всяка комуникационна система. Интензивно изследвани в продължение на десетилетия за тях е развита обширна и дълбока теория. Макар че нейното разглеждане излиза извън рамките на този курс, за пълнота ще изложим накратко основните принципи и схеми на модулация.

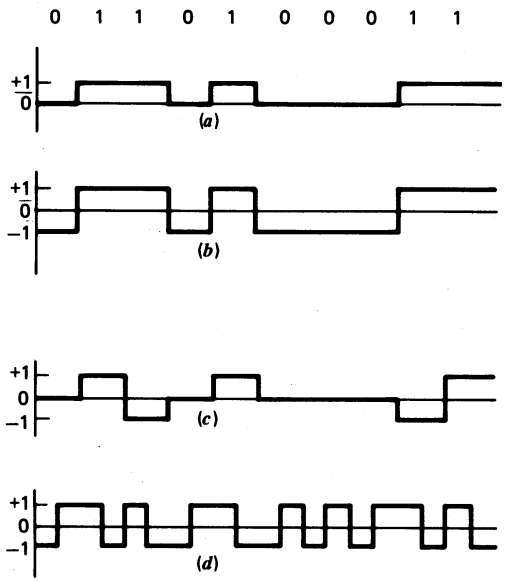
Може да разделим сигналите на два типа: ***нискочестотни (low-pass)***, чиято честота не надхвърля някаква горна граница и ***bandpass*** сигнали, чиято честота се колебае около някаква средна относително висока стойност, наречена *носеща честота*. Пример за първия вид сигнал е звуковият, а за втория излъчваният от радиостанциите.

В повечето цифрови устройства двоичните данни се представят чрез поредица от правоъгълни импулси наричани ***baseband formats***. Четири такива формата са представени на фигура 2.1, но трябва да отбележим, че съществуват десетки подобни формати. По същество те представляват нискочестотни сигнали и макар да се използват за предаване на данни дори по коаксиални и оптични кабели, те са неподходящи за пренос на големи разстояния и мултиплексиране. Затова с тях се модулират bandpass сигнали.

Типичният bandpass (в това число и модулираният) сигнал има вида:

$$s(t) = A(t)\cos[2\pi f_c(t) \cdot t + \phi(t)], \tag{2.1}$$

където $A(t)$ е ***амплитудата***, $f_c(t)$ ***честотата*** (в herz) и $\phi(t)$ ***фазата***.



Фигура 2.1: Представяне на двоични данни.

В общия случай всички те са функции на времето. Амплитудата се нарича често *естествена обвиваща* на сигнала.

Използвайки свойствата на тригонометричните функции може да запишем (2.1) във вида

$$s(t) = s_I(t)\cos(2\pi f_c t) - s_Q(t)\sin(2\pi f_c t), \tag{2.2}$$

където $s_I(t) = A(t)\cos\phi(t)$ и $s_Q(t) = A(t)\sin\phi(t)$.

Когато $s(t)$ е преставен във вида (2.2) казваме, че е зададен в ***канонична форма***. Компонентите $s_I(t)$ и $s_Q(t)$ се наричат, съответно, ***in-phase*** и ***quadrature*** компоненти, като и двете са нискочестотни (low-pass) сигнали.

Друго представянена $s(t)$ е така нареченото ***complex representation***:

$$s(t) = Re[\hat{s}(t)e^{i2\pi f_c t}], \tag{2.3}$$

където

$$\hat{s}(t) = s_I(t) + is_Q(t) = A(t)e^{i\phi(t)},$$

($e^{ix} = \cos x + i\sin x$), и $Re[.]$ означава реалната част на израза в скобите.

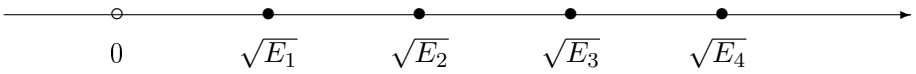
$\hat{s}(t)$ се нарича ***комплексна обвиваща (complex envelope)*** или ***low-pass еквивалент*** на $s(t)$ ($\hat{s}(t)$ е нискочестотен сигнал).

В зависимост от това коя компонента изменя нискочестотният сигнал, модулацията бива амплитудна, честотна или фазова като се ползват и всевъзможните им комбинации.

Амплитудна модулация(ASK)

$$s_i(t) = \sqrt{\frac{2E_i(t)}{T}} \cos(2\pi f_c t + \phi) \quad i = 1, \ldots, M,\ 0 \leq t \leq T,$$

където $\sqrt{2E_i(t)/T}$, приема M различни стойности, а фазовото отместване ϕ е произволна константа. Носещата честота f_c е също константа. Сигналът $s_i(t)$ се изобразява като точка върху реалната ос съответстваща на корен квадратен $\sqrt{E_i}$ от енергията му.



Честотна модулация (FSK).

$$s_i(t) = \sqrt{\frac{2E}{T}} \cos(2\pi f_i t + \phi) \quad i = 1, \ldots, M,\ 0 \leq t \leq T,$$

където f_i приема M различни стойности, а фазовото отместване ϕ е произволна константа. На практика, M е най-често степен на двойката.

Фазова модулация (PSK)

PSK е разработена в началните периоди на космическите комуникации. Понастоящем широко се използва във военни и комерсиални системи. Аналитично се задава с

$$s_i(t) = \sqrt{\frac{2E}{T}} \cos[2\pi f_c t + \phi_i(t)] \quad i = 1, \ldots, M,\ 0 \leq t \leq T,$$

където $\phi_i(t)$ приема M различни стойности, обикновено задавани с

$$\phi_i(t) = \frac{2\pi i}{M} \quad i = 1, \ldots, M.$$

E е енергията, T е времетранието за предаден символ, и $0 \leq t \leq T$.

Една схема за модулация се нарича *двоична*, ако $M = 2$ и *многостепенна (multilevel)*, за по-големи стойности на M. При $M = 2^k$ входната редица се разделя на блокове от по k бита и всеки блок се трансформира във вълнов сигнал с дължина T (наричан символ). Очевидно те са M различни.

При двоичната PSK (BPSK), модулиращия сигнал отмества фазата на $s_i(t)$ на нула или π. Такива сигнали се наричат *antipodal* .

Модулация от по-виска размерност и Quadrature-Amplitude модулация (QAM).

По-добра комуникация при дадено съотношение сигнал/шум се получава, ако k бита се изобразят в $n-$мерен сигнал $\mathbf{s}(t) = (s_1(t), \ldots, s_n(t))$. Такъв сигнал се изобразява с точка в $n-$мерното пространство и съвкупността от всички сигнали се нарича *signal constellation*.

Нека $\{\psi_j(t)\}_{j=1}^n$ е множество от n ортонормирани базисни функции, т.е.

$$\int_0^T \psi_i(t)\psi_j(t)\,dt = \delta_{ij} = \begin{cases} 1, & \text{i=j;} \\ 0, & i \neq j. \end{cases}$$

Нека някаква съвкупност от техни линейни комбинации е избрана като *множество от сигналите*. Тогава всеки сигнал

$$\mathbf{s_i}(t) = \sum_{j=1}^{n} s_{ij}\psi_j(t)$$

има енергия

$$E_i = \int_0^T \mathbf{s_i}^2(t)\,dt = \sum_{j=1}^{n}\sum_{l=1}^{n} s_{ij}s_{il}\int_0^T \psi_i(t)\psi_j(t)\,dt = \sum_{j=1} s_{ij}^2.$$

Един широко прилаган тип модулация е M-арната *quadrature amplitude modulation* (M-QAM). В този случай $n = 2$ и $i-$тия сигнал има вида

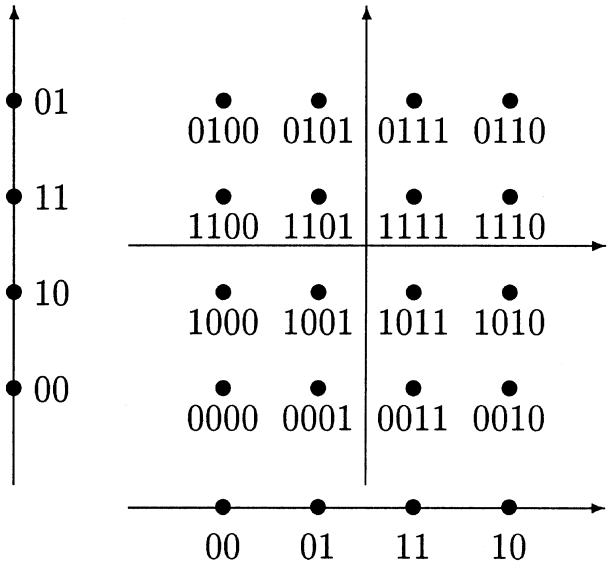
$$\mathbf{s}_i(t) = (s_I(t), s_Q(t)), \quad \text{където}$$

$$s_I(t) = a_i\sqrt{\frac{2}{T}}\cos\omega_0 t, \qquad s_Q(t) = b_i\sqrt{\frac{2}{T}}\sin\omega_0 t$$

се наричат *in-phase* (I) и *quadrature* (Q) компонента, съответно. Най-често $M = 2^k$, а реалните числа a_i и b_i принадлежат на множеството

$$\{\pm d, \pm 3d, \ldots, \pm(M/4 - 1)d\},$$

където d е половината разстояние между две съседни точки.



Фигура 2.2: Съзвездие за 16-QAM модулация с код на Грей.

2.2 Кодове контролиращи грешки - въведение.

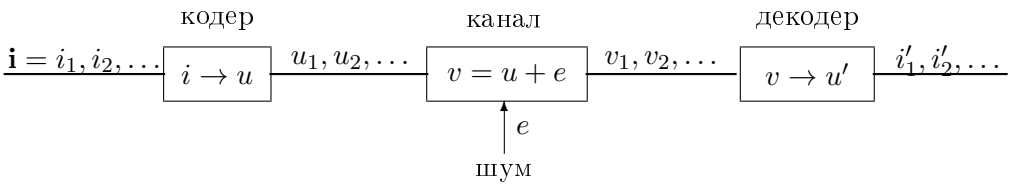
В §1.3 въведохме понятието пропускателна способност на комуникационен канал и разгледахме някои модели на такива канали. Както отбелязахме предаването по зашумен канал може да се опише със следната абстрактна схема дадена на фигура 2.3:

Обикновено $A = B = I$, но по принцип те могат и да бъдат различни. Също така най-често за азбука се избира крайно поле $\mathbb{F} = GF(q)$ с q елемента. Най-простия случай $q = 2$, т.е. $\mathbb{F} = \{0, 1\}$ е и най-използван в практиката.

Всяка комуникационна система освен със споменатата по-горе пропускателна способност на канал за предаване (или съхранение) на данни се характеризира и със следните параметри:

-*скорост на предаване на символ* (*bound rate*): измерва се в символ/секунда и се дефинира като $1/T$, където T е времето за предаване на един символ.

-*информационна скорост (скорост на кода) (information rate)*: дефинира се като отношението $R = k/n$ на информационните битове към изпратените по канала битове за осъществяване на предаването. Информационната скорост показва, каква част от информационния символ се предава с един



$i_\nu \in I$ - азбука на източника на информация;
$u_j \in A$ - входна азбука на канала;
$v_j \in B$ - изходна азбука на канала.

Фигура 2.3: Математически модел на комуникационна система.

бит по канала

-*скорост на предаване на данни (data rate)*: дефинира се като R/T и се измерва в битове/секунда (bps).

Съществуват два основни типа кодове за контрол на грешките при предаване или съхранение на информацията:

- ***Блоков код***
- ***Конволюционен код (convolutional code)***.

При първия тип кодове информационната редица се разделя на блокове с равна дължина (например от по k символа) и всеки от тях се обработва независимо. Кодерът заменя информационния блок $\mathbf{i}= (i_1, \ldots, i_k)$ с n-орка $\mathbf{u}= (u_1, \ldots, u_n)$ от символи на канала.

Кодерът при конволюционните кодове оперира с информационната поредица без да я разбива на независими блокове. По-точно, за да я обработи той също я разделя на блокове от по k символа, но изходната n-орка зависи и от предходните информационни блокове.

Конволюционните кодове, също както блоковите са били обект на интензивно изследване през последните десетилетия от много математици и в частност от специалисти по алгебра, но в настоящето изложение ние ще се ограничим само с блокови кодове.

2.2.1 Основни понятия и твърдения.

Нека $\mathbb{F} = GF(q)$ е крайното поле с q елемента и $\mathbb{F}^n$ е n-мерното векторно пространство над $\mathbb{F}$, т.е.

$$\mathbb{F}^n = \{(x_1, \ldots, x_n) \mid x_i \in \mathbb{F}\}$$

Дефиниция 2.2.1 *Всяко подмножество $\mathcal{C}$ на $\mathbb{F}^n$ се нарича* ***q-ичен блоков код с дължина*** *n. Линейните подпространства се наричат съответно* ***линейни кодове*** *с дължина n.*

Елементите на един код $\mathcal{C}$ се наричат ***кодови думи*** и техният брой се бележи с $M = |\mathcal{C}|$. За всеки блоков код $\mathcal{C}$ с дължина n и $M = |\mathcal{C}|$ кодови думи ще казваме, че $\mathcal{C}$ е (n, M) код.

Ако $\mathcal{C}$ е линеен код и има размерност k като линейно пространство, то очевидно мощността му е $|\mathcal{C}| = q^k$. За такъв код ще казваме, че е $[n, k]$ код, а k ще наричаме ***размерност*** на кода $\mathcal{C}$.

Дефиниция 2.2.2 ***Скорост (на предаване на информация)*** *на q-ичния код $\mathcal{C}$ се дефинира като отношението*

$$R = \log_q |C|/n.$$

В линейния случай $R = k/n$.

Най-прости примери на блокови кодове.

Пример 2.2.1 *Код с обща проверка на четност* (код с голяма скорост, но лоши коригиращи способности):

Към всеки k информационни символа $(a_1, \ldots, a_k)$, $a_i \in F$ добавяме $(k+1)$-и символ a_{k+1} така, че $a_1 + a_2 + \cdots + a_{k+1} = 0$ (сумирането е в $\mathbb{F}$). Полученият по този начин код е линеен $[k+1, k]$ код. Например, при $k = 7$, $\mathbb{F} = GF(2)$ се получава $[8, 7]$ двоичен код

Пример 2.2.2 *Код с просто повторение* (твърде ниска скорост, но добри коригиращи способности):

Всеки информационен символ се повтаря n пъти (обикновено n е нечетно). Например, ако $n = 5$, то $0 \to 00000$, и $1 \to 11111$. Очевидно, кодът с просто повторение с дължина n е линеен $[n, 1]$ код и има скорост $R = 1/n$.

Пример 2.2.3 *Двоичен $[6, 3]$ код*:

Към всеки блок от 3 информационни бита (a, b, c) се добавят 3 нови:

$$x = a + b, \;\; y = a + c, \;\; z = b + c$$

Например $011 \to 011110$, $100 \to 100110$. Броят на кодовите думи в получения код е $|\mathcal{C}| = 2^3 = 8$ и лесно се проверява, че $\mathcal{C}$ е линеен със скорост $R = 1/2$.

Пример 2.2.4 [7,4] *Двоичен код на Хеминг* $\mathcal{H}_3$:

Този код принадлежи към семейство кодове предложено от Хеминг (Hamming) в 1957 г., което се явява исторически първият клас нетривиални кодове изправящи грешки. Откриването му може да се счита за рождена дата на теория на кодирането като самостоятелна дисциплина.
Двоичният $[7,4]$ код на Хеминг се конструира като към всеки 4 информационни бита (i_1, i_2, i_3, i_4) се добавят три проверочни (p_1, p_2, p_3) определени по правилото

$$p_1 = i_1 + i_2 + i_3$$
$$p_2 = i_2 + i_3 + i_4$$
$$p_3 = i_1 + i_2 + i_4$$

Полученият код е линеен код със скорост $R = 4/7$.

Кодовете контролиращи грешки се използват за две цели, едновременно или по отделно:

- откриване на грешки

- изправяне на грешки.

В режим на откриване на грешки декодерът установява кога се е появила грешка или с отнапред зададедена вероятност (≈ 1) приема, че полученият блок $\mathbf{v}$ съвпада с изпратената кодова дума $\mathbf{u}$. Кодът в пример 2.2.1 е типичен код за откриване на грешки.

Когато кодът се използва за изправяне на грешки, след откриването на грешка декодерът може да процедира по два начина:

- пълно декодиране: Всеки получен $n-$мерен вектор $\mathbf{v}$ се декодира в кодова дума, от която е най-вероятно да е бил получен;

- непълно декодиране: В този случай се декодират само тези от получените вектори $\mathbf{v}$, за които вероятността от грешно декодиране е под някаква отнапред зададена стойност (≈ 0). За останалите декодера дава "отказ от декодиране" и игнорира съобщението, или иска неговото повторение. Прилага се в системи, където достоверността на информацията е много съществена и се предпочита нейната загубата пред изкривяването ѝ и/или когато е възможно препредаване ѝ .

Правило за декодиране (и в двата му варианта), при което декодера изобразява получената поредица $\mathbf{v}$ в кодова дума $\mathbf{c}$, за която е най-вероятно да се предполага, че е била изпратена, се нарича ***декодиране по максимално правдоподобие*** (***maximum - likelihood (ML) decoding rule***.) При него получената дума $\mathbf{v}$ се декодира в такава кодова $\mathbf{c}$, която максимизира вероятността $\Pr(\mathbf{v}$ получено $|\ \mathbf{c}$ изпратено$)$.

Правилото за декодиране по максимално правдоподобие се основава на предположението, че всички кодови думи са равновероятни.

Нека разгледаме код $\mathcal{C}$ над полето $\mathbb{F}$ с правило за декодиране (използва се и терминът схема за декодиране) f, т.е. зададено с изображение

$$f : \left| \begin{array}{lcl} \mathbb{F}^n & \to & \mathcal{C} \\ v & \to & f(v) = c \end{array} \right.$$

При двата режима на декодиране е изпълнено съответно:

- при пълно декодиране: f е дефинирано за всяко $v \in \mathbb{F}^n$;

- при непълно декодиране: в този случай областта от значения на f е обединението на $\mathcal{C}$ със специален символ означаващ "отказ от декодиране" и за някои вектори v този символ е стойността на функцията f.

Нека $p(E|c) = \Pr(\text{грешно декодиране} \mid u := c)$ е вероятността от грешно декодиране при избраната схема за декодиране и при условие, че е изпратена кодовата дума c, т.е. $u := c$.

Дефиниция 2.2.3 *Под* ***вероятност за грешно декодиране на дума (word error rate)*** *се разбира*

$$P_e \stackrel{\text{def}}{=} \sum_{c \in C} p(c) p(E|c),$$

където $p(c)$ е вероятността за изпращане на кодовата дума c.

Най често кодовите думи са равновероятни (или се приема, че е така), т.е. $p(c) = 1/M$, където M е броя на кодовите вектори. В такъв случай

$$P_e = \frac{1}{M} \sum_{c \in C} p(E|c) = \frac{1}{M} \sum_{i=1}^{M} \Pr(\text{грешно декодиране} \mid c_i \text{ изпратено})$$

За да се опишат по-добре възможностите на даден код да открива и изправя грешки в $n-$мерното векторно пространство $\mathbb{F}^n$ се въвежда метрика.

Дефиниция 2.2.4 *Под* ***Хемингово разстояние*** $d(x, y)$ *между два вектора x и y от $\mathbb{F}^n$ разбираме броя на позициите, в които те се различават:*

$$d(x, y) \stackrel{\text{def}}{=} |\{i | x_i \neq y_i\}|.$$

Дефиниция 2.2.5 *Нека $\mathcal{C} \subset \mathbb{F}^n$ е код с дължина n.* ***Минимално разстояние*** *$d(\mathcal{C})$ на $\mathcal{C}$ се нарича най-малкото измежду хеминговите разстояния между две различни кодови думи, т.е.*

$$d(\mathcal{C}) \stackrel{\text{def}}{=} \min_{x \neq y \in \mathcal{C}} d(x, y),$$

където минимумът е взет по всички двойки различни кодови думи на $\mathcal{C}$.

Лесно се проверява, че хеминговото разстояние $d(x, y)$ е метрика, т.е.

(i) $d(x, y) \geq 0$ като равенство се достига при $x = y$
(ii) $d(x, y) = d(y, x)$
(iii) $d(x, y) \leq d(x, z) + d(y, z)$ (неравенство на триъгълника)

Дефиниция 2.2.6 ***Хеминогово тегло*** $\mathrm{wt}(c)$ *на кодова дума $c \in \mathcal{C}$ наричаме броя на ненулевите ѝ позиции, а под* ***минимално тегло*** $\mathrm{wt}(\mathcal{C})$ *на $\mathcal{C}$ се разбира теглото на най-леката ненулева кодова дума.*

Очевидно, в сила е $\mathrm{wt}(c) = d(c, o)$ и ако $\mathcal{C}$ е линеен, то $d(C) = \min d(c_i, c_j)$ $= \min d(o,\ c_i - c_j) = wt(C)$.

При двоичен симетричен канал един естествен подход е декодирането да се извърши по ***правилото на най-близката кодова дума***: полученият вектор $\mathbf{v}$ се декодира в кодовата дума $\mathbf{c}$, която е на най-малко хеминогово разстояние от него.

Твърдение 2.2.7 *За двоичен симетричен канал с $p < 1/2$ декодирането по максимално правдоподобие е еквивалентно на декодиране по правилото на най-близката кодова дума.*

Доказателство. За всеки $x, y \in \mathbb{F}^n$ с $d(x, y) = d$ е в сила

$$\Pr(y \text{ получено } \mid x \text{ изпратено}) = p^d(1 - p)^{n-d}$$

При $p < 1/2$, вероятността е максимална, когато d е минимално. $\diamond$

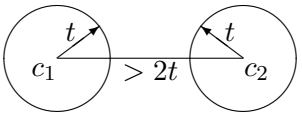
Дефиниция 2.2.8 *Казваме, че кодът $\mathcal{C}$* ***открива до t грешки****, ако при промяна от една до t позиции в коя и да е кодова дума не може да се получи друга кодова дума.*

Дефиниция 2.2.9 *Казваме, че кодът $\mathcal{C}$* ***изправя*** *t* ***грешки****, ако при декодиране по правилото на най-близката кодова дума може да се поправи всяка конфигурация от най-много t грешки. Ако $\mathcal{C}$ изправя t грешки, но има поне един вектор-грешка с тегло $t+1$, който не може да бъде определен, то $\mathcal{C}$ наричаме* ***точно*** *t* ***грешки коригиращ****.*

Ясно е, че $\mathcal{C}$ изправя t грешки, точно когато за всеки вектор $v \in F^n$, съществува най-много една кодова $c \in C$, такава че $d(v,c) \leq t$.

Теорема 2.2.10 *Кодът $\mathcal{C}$ изправя t грешки тогава и само тогава, когато $d(\mathcal{C}) \geq 2t+1$ и открива t грешки тогава и само тогава, когато $d(\mathcal{C}) \geq t+1$.*

Доказателство. Нека $d(\mathcal{C}) \geq 2t+1$ и $v \in \mathbb{F}^n$. Да предположим, че съществуват $c_1,\ c_2 \in \mathcal{C}$, такива че $d(c_1,v) \leq t, \quad d(c_2,v) \leq t$. Тогава неравенството на триъгълника (iii) дава $d(c_1,c_2) \leq t+t = 2t$, което противоречи на $d(\mathcal{C}) > 2t$.



Обратно, нека $\mathcal{C}$ е код поправящ t грешки и да предположим, че $d(\mathcal{C}) \leq 2t$. Тогава съществуват $c_1, c_2 \in C$ с $d(c_1,c_2) \leq 2t$ и лесно се построява $v \in \mathbb{F}^n$, такова че $d(v,c_1) = t,\ d(v,c_2) \leq t$, което разбира се е невъзможно.

$\diamond$

Накрая на този параграф ще дадем без доказателство теоремата на Шенон за кодиране при наличие на шум в дискретен канал без памет.

Теорема 2.2.11 *Ако е даден дискретен канал без памет с капацитет C и $R < C$ е произволно положително число, то съществува редица от кодове $\{\mathcal{C}_n \mid 1 \leq n < \infty\}$, такива че:*

(1) $\mathcal{C}_n$ е с блокова дължина n и има $\lfloor 2^{nR} \rfloor$ кодови думи;

(2) Вероятността за грешно декодиране $P_e(\mathcal{C}_n)$ удовлетворява неравенството

$$P_e(\mathcal{C}_n) \leq Ae^{-Bn},$$

където A и B зависят само от канала и R.

Тази теорема показва, че "добри" кодове съществуват и остава теорията на кодирането да ги намери.

2.3 Линейни кодове.

Нека $\mathcal{C}$ е $[n,k]$ q-ичен линеен код.

Дефиниция 2.3.1 ***Пораждаща матрица*** *на $\mathcal{C}$ наричаме всяка $k \times n$ матрица, чиито редове образуват базис на $\mathcal{C}$ като линейно пространство.*

Ако $\mathbf{G}$ е пораждаща матрица на $\mathcal{C}$, то

$$\mathcal{C} \equiv \{a.\mathbf{G} \mid a \in \mathbb{F}^k\} \tag{2.4}$$

Очевидно, (2.4) задава съответствие между блока от k информационни символа и кодовата дума, в която той се кодира, т.е. задава правило за кодиране. Варирайки пораждащите матрици получаваме различни правила за кодиране (наричани още ***кодери***) посредством кода $\mathcal{C}$.

Казваме, че $\mathbf{G}$ е в ***стандартна (систематична) форма***, ако $\mathbf{G} = (\mathbf{I}_k \mid \mathbf{P})$, където $\mathbf{I}_k$ е $k \times k$ единичната матрица и $\mathbf{P}$ е $k \times (n-k)$ матрица.

Ако $\mathbf{G}$ е в стандартна форма, то първите k символа на всяка кодова дума са точно информационния блок $\mathbf{i}$ и се наричат ***информационни символи***. Останалите символи се наричат ***проверочни*** и се определят еднозначно от информационните.

Дефиниция 2.3.2 *Кодът $\mathcal{C}$ се нарича* ***систематичен в k позиции*** *(и символите в тези позиции се наричат информационни), ако $|\mathcal{C}| = q^k$ и съществува точно една кодова дума за всеки избор на стойностите в тези k позиции, т.е. всяка k-орка $\mathbf{i} \in \mathbb{F}^k$ се среща точно веднъж в тези k координати. Процедура за кодиране, която изобразява k-орката $\mathbf{i} = (i_1, \ldots, i_k)$ в кодова дума с точно тази k-орка в информационните координати, се нарича* ***систематично кодиране***.

Например, много по-удобно е пораждащата матрица на цикличен код да се използва във вида $\mathbf{G} = (\mathbf{P}\ \mathbf{I}_k)$ и тази форма се нарича разбира се систематична.

Примери:

Пример 2.2.1 (Код с обща проверка на четност):

$$\mathbf{G} = \begin{pmatrix} 1 & & & 1 \\ & 1 & & 1 \\ & & \ddots & \vdots \\ & & 1 & 1 \end{pmatrix}.$$

Пример 2.2.2 (Код с просто повторение):

$$\mathbf{G} = (1 \underbrace{1 \ldots 1}_{n-1}).$$

Пример 2.2.3 ([6, 3] двоичен код):

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

Пример 2.2.4 ([7,4] код на Хеминг):

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

Упражнение 2.3.1 *Напишете всички кодови думи на троичния код с пораждаща матрица* $\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 2 & 1 & 0 & 2 \end{pmatrix}$ *и стандартната форма на* G.

Нека $x \circ y = x_1y_1 + \cdots + x_ny_n$, където операциите са в $\mathbb{F}$, е ***вътрешното произведение*** на два вектора $x, y \in \mathbb{F}^n$. Ако то е нула ще казваме, че векторите са ***ортогонални.***

Дефиниция 2.3.3 ***Дуален код*** *на* $\mathcal{C}$ *(бележим с* $\mathcal{C}^\perp$*) се нарича множеството от всички вектори на* $\mathbb{F}^n$, *които са ортогонални на всяка кодова дума от* $\mathcal{C}$. *т.е.*

$$\mathcal{C}^\perp = \{v \in \mathbb{F}^n \mid v \circ c = 0, \ \forall c \in \mathcal{C}\},$$

Ясно, че $\mathcal{C}^\perp$ е точно дуалното на $\mathcal{C}$ пространство в $\mathbb{F}^n$ и както е добре известно от линейната алгабра $\mathcal{C}^\perp$ се явява линеен код (линейно пространство) с размерност $n - k$, т.е. $\mathcal{C}^\perp$ е $[n, n - k]$ код.

Дефиниция 2.3.4 *Всяка пораждаща матрица* $\mathbf{H}$ *на* $\mathcal{C}^\perp$ *се нарича* ***проверочна матрица*** *на* $\mathcal{C}$.

От $x \circ c = 0$ за всяко $x \in \mathcal{C}^\perp$, следва, че

$$\begin{aligned} &c\mathbf{H}^\tau = o \quad \forall c \in \mathcal{C} \\ &\mathbf{G}\mathbf{H}^\tau = O \\ &\mathbf{G}x^\tau = o \ (x\mathbf{G}^\tau = o) \ \ \forall x \in \mathcal{C}^\perp \end{aligned}$$

за всяка пораждаща и проверочна матрица на C. По традиция уравненията ($(n-k)$ на брой) зададени с $c\mathbf{H}^\tau = o$ се наричат ***проверочни.***

Примери: Ако $\mathcal{C}$ е двоичния $[6,3]$ код от пример 2.2.3, то проверочните му уравнения са

$$\left|\begin{array}{l} a+b+x=0 \\ a+c+y=0 \\ b+c+z=0 \end{array}\right.$$

и следователно съответната им проверочна матрица ще има вида

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

Твърдение 2.3.5 *Ако $\mathcal{C}$ е систематичен код с пораждаща матрица $\mathbf{G} = (\mathbf{I}_k\mathbf{P})$, то $(n-k)\times n$ матрицата $\mathbf{H} = (-\mathbf{P}^\tau\ \mathbf{I}_{n-k})$ е проверочна матрица за C.*

> ***Доказателство.*** Всички редове на $\mathbf{H}$ са линейно независими и директната проверка чрез умножаване на матриците доказва твърдението.
>
> $\diamond$

Пример 2.3.1 Проверочната матрица на $[7,4]$ кода на Хеминг (пример 2.2.4) е

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

Упражнение 2.3.2 *Намерете проверочни матрици за всички кодове от дадените по-горе примери.*

Дефиниция 2.3.6 *Кодът $\mathcal{C}$ се нарича **самодуален**, ако $\mathcal{C}^\perp \equiv C$.*

Самодуалните кодове са били и са предмет на интензивни изследвания, за които се привлича разнообразен и мощен математически апарат като теорията на инвариантите и др.

Теорема 2.3.7 *Кодът $\mathcal{C}$ има минимално разстояние $\mathrm{wt}(\mathcal{C}) = w$ тогава и само тогава, когато всеки $w-1$ стълба на коя и да е негова проверочна матрица $\mathbf{H}$ са линейно независими.*

Доказателство. Нека $c = (c_1, \ldots, c_n) \in C$ и $\overline{h}_j$ е j-я стълб на $\mathbf{H}$. Доказателството следва непосредствено от равенствата

$$c\mathbf{H}^\tau \;=\; 0 \iff c_1\overline{h}_1 + c_2\overline{h}_2 + \cdots + c_n\overline{h}_n \;=\; 0$$

Те показват, че на всяка кодова дума $c \in \mathcal{C}$ с тегло $\mathrm{wt}(c) = w$ съответстват w линейно зависими стълба и обратно. Следователно минималното тегло $\mathrm{wt}(C) = w$ тогава и само тогава, когато никоя нетривиална линейна комбинация на $\leq w-1$ стълба не се анулира.

◇

Следствие 2.3.8 *Кодът $\mathcal{C}$ изправя до t грешки тогава и само тогава, когато всеки $2t$ стълба на негова проверочна матрица са линейно независими.*

Теорема 2.3.9 (Граница на Синглитън). *Минималното разстояние на всеки линеен $[n, k]$ код удовлетворява неравенството*

$$d(\mathcal{C}) \leq n - k + 1. \tag{2.5}$$

Доказателство. Съгласно теорема 2 рангът $\mathrm{rank}(\mathbf{H})$ на проверочната матрица $\mathbf{H}$ е поне $d(\mathcal{C})-1$. От друга страна $\mathrm{rank}(\mathbf{H}) = n - k$, което доказва неравенството.

◇

Дефиниция 2.3.10 *Линеен код достигащ границата на Синглитън, т.е. когато се достига равенство, се нарича* ***код с достижимо максимално разстояние*** *или, за краткост,* MDS *код.*

MDS кодовете притежават редица хубави свойства и затова са предмет на интензивни изследвания.

Нека $\pi \in S_n$ е пермутация на елементите $\{1, 2, \ldots, n\}$. Изображението на $\mathbb{F}^n$ върху себе си дефинирано с $\pi :\ \mathbf{c} \to \mathbf{c}'$, където $c'_i = c_{\pi(i)}, \quad 1 \leq i \leq n$ за всяко $\mathbf{c} \in \mathbb{F}^n$ ще наричаме ***пермутация на позициите***.

Дефиниция 2.3.11 *Два кода над $GF(q)$ се наричат* ***еквивалентни****, ако се получават един от друг с пермутация на позициите и/или умножение на елементите на дадена(и) позиция(и) на всяка кодова дума с ненулев елемент на полето. Ако само пермутация на позициите е достатъчна за да се изобрази единия код в другия те се наричат* ***изоморфни****. Всеки изоморфизъм на един код $\mathcal{C}$ върху себе си се нарича* ***автоморфизъм****,*

а съвкупността от всички автоморфизми на $\mathcal{C}$ (която е група относно композицията на изображения) - ***група от автоморфизмите на*** *$\mathcal{C}$. Бележи се с $Aut(\mathcal{C})$.*

При $q = 2$, очевидно, понятията изоморфни и еквивалентни съвпадат.

Както е добре известно от линейната алгебра, всяка матрица може да се приведе чрез елементарни операции по редове и разместване на стълбове към стандартна форма (трапецовидна - с нулеви последни редове и единици разположени по диагонала, явяващи се първи елементи на ненулевите редове). Следователно, всеки линеен код е еквиалентен на систематичен. Нещо повече, всеки $[n, k]$ код е систематичен най-малко в една k-орка от позиции (ако за трансформацията ползваме само операции по редове).

Дефиниция 2.3.12 *Нека $\mathcal{C}$ е линеен $[n, k]$ код с проверочна матрица $\mathbf{H}$. За всяко $\mathbf{v} \in \mathbb{F}^n$, произведението $\mathbf{s} = \mathbf{v}\mathbf{H}^\tau$ се нарича* ***синдром*** *на $\mathbf{v}$.*

Да отбележим, че синдромът е вектор-ред с дължина $(n-k)$ и е нулев тогава и само тогава, когато $\mathbf{v}$ е кодова дума.

Нека $\mathcal{C} = \{\mathbf{c}_1{=}o, \mathbf{c}_2, \ldots, \mathbf{c}_M\}$ е q-ичен линеен $[n, k]$ код. Да разгледаме множеството от съседните класове на $\mathbb{F}^n$ по $\mathcal{C}$, т.е. множеството

$$\mathbf{v} + \mathcal{C} = \{\mathbf{v} + \mathbf{c} \mid \mathbf{c} \in \mathcal{C}\}.$$

Известно е, че различните съседни класове са непресичащи се множества, $|\mathbf{v} + \mathcal{C}| = |\mathcal{C}| = M = q^k$ и техният брой е равен на q^{n-k}. ***Лидер*** на съседния клас $\mathbf{v} + \mathcal{C}$ се нарича вектор с минимално тегло в съседния клас. (Ако има няколко вектора с минимално тегло за лидер се фиксира кой и да е от тях.)

Нека сега да разположим всички вектори на $\mathbb{F}^n$ така както е показано в таблица 2.1 - съседните класове по $\mathcal{C}$ се разполагат в редове, започвайки с кода $\mathcal{C}$.

Този начин на подреждане на всички вектори на $\mathbb{F}^n$ се нарича ***стандартно разполагане на Слепян*** (Slepian standard array).

Твърдение 2.3.13 *Два вектора u и v на $\mathbb{F}^n$ са от един и същ съседен клас по $\mathcal{C}$ тогава и само тогава, когато имат един и същ синдром, т.е. съществува взаимно-еднозначно съответствие между съседните класове и синдромите.*

> ***Доказателство.*** u и v са от един и същ съседен клас точно тогава, когато $u - v \in C \Leftrightarrow (u - v)\mathbf{H}^\tau = 0 \Leftrightarrow u\mathbf{H}^\tau = v\mathbf{H}^\tau$.
>
> $\diamond$

Таблица 2.1: Стандартно разполагане на Слепян.

	Елементи на съседния клас				Синдроми
$o+C$	o	c_2	$\dots$	c_M	o
v_2+C	v_2	v_2+c_2	$\dots$	v_2+c_M	s_2
v_3+C	v_3	v_3+c_2	$\dots$	v_3+c_M	s_3
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
v_N+C	v_N	v_N+c_2	$\dots$	v_N+c_M	s_N
$v_{N+1}+C$	v_{N+1}	$v_{N+1}+c_2$	$\dots$	$v_{N+1}+c_M$	s_{N+1}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$v_{q^{n-k}}+C$	$v_{q^{n-k}}$	$v_{q^{n-k}}+c_2$	$\dots$	$v_{q^{n-k}}+c_M$	$s_{q^{n-k}}$

Твърдение 2.3.14 *Векторите на $\mathbb{F}^n$ с тегло $\leq t$ са в различни съседни класове по $\mathcal{C}$ тогава и само тогава, когато минималното тегло на кода $d(\mathcal{C}) \geq 2t+1$.*

Доказателство. Ще бъде едно полезно упражнение за читателя.

От казаното дотук получаваме, че при $t = \lfloor (d-1)/2 \rfloor$ (т.е. $d = 2t+1$ или $2t+2$) е изпълнено

$$N = 1 + \binom{n}{1}(q-1) + \cdots + \binom{n}{t}(q-1)^t$$

(виж стандартното разполагане на Слепян) и следователно всички съседни класове, чиито лидери са с тегло $\leq t$ са разположени над линията.

Стандартното разполагане дава прост начин за декодиране на получения вектор и илюстрира много добре разликата между пълното и непълно декодиране. При пълното декодиране декодера изобразява получения вектор v в кодовата дума над него (в първия ред на стълба съдържащ v) в стандартното разполагане. При непълното декодиране правилото е същото, но v се декодира тогава и само тогава, когато е разположено над линията в стандартното разполагане на Слепян. Ако е под линия се дава "отказ от декодиране" или някакъв друг фиксиран символ.

Използването на синдромите позволява декодирането да се опрости още повече. За тази цел записваме представителите на съседните класове в растящ ред на техните тегла като първи стълб, а съответните им синдроми

като втори стълб на една таблица (виж Таблица 2.2) Тя задава следния алгоритъм за декодиране

Таблица 2.2:

Представител	Синдром
o	o
v_2	s_2
$\vdots$	$\vdots$
v_N	s_N
v_{N+1}	s_{N+1}
$\vdots$	$\vdots$
$v_{q^{n-k}}$	$s_{q^{n-k}}$

Алгоритъм.

1) При получаване на v, се пресмята синдрома $s = v\mathbf{H}^\tau$;

2) От горната таблица се намира съответния на получения синдром лидер u;

3) v се декодира в кодовата дума $c = v - u$.

При реализация на непълно кодиране стъпка 3 се изпълнява тогава и само тогава, когато s е в горната част на таблицата (над линията). В противния случай декодера счита, че са станали повече от $t = \lfloor (d-1)/2 \rfloor$ грешки (и дава заявка за повторно предаване на данните, ако такава опция е предвидена в дадената реализация).

Лесно се вижда (покажи!), че описаният алгоритъм се явява декодиране по правилото на най-близката кодова дума.

Използването на синдромите намалява на порядък необходимите за съхранение данни - не е нужна да се съхранява цялата таблица на Слепян (2^n вектора), а само 2^{n-k} синдрома. Ако лидерите се наредят лексикографски, то отпада необходимостта от съхраняване и на таблицата от лидери на съседни класове и съответните им синдроми. Този алгоритъм се нарича ***step-by-step*** и за подробното му описание препращаме читателите към книгата на Питерсон и Уелдън [4].

Пример. При двоичния [7,4] код на Хеминг таблицата на съседните класове и синдромите им изглежда така:

Представител	Синдром
0000000	000
1000000	101
0100000	111
0010000	110
0001000	011
0000100	100
0000010	010
0000001	001
$\emptyset$	$\emptyset$

Упражнение 2.3.3 *Покажете, че при декодиране посредством стандартното разполагане вероятността за грешка при декодиране в случая на двоичен симетричен канал* **BSC** *е*

$$P_e = 1 - \sum_{j=1}^{n} \alpha_j p^j (1-p)^{n-j},$$

където α_j *е броя на лидерите на съседни класове с тегло* j.

2.4 Радиус на покритие и граници за параметрите на код.

Дефиниция 2.4.1 ***Кълбо с радиус*** r ***и център*** *с наричаме съвкупността от* $n-$*мерни вектори на разстояние най-много* r *от* c, *т.е.*

$$B(c,r) \stackrel{\text{def}}{=} \{v \in F^n \mid d(c,v) \leq r\}$$

Съвкупноста от $n-$*мерни вектори на разстояние точно* r *от* c *се нарича съответно* ***сфера***.

Дефиниция 2.4.2 ***Радиус на пакетиране на кода*** $\mathcal{C}$ *се нарича най-голямото естествено число* $t(\mathcal{C})$ *такова, че кълбата с радиус* $t(\mathcal{C})$ *около всяка кодова дума не се пресичат, т.е. те представляват* $|\mathcal{C}|$ *на брой две по две непресичащи се подмножества на* F^n.

Очевидно $t(\mathcal{C}) = \lfloor (d-1)/2 \rfloor$ и кодът коригира до $t(\mathcal{C})$ грешки. Ако $\mathcal{C}$ е линеен, то $t(\mathcal{C})$ представлява най-голямото тегло измежду теглата на лидерите на съседните класове по $\mathcal{C}$ намиращи се над чертата в стандарното разполагане на Слепян.

Дефиниция 2.4.3 ***Радиус на покритие*** *на кода $\mathcal{C}$ се нарича най-малкото естествено число $R(\mathcal{C})$ такова, че кълбата с радиус $R(\mathcal{C})$ и центрове кодовите думи покриват (т.е. съдържат) всички вектори на F^n.*

Ако $\mathcal{C}$ е линеен, то $R(\mathcal{C})$ представлява най-голямото тегло измежду теглата на лидерите на съседните класове по $\mathcal{C}$.

Теорема 2.4.4 (Граница на сферичната опаковка). *Ако $\mathcal{C}$ е (n, M, d) q-ичен код с радиус на пакетиране t, то*

$$M(1 + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \cdots + \binom{n}{t}(q-1)^t) \leq q^n.$$

Доказателство. Да разгледаме кълбата с радиус t описани около всяка кодова дума. Всяко от тези M на брой кълба съдържа по

$$\sum_{i=0}^{t} \binom{n}{i}(q-1)^i$$

вектора. Вземайки предвид, че кълбата не се пресичат и броят на векторите в $|F^n| = q^n$, получаваме неравенството. ◇

Границата на сферичната опаковка е известна още под името ***граница на Хеминг***. За линеен $[n, k, d]$-код тя изглежда така

$$\sum_{i=0}^{t} \binom{n}{i}(q-1)^i \leq q^{n-k},$$

където $t = \lfloor (d-1)/2 \rfloor$.

Дефиниция 2.4.5 *Кодовете, при които радиусите на покритие и на пакетиране съвпадат се наричат* ***съвършени кодове.*** *Код коригиращ t грешки с радиус на покритие $t+1$ се нарича* ***квазисъвършен***.

Съвършените кодове са тези, при които границата на Хеминг се достига, т.е. неравенството се обръща в равенство.

Дефиниция 2.4.6 *q-ичен код на Хеминг $\mathcal{H}_r$ се нарича $q-$ичен код с проверочна $r \times \frac{q^r-1}{q-1}$ матрица $\mathbf{H}$, имаща за стълбове по един представител на всяко едномерно подпространство на $\mathbb{F}^r$.*

Твърдение 2.4.7 *$q-$ичният код на Хеминг има параметри*

$$\left[\frac{q^r-1}{q-1},\ \frac{q^r-1}{q-1}-r,\ 3\right]$$

и е съвършен за всяко $r \geq 2$.

Доказателството оставяме като упражнение на читателя. В частност двоичният код на Хеминг има параметри $[2^r-1,\ 2^r-1-r,\ 3]$. При $r=3$ получаваме пример 2.2.4.

Дефиниция 2.4.8 *Двоичен **симплексен код** S_r наричаме дуалния код на кода на Хеминг $\mathcal{H}_r$.*

Упражнение 2.4.1 *Симплексният код S_r е $[2^r-1,\ r,\ 2^{r-1}]$ код и всяка негова кодова дума има едно и също тегло 2^{r-1}.*

Симплексният код е пример на ***константно-тегловен код***.

Теорема 2.4.9 *Нека $\mathcal{C}$ е линеен код с проверочна матрица $\mathbf{H}$. Кодът има радиус на покритие R тогава и само тогава, когато всеки ненулев вектор в F^{n-k} може да бъде представен като линейна комбинация на най-много R стълба на $\mathbf{H}$. За всяко естествено число $w \leq R$ съществува лидер на съседен клас с тегло w.*

Доказателство. Нека e е лидер на съседен клас с $\mathrm{wt}(e) = w$. Тогава синдромът $s = e\mathbf{H}^\tau$ е линейна комбинация на w стъба на $\mathbf{H}$. Обратно нека $s \in \mathbb{F}^{n-k}$ е произволен стълб с дължина $n-k$. Ще покажем, че е синдром. Наистина системата $s = x\mathbf{H}^\tau$ е съвместима, тъй като $\mathrm{rank}(\mathbf{H}) = n-k$ и следователно съществува $v \in \mathbb{F}^n$, за което $s = v\mathbf{H}^\tau$. Ако $v \in e + \mathcal{C}$, където $\mathrm{wt}(e) = w$, то s е линейна комбинация на w стъба на $\mathbf{H}$. Сега твърдението следва от факта, че R е максималното възможно тегло на лидер на съседен клас.

Нека e е лидер на съседен клас с $\mathrm{wt}(e) = w$ и $e = e' + e''$, където $\mathrm{wt}(e') = 1$. Ако означим с $\overline{h}_1, \ldots, \overline{h}_n$ стълбовете на $\mathbf{H}$, то синдромът

$$s = e\mathbf{H}^\tau = a_1\overline{h}_{i_1} + a_2\overline{h}_{i_2} + \cdots + a_w\overline{h}_{i_w}.$$

Да предположим, че e'' се получава с отстраняването на първата ненулева координата. Тогава

$$e''\mathbf{H}^\tau = a_2\overline{h}_{i_2} + \cdots + a_w\overline{h}_{i_w} = s - a_1\overline{h}_{i_1}.$$

Да допуснем, че $e'' \in e_1 + \mathcal{C}$, където $\mathrm{wt}(e_1) \leq w-2$. Тогава

$$s - a_1\overline{h}_{i_1} = e''\mathbf{H}^\tau = b_1\overline{h}_{j_1} + \cdots + b_{w-2}\overline{h}_{j_{w-2}}.$$

Следователно

$$s = a_1\overline{h}_{i_1} + b_1\overline{h}_{j_1} + \cdots + b_{w-2}\overline{h}_{j_{w-2}}$$

е линейна комбинация на $w-1$ стълба, което е противоречие с избора на e лидер с тегло w.

◇

Упражнение 2.4.2 *Докажете, че $R(C) \leq n-k$ за $[n,k]$ код.*

Нека с $A_q(n,d)$ означим максималното M, за което съществува q-ичен (n,M,d) код.

Теорема 2.4.10 (Граница на Синглитън). *За всеки n,d,q естествени числа, $n \geq d$, $q \geq 2$ е изпълнено*

$$A_q(n,d) \leq q^{n-d+1}.$$

Доказателство. Ако отстраним фиксирани $d-1$ координатни позиции от всяка кодова дума на произволен $(n,\ M,\ d)$ код, получените $M = |C|$ на брой вектори ще бъдат различни тъй като минималното разстояние е d. Вземайки предвид, че дължината им е $n-d+1$, можем да заключим, че $M \leq q^{n-d+1}$. Следователно, максимално възможният обем на код удовлетворява $A_q(n,d) \leq q^{n-d+1}$.

◇

Забележка. Ако кодът е линеен се получава точно теорема 2.3.9.

Теорема 2.4.11 (Граница на Плоткин). *Ако n,d,q са естествени числа, $n \geq d$, $q \geq 2$ с $d > \frac{q-1}{q}.n$ тогава*

$$A_q(n,d) \leq \frac{qd}{qd-(q-1)n}.$$

Доказателство. Нека C (n, M, d) q-ичен код. Да разгледаме $M \times n$ матрица състояща се от всички кодови думи и нека S е сумата от разстоянията между всеки две кодови думи. Очевидно $S \geq M(M-1)d$. От друга страна, нека j-тият символ от азбуката на кода $(0 \leq j \leq q-1)$ се среща m_j пъти в (например) първия стълб на матрицата от всички кодови думи. Тогава приносът на първия стълб в S е $\sum_{j=0}^{q-1} m_j(M - m_j)$, където $\sum_{j=0}^{q-1} m_j = M$. Използвайки неравенството на Коши-Буняковски получаваме

$$\sum_{j=0}^{q-1} m_j(M - m_j) = M^2 - \sum_{j=0}^{q-1} m_j^2 \leq M^2 - \frac{1}{q}(\sum_{j=0}^{q-1} m_j)^2 = \frac{q-1}{q}M^2$$

Тъй като тази оценка е в сила за всеки стълб, можем да заключим, че $S \leq n\frac{q-1}{q}M^2$. Следователно $(M-1)d \leq \frac{q-1}{q}nM$, което влече твърдението на теоремата.

◇

Теорема 2.4.12 (Граница на Варшамов-Джилберт). *Ако следното неравенство е в сила*

$$1 + \binom{n-1}{1}(q-1) + \binom{n-1}{2}(q-1)^2 + \cdots + \binom{n-1}{d-2}(q-1)^{d-2}) < q^{n-k},$$

то съществува q-ичен $[n, k]$ линеен код с минимално разстояние поне d.

Доказателство. Да предположим, че q, n, k, d удовлетворяват горното неравенство. Ще конструираме $(n-k) \times n$ матрица $\mathbf{H}$, на която всеки $d-1$ стълба са линейно независими. За първи стълб на $\mathbf{H}$ избираме $(n-k)$-орка от F^{n-k}. Като втори стълб можем да изберем произволна $(n-k)$-орка, която не е пропорционална на първия стълб. Изобщо, последователно избираме стълбовете на матрицата така, че всеки да не бъде линейна комбинация на $d-2$ или по-малко от предходните. В такъв случай при избора на i-тия стълб броят на $(n-k)$-орките, от които не можем да избираме е

$$N(i) = 1 + \binom{i-1}{1}(q-1) + \binom{i-1}{2}(q-1)^2 + \cdots + \binom{i-1}{d-2}(q-1)^{d-2}$$

Следователно, имаме $q^{n-k} - N(i)$ възможности за избор. Доколкото $q^{n-k} - N(i) > 0$ за $i = n$ и следователно за всяко $i \leq n$, ние можем успешно да конструираме проверочната матрица $\mathbf{H}$.

◇

Следствие 2.4.13 *$A_q(n,d) \geq q^{k_0}$, където k_0 е максималното естествено число k удовлетворяващо*

$$q^k < \frac{q^n}{\sum_{i=0}^{d-2} \binom{n-1}{i}(q-1)^i}.$$

Упражнение 2.4.3 *Покажете, че $A_q(n,d) \geq q^n / \sum_{i=0}^{d-1} \binom{n}{i}(q-1)^i$.*

Следствие 2.4.14 *Съществува $[n,k.d]$ код удовлетворяващ неравенството*

$$\sum_{i=0}^{d-2} \binom{n}{i}(q-1)^i \geq q^{n-k}.$$

По-долу ще опишем най-простите начини за построяване на нови кодове от дадени други такива.

1. Удължаване на код: Увеличаване на блоковата дължина чрез добавяне на един или повече проверочни символа към всяка кодова дума, т.е. без да променяме броя на кодовите думи.

Дефиниция 2.4.15 *Ако $\mathcal{C}$ е код с блокова дължина n над азбуката $\mathbb{F}_q$,* ***удължен код*** *на $\mathcal{C}$ се нарича*

$$\widehat{\mathcal{C}} \stackrel{\text{def}}{=} \{(c_0, c_1 \ldots, c_n) \mid (c_1 \ldots, c_n) \in \mathcal{C},\ c_0 + \cdots + c_n = 0\}$$

Забележка 2.4.16 *В литературата често се употребява думата разширен код за означаване на код удължен с проверка по четност. Причината за това е традицията и донякъде неподходящи преводи на чуждестранни книги. Ние ще употребяваме термина удължен тъй като името разширен съответства по-добре на друга операция (виж по-долу).*

Ако $\mathcal{C}$ е линеен код, удължаването му е еквивалентно на добавяне на нов стълб към пораждащата му матрица $\mathbf{G}$, съответно на нов стълб и ред към проверочната му матрица $\mathbf{H}$. Следователно, проверочната матрица на $\widehat{\mathcal{C}}$ има вида:

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

В двоичния случай процесът е просто добавяне на обща проверка по четност и очевидно, че $d(\widehat{\mathcal{C}}) = d(C) + 1$, ако минималното разстояние $d(\mathcal{C})$ е нечетно. Например, когато $\mathcal{C}$ е $[2^m - 1,\ 2^m - m - 1,\ 3]$ код на Хеминг $\widehat{\mathcal{C}}$ има параметри $[2^m,\ 2^m - 1 - m,\ 4]$ и се нарича ***удължен (extended) код на Хеминг***.

Дефиниция 2.4.17 *Дуалният код на удължения Хемингов код се нарича* ***код на Рид-Малер от първи ред*** *и се бележи с* $R(1, m)$.

От дефиницията веднага следва, че написаната по-горе матрица $\widehat{\mathbf{H}}$ е пораждаща матрица на кода на Рид-Малер от първи ред $R(1, m)$, когато $\mathbf{H}$ е проверочна матрица на Хеминговия код.

Обратният процес на удължаването е:

2. Скъсяване на код: скъсява се дължината на кода чрез отстраняване на координатна позиция без да се намалява обема на кода. Ако $\mathcal{C}$ е q-ичен (n, M, d) код с $d \geq 2$, то скъсеният код $\mathcal{C}^*$ има параметри $n^* = n - 1,\ \ M^* = M\ \ d^* = d$ или $d - 1$, но най-често $d - 1$.

При линейните кодове скъсяването съответства на отстраняване на стълб от пораждащата матрица, при което обаче рангът ѝ не намалява. Върху проверочната матрица операцията рефлектира в отстраняване на ред и стълб.

Упражнение 2.4.4 *Докажете, че двоичен* $(n, M, 2t + 1)$ *код съществува тогава и само тогава, когато* $(n + 1, M, 2t + 2)$ *код съществува.*

3. Уголемяване на код е процес на добавяне на нови кодови думи, т.е. увеличаване броя на информационните символи на кода без да се променя дължината му. Общ метод за увеличаване обема на двоичен код $\mathcal{C}$ е включването към него на допълнението на всяка кодова дума. Под ***допълнение*** $\overline{c}$ на двоичния вектор c се разбира векторът, получен от c чрез заместване на нулите с единици и обратно, т.е. $\overline{c} = \mathbf{1} + c$, където $\mathbf{1}$ означава вектора състоящ се само от единици. Тогава уголеменият код ще бъде $\mathcal{C} \cup \overline{\mathcal{C}} = \mathcal{C} \cup \{\mathbf{1} + \mathcal{C}\}$.

Упражнение 2.4.5 *Ако* $\mathcal{C}$ *е двоичен* (n, M, d) *код, то*

$$d(\mathcal{C} \cup \overline{\mathcal{C}}) \ = \ \min\{d,\ n - d_{max}\},$$

където d_{max} *е максималното разстояние между кодовите думи на* $\mathcal{C}$.

Упражнение 2.4.6 *Ако $\mathcal{C}$ е двоичен линеен $[n, k, d]$ код, който не съдържа вектора само от единици* $\mathbf{1}$*, то $\mathcal{C} \cup \overline{C}$ е двоичен линеен $[n, k+1, d')$ код, където*

$$d' = \min\{d,\ n - w_{max}\}.$$

В случая на линейни кодове, увеличаването обема на кода рефлектира в добавяне на ред (линейно независим от предходните) към пораждащата матрица $\mathbf{G}$ и отстраняване на ред в проверочната $\mathbf{H}$.

4. Свиване на код: това е обратния на уголемяването процес, т.е. намаляване броя на кодовите думи чрез отстраняване на някои от тях. В линейния случай операцията е отстраняване на ред в $\mathbf{G}$ и съответно добавяне на ред (с увеличаване на ранга) в $\mathbf{H}$. Получените кодове в този случай са също линейни.

Упражнение 2.4.7 *Нека $\mathcal{C}$ е двоичен $[n, k, d]$ линеен код с поне една кодова дума с нечетно тегло. Тогава точно половината от кодовите думи са с нечетно тегло и полученият след свиване на $\mathcal{C}$ код има параметри $[n, k-1, d']$, където $d' \geq d$.*

Следващите операции комбинират по две от описаните по-горе.

Съкращаване на код: скъсяване дължината на кода чрез намаляване броят на информационните символи, т.е това е процес на едновременно скъсяване и свиване на кода. Най-често това се осъществява чрез запазване само на тези кодови думи, които имат даден символ във фиксирана позиция, т.е. $c_i = a$. Тази операция се нарича понякога съкращаване по $c_i = a$.

Упражнение 2.4.8 *Ако $\mathcal{C}$ е двоичен $[n, k, d]$ код, то съкращаване по $c_i = 0$ води до $[n-1, k-1, d]$ код.*

Разширяване на код: при тази операция дължината нараства посредством добавяне на информационни символи, т.е. представлява комбинация от удължаване и увеличаване на обема на кода. Например, чрез разширение на симплексния $[2^k - 1,\ k,\ 2^{k-1}]$ код с добавяне на допълнението на всяка кодова дума и проверка почетност се получава $[2^k,\ k,\ 2^{k-1}]$ кодът на

Рид-Малер от първи ред. Тази операция отговаря на конструиране на пораждаща матрица от вида

$$\overline{\mathbf{G}} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & & & & \\ \vdots & & & \mathbf{G} & \\ 0 & & & & \end{pmatrix},$$

където $\mathbf{G}$ е пораждащата матрица на изходния код.

Забележка 2.4.18 *Напомняме отново, че думата разширен код по традиция в някои книги се ползва за означаване на удължен с проверка по четност код, което е различно от код получен с операцията "разширение".*

Дефиниция 2.4.19 *Нека $\mathcal{C}$ е линеен $[n,k]$ код и $\mathbf{u} \in \mathcal{C}$ е кодова дума с тегло w.* ***Остатъчен код относно*** $\mathbf{u}$ *се нарича кодът $Res(\mathcal{C},\mathbf{u})$, който е получен чрез задрасквање на всички позиции, където $\mathbf{u}$ има ненулеви символи.*

Нека $\mathbf{G}$ е пораждаща матрица на $\mathcal{C}$ с първи ред u. Тогава $Res(\mathcal{C},u)$ има пораждаща матрица G_0, където

$$\mathbf{G} = \begin{pmatrix} 0\,0\dots0 & \overbrace{*\,*\dots*}^{w} \\ G_0 & G_1 \end{pmatrix}$$

Лема 2.4.20 *Ако съществува q-ичен линеен $[n,k,d]$ код, то съществува и $[n-d,\ k-1,\ d_0]$ код с $d_0 \geq \lceil d/q \rceil$ ($\lceil x \rceil$ означава най-малкото цяло число $\geq x$).*

Доказателство. Нека $u \in \mathcal{C}$ е с тегло $\mathrm{wt}(u) = d$. С точност до еквивалентност на кодове може да предположим, че $u = (0\dots0|\,u_1\,)$, $u_1 \in \mathbb{F}^d$ и за пораждаща матрица $\mathbf{G}$ е избрана матрица с първи ред u, т.е.

$$\mathbf{G} = \begin{pmatrix} \overbrace{0\,0\dots0}^{n-d} & \overbrace{*\,*\dots*}^{d} \\ G_0 & G_1 \end{pmatrix}$$

Допускането, че $rank(G_0) < k-1$ влече, че съществува и още един кодов вектор имащ само нули в първите $n-d$ позиции. Но в такъв случай съществува (с евентуално добавяне на подходяща кодова дума кратна на u) и кодова дума с тегло помалко от d, което е противоречие. Следователно G_0 поражда $[n-d,\ k-1]$ линеен код и нека d_0 е минималното му тегло. Нека $c_0 \in Res(\mathcal{C}, u)$ и $c = (c_0|c_1)$ е кодова дума на $\mathcal{C}$ имаща вектора c_0 за първи $n-d$ позиции. Сравнявайки c с u нека b_i е броя на ненулевите координати на c_1, които съвпадат с i-тото кратно на $u = (0|u_1)$, $i = 1, 2, ..., q-1$. (т.е. с вектора $i.u$). Тогава $b_1 + b_2 + \cdots + b_{q-1} = \mathrm{wt}(c_1)$ и най-голямото число b измежду $\{b_j\}$ удовлетворява $b \geq \frac{\mathrm{wt}(c_1)}{q-1}$. Ако b съответства на кратното βu, то кодовата дума $c_1 - \beta u$ на $\mathcal{C}$ е вектора с най-много нули в последните d позиции и

$$\mathrm{wt}(c_0) + d - b = \mathrm{wt}(c - \beta u) \geq d.$$

Следователно $\mathrm{wt}(c_0) \geq b \geq \frac{\mathrm{wt}(c_1)}{q-1}$. Тъй като $\mathrm{wt}(c_0) + \mathrm{wt}(c_1) = \mathrm{wt}(c) \geq d$, то

$$q.\mathrm{wt}(c_0) = (q-1)\mathrm{wt}(c_0) + \mathrm{wt}(c_0) \geq \mathrm{wt}(c_1) + \mathrm{wt}(c_0) \geq d.$$

Следователно $\mathrm{wt}(c_0) \geq \frac{d}{q}$ и тъй като $\mathrm{wt}(c_0)$ е цяло число и c_0 е произволно, имаме $d_0 \geq \lceil \frac{d}{q} \rceil$.

◇

Теорема 2.4.21 (граница на Грийсмър). *Минимално възможната стойност на n, за която съществува $[n,\ k,\ d]$ код удовлетворява*

$$n \geq g_q(k,d) \ = \ \sum_{j=0}^{k-1} \left\lceil \frac{d}{q^j} \right\rceil .$$

Доказателство. Следва от лема 2.4.20 с индукция по k.

◇

Дефиниция 2.4.22 *Линейните кодове с параметри $[g_q(k,d),\ k,\ d]$ се наричат* ***Грийсмърови.***

Такива кодове съществуват (например симплекс кодовете), но за някои двойки $(k,\ d)$ минимално възможната блокова дължина $n > g_q(k,d)$.

Дефиниция 2.4.23 *Минималната стойност на n, за която съществува $q-$ичен $[n, k, d]$ код се бележи с $n_q(k,d)$. Кодовете с такива блокови дължини се наричат* ***оптимални****. Очевидно $n_q(n,d) \geq g_q(k,d)$.*

Например, $g_2(6,3) = 9$, но $n_2(6,3) = 10$. ([10,6,3] код може да се получи чрез съкращаване последователно пет пъти на [15,11,3] кода на Хеминг.)

2.5 Уравнения на Мак-Уйлямс.

Дефиниция 2.5.1 *Нека $\mathcal{C}$ е код с дължина n и $A_i = |\{u \in \mathcal{C} \mid \mathrm{wt}(u) = i\}|$. Множеството $\{A_0, A_1, \ldots, A_n\}$ се нарича* ***тегловен спектър (тегловно разпределение)*** *на C.*

Дефиниция 2.5.2 ***Хемингова*** *тегловна функция на кода C наричаме*

$$W_C(x,y) = \sum_{u \in \mathcal{C}} x^{n-wt(u)} y^{wt(u)} = \sum_{i=0}^{n} A_i x^{n-i} y^i.$$

Под наименованието "уравнения на Мак-Уйлямс" (MacWilliams) се разбира съвкупността от тъждества даващи връзката между разпределенията на теглата на един линеен код и неговия дуален. Те са един твърде често използван апарат в теория на кодирането. Ще ги извлечем в малко по-обща форма. Читателите, които намерят следващите резултати за твърде алгебрични спокойно могат да пропуснат доказателствата и да се задоволят с формулировката на тъждествата само в Хеминговия случай (теорема 2.5.12).

Нека подредим елементите на крайното поле $\mathbb{F} = GF(q)$ в някакъв ред, т.е. да считаме, че $\mathbb{F} = GF(q) = \{\alpha_0, \alpha_1, \ldots, \alpha_{q-1}\}$.

Дефиниция 2.5.3 *Нека $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ е произволен вектор от $n-$мерното пространство F^n.* ***Композит*** *на вектора $\mathbf{u}$ наричаме наредената $q-$орка*

$$\mathbf{comp}(\mathbf{u}) = (s_0, s_1, \ldots, s_{q-1}), \text{ където } s_i = s_i(\mathbf{u}) = |\{u_j = \alpha_i\}|, \quad \sum_{i=0}^{q-1} s_i = n.$$

Дефиниция 2.5.4 ***Пълна тегловна функция*** *на кода $\mathcal{C}$ наричаме полинома на q променливи $z_0, \ldots, z_{q-1}$:*

$$\mathcal{W}_C(z_0, \ldots, z_{q-1}) \stackrel{def}{=} \sum_{u \in \mathcal{C}} z_0^{s_0(u)} z_1^{s_1(u)} \ldots z_{q-1}^{s_{q-1}(u)} = \sum_{t=(t_1,\ldots,t_{q-1})} A(t) z_0^{t_0} \ldots z_{q-1}^{t_{q-1}},$$

където $A(t)$ е броя на кодовите думи $u \in \mathcal{C}$ имащи композит $\mathbf{comp}(\mathbf{u}) = (t_0, \ldots, t_{q-1})$.

Пример 2.5.1 Нека $\mathcal{C}$ е $[4, 2, 3]$ троичен код с пораждаща матрица $\begin{pmatrix} 1 & 0 & 2 & 2 \\ 0 & 1 & 2 & 1 \end{pmatrix}$, т.е.

$$C = \{0000,\ 1022,\ 0121,\ 1110,\ 2011,\ 0212,\ 2220,\ 2102,\ 1201\}.$$

$$comp(0000) = (4, 0, 0);\ comp(1022) = (1, 1, 2);\ comp(0121) = (1, 2, 1);$$

$$comp(1110) = (1, 3, 0);\ comp(2011) = (1, 2, 1);\ comp(0212) = (1, 1, 2);$$

$$comp(2220) = (1, 0, 3);\ comp(2102) = (1, 1, 2);\ comp(1201) = (1, 2, 1).$$

Следователно

$$\mathcal{W}_C(z_0, z_1, z_2) = z_0^4 + 3z_0 z_1 z_2^2 + 3z_0 z_1^2 z_2 + z_0 z_1^3 + z_0 z_2^3.$$

Дефиниция 2.5.5 *Нека $\mathcal{C}$ е код с дължина n и $A_i = |\{u \in \mathcal{C} \mid \mathrm{wt}(u) = i\}|$. Множеството $\{A_0, A_1, \ldots, A_n\}$ се нарича* ***тегловен спектър (тегловно разпределение)*** *на C.*

Хеминговата тегловна функция може да се получи от пълната тегловна функция с полагането $z_0 = x$ и $z_1 = \ldots = z_{q-1} = y$.

Аналогично, ако номерираме елементите на полето $\mathbb{F}$ така, че $\alpha_i = \alpha_{q-i}$ и положим $z_{q-i} = z_i$, $i = 1, 2, \ldots, (q-1)/2$, ще получим тегловна функция наречена ***Лиева*** и тя отразява разпределението на теглата на кода, но при лиева метрика.

Нека ξ е $p-$ти примитивен корен на единицата в полето $\mathbb{C}$ на комплексните числа, т.е. $\xi = \cos\frac{2\pi}{p} + \imath \sin\frac{2\pi}{p}$. Да фиксираме един базис на $\mathbb{F} = GF(q)$, $q = p^m$, над $\mathbb{Z}_p$. Тогава на всеки елемент $\alpha \in \mathbb{F}$ еднозначно съответства $m-$мерен вектор $(a_1, \ldots, a_m)$ над $\mathbb{Z}_p$, който ще отъждествяваме с α.

Дефиниция 2.5.6 ***Характер на*** $\mathbb{F} = GF(q)$ ***съответстващ на*** α *ще наричаме всяко изображение* $\mathcal{X}_\alpha : \mathbb{F} \to \mathbb{C}^*$ *дефинирано с равенството:*

$$\mathcal{X}_\alpha(\beta) \stackrel{def}{=} \xi^{\alpha\circ\beta} = \xi^{a_1b_1+\cdots+a_mb_m}.$$

Тривиално се проверяват следните свойства:

$$\begin{array}{ll} \mathcal{X}_0(\beta) & = 1, \\ \mathcal{X}_\alpha(\beta) & = \mathcal{X}_\beta(\alpha) \\ \mathcal{X}_\alpha(\beta+\gamma) & = \mathcal{X}_\alpha(\beta)\mathcal{X}_\alpha(\gamma) \\ \mathcal{X}_{\alpha+\beta}(\gamma) & = \mathcal{X}_\alpha(\gamma)\mathcal{X}_\beta(\gamma) \end{array}$$

за всяко α, β, γ от F.

Горните свойства показват, че характерите са хомоморфизми на адитивната група на $\mathbb{F}$ в групата на комплексните числа с модул 1 и образуват група изоморфна на адитивната група на $\mathbb{F}$.

Лема 2.5.7 *За всеки елемент* $\alpha \in \mathbb{F} = GF(q)$ *е в сила:*

$$\sum_{\beta\in\mathbb{F}} \mathcal{X}_\alpha(\beta) = \begin{cases} 0, & \alpha \neq 0 \\ q, & \alpha = 0. \end{cases}$$

Доказателство.

$$\begin{aligned} \sum_{\beta\in\mathbb{F}} \mathcal{X}_\alpha(\beta) &= \sum_{(b_1,\ldots,b_m)} \xi^{a_1b_1+\cdots+a_mb_m} = \sum_{(b_1,\ldots,b_m)} \xi^{a_1b_1}\xi^{a_2b_2}\ldots\xi^{a_mb_m} = \\ &= \sum_{(b_1,\ldots,b_{m-1})} \xi^{a_1b_1}\ldots\xi^{a_{m-1}b_{m-1}}.(\sum_{b_m\in Z_p} \xi^{a_mb_m}) = \ldots \\ &= (\sum_{b_1\in Z_p} \xi^{a_1b_1})\ldots(\sum_{b_m\in Z_p} \xi^{a_mb_m}). \end{aligned}$$

Но за $\alpha \neq 0$ съществува поне едно $a_l \neq 0$, т.е. $\xi^{a_l} \neq 1$, откъдето получаваме

$$\sum_{b_l\in\mathbb{Z}_p} \xi^{a_lb_l} = \sum_{b_l=0}^{p-1} (\xi^{a_l})^{b_l} = \frac{1-(\xi^{a_l})^p}{1-\xi^{a_l}} = 0.$$

Следователно $\sum_{\beta\in\mathbb{F}} \mathcal{X}_\alpha(\beta) = 0$.
Ако $\alpha = 0$, то $\mathcal{X}_0(\beta) = 1$, за всяко $\beta \in \mathbb{F}$, откъдето следва и второто равенство.

$\diamond$

Нека $f : \mathbb{F}^n \to L$ е произволно изображение на $\mathbb{F}^n$ в някакво линейно пространство L над полето на комплексните числа $\mathbb{C}$. Да фиксираме един нетривиален характер на $\mathbb{F}$, например $\mathcal{X} = \mathcal{X}_1 : \quad \mathcal{X}(\gamma) = \xi^{c_1}$, където $\gamma = (c_1, \ldots, c_m)$.

Дефиниция 2.5.8 ***Характер на*** $\mathbb{F}^n$ ***в*** $\mathbb{C}^*$ ***съответстващ на*** $u \in \mathbb{F}^n$ *наричаме изображението:*

$$\mathcal{X}_u(v) \stackrel{def}{=} \mathcal{X}(u \circ v) = \prod_{i=1}^{n} \mathcal{X}(u_i v_i), \quad v \in \mathbb{F}^n.$$

Дефиниция 2.5.9 *Преобразуване на Адамар на изображението* $f : \mathbb{F}^n \to L$ *наричаме*

$$\hat{f}(u) \stackrel{def}{=} \sum_{v \in \mathbb{F}^n} \mathcal{X}_u(v) f(v).$$

Очевидно, че $\hat{f}$ е също изображение на $\mathbb{F}^n \to L$.

Лема 2.5.10 *Ако* $\mathcal{C}$ *е* $[n,\ k]$ *код над* $\mathbb{F} = GF(q)$, *то*

$$\sum_{v \in \mathcal{C}^\perp} f(v) \ = \ \frac{1}{|C|} \sum_{u \in \mathcal{C}} \hat{f}(u).$$

Доказателство.

$$\hat{f}(u) = \sum_{v \in \mathbb{F}^n} \mathcal{X}_u(v) f(v) = \sum_{v \in \mathbb{F}^n} \mathcal{X}(u \circ v) f(v)$$

Следователно

$$\sum_{u \in \mathcal{C}} \hat{f}(u) = \sum_{u \in \mathcal{C}} \sum_{v \in \mathbb{F}^n} \mathcal{X}(u \circ v) f(v) = \sum_{v \in \mathbb{F}^n} [\sum_{u \in \mathcal{C}} \mathcal{X}(u \circ v)] f(v).$$

Ако $v \in \mathcal{C}^\perp$, то $\mathcal{X}(u \circ v) = \mathcal{X}(0) = 1$ за всяко $u \in \mathcal{C}$. Следователно

$$\sum_{u \in \mathcal{C}} \mathcal{X}(u \circ v) = |\mathcal{C}|.1 = \ |\mathcal{C}|.$$

Нека $v \notin \mathcal{C}^{\perp}$ и $e_1, \ldots, e_k$ е един базис на $\mathcal{C}$ над $\mathbb{F}$. Тогава $u = u_1e_1 + \cdots + u_ke_k,\ \ u_j \in \mathbb{F}$ и

$$\sum_{u\in\mathcal{C}} \mathcal{X}(u \circ v) = \sum_{(u_1,\ldots,u_k)} \mathcal{X}((u_1e_1 + \cdots + u_ke_k,\ v)) =$$

$$\sum_{(u_1,\ldots,u_k)} \prod_{j=1}^{k} \mathcal{X}(u_j(e_j \circ v)) = \sum_{(u_1,\ldots,u_k)} \prod_{j=1}^{k} \mathcal{X}(u_jw_j) =$$

$$(\sum_{u_1\in\mathbb{F}} \mathcal{X}(u_1w_1)).(\sum_{u_2\in\mathbb{F}} \mathcal{X}(u_2w_2)) \ldots (\sum_{u_k\in\mathbb{F}} \mathcal{X}(u_kw_k)),$$

където $w_j = (e_j, v)$. Но поне една от стойностите $w_j \neq 0$ когато $v \notin \mathcal{C}^{\perp}$. Следователно, u_jw_j описва всички елементи на полето $\mathbb{F}$, откъдето и Лема 2.5.7 следва, че $(\sum_{u_j\in\mathbb{F}} \mathcal{X}(u_jw_j)) = 0$. Следователно

$$\sum_{u\in\mathcal{C}} \mathcal{X}(u \circ v) = 0 \quad \text{за всяко } \ v \notin \mathcal{C}^{\perp},$$

откъдето и казаното по-горе следва твърдението на лемата. ◇

Теорема 2.5.11 *Ако $\mathcal{C}$ е $[n,k]$ код над $\mathbb{F} = GF(q)$ с тегловна функция $\mathcal{W}_C(z_0, z_1, \ldots, z_{q-1})$, то пълната тегловна функция на дуалния му код $\mathcal{C}^{\perp}$ се задава с:*

$$\mathcal{W}_{C^{\perp}}(z_0, z_1, \ldots, z_{q-1}) =$$
$$\frac{1}{|\mathcal{C}|}\mathcal{W}_C(\sum_{j=0}^{q-1} \mathcal{X}(\alpha_0\alpha_j)z_j,\ \sum_{j=0}^{q-1} \mathcal{X}(\alpha_1\alpha_j)z_j,\ \ldots,\ \sum_{j=0}^{q-1} \mathcal{X}(\alpha_{q-1}\alpha_j)z_j).$$

Доказателство. Нека

$$f : \left| \begin{array}{lcl} \mathbb{F}^n & \to & \mathbb{C}[z_0, \ldots, z_{q-1}] \\ f(u) & = & z_0^{s_0(u)} \ldots z_{q-1}^{s_{q-1}(u)}. \end{array} \right.$$

Тогава

$$\begin{aligned} \hat{f}(u) &= \sum_{v\in\mathbb{F}^n} \mathcal{X}_u(v)f(v) = \sum_{v\in F^n} \mathcal{X}(u \circ v)z_0^{s_0(v)} \ldots z_{q-1}^{s_{q-1}(v)} = \\ &= \sum_{v\in\mathbb{F}^n} \mathcal{X}(u_1v_1)\mathcal{X}(u_2v_2) \ldots \mathcal{X}(u_nv_n)z_0^{s_0(v)} \ldots z_{q-1}^{s_{q-1}(v)}. \end{aligned}$$

Нека s_{ij} е броят на координатите на v, чиято стойност е α_j, а номерът им е измежду тези, в които u има стойност α_i, т.е. s_{ij} е броя на двйките (α_i, α_j) измежду двойките (u_l, v_l). Ясно е, че

$$s_{i0} + s_{i1} + \cdots + s_{i,q-1} = s_i(u)$$

$$s_{0j} + s_{1j} + \cdots + s_{q-1,j} = s_j(v).$$

Нека $i_1, i_2, \ldots, i_{s_0(u)}$ са позициите, в които u има стойност α_0, в тези с номера $j_1, j_2, \ldots, j_{s_1(u)}$ - тези с α_1 и т.н. позициите с номера $l_1, l_2, \ldots, l_{s_{q-1}(u)}$ на u имат стойност α_{q-1}. Тогава

$$\begin{aligned}
&\mathcal{X}(u_1v_1)\mathcal{X}(u_2v_2)\ldots\mathcal{X}(u_nv_n)z_0^{s_0(v)}\ldots z_{q-1}^{s_{q-1}(v)} = \\
&[\mathcal{X}(\alpha_0v_{i_1})\ldots\mathcal{X}(\alpha_0v_{i_{s_0}})z_0^{s_{00}}z_1^{s_{01}}\ldots z_{q-1}^{s_{0,q-1}}]\times \\
&[\mathcal{X}(\alpha_1v_{j_1})\ldots\mathcal{X}(\alpha_1v_{j_{s_0}})z_0^{s_{10}}z_1^{s_{11}}\ldots z_{q-1}^{s_{1,q-1}}]\times\cdots\times \\
&[\mathcal{X}(\alpha_{q-1}v_{l_1})\mathcal{X}(\alpha_{q-1}v_{j_2})\ldots\mathcal{X}(\alpha_{q-1}v_{i_{s_{q-1}}})z_0^{s_{q-1,0}}z_1^{s_{q-1,1}}\ldots z_{q-1}^{s_{q-1,q-1}}] = \\
&[\mathcal{X}^{s_{00}}(\alpha_0\alpha_0)\mathcal{X}^{s_{01}}(\alpha_0\alpha_1)\ldots\mathcal{X}^{s_{0,q-1}}(\alpha_0\alpha_{q-1})z_0^{s_{00}}z_1^{s_{01}}\ldots z_{q-1}^{s_{0,q-1}}]\times \\
&[\mathcal{X}^{s_{10}}(\alpha_1\alpha_0)\mathcal{X}^{s_{11}}(\alpha_1\alpha_1)\ldots\mathcal{X}^{s_{1,q-1}}(\alpha_1\alpha_{q-1})z_0^{s_{10}}z_1^{s_{11}}\ldots z_{q-1}^{s_{1,q-1}}]\times\cdots\times \\
&[\mathcal{X}^{s_{q-1,0}}(\alpha_{q-1}\alpha_0)\mathcal{X}^{s_{q-1,1}}(\alpha_{q-1}\alpha_1)\ldots\mathcal{X}^{s_{q-1,q-1}}(\alpha_{q-1}\alpha_{q-1}) \\
&z_0^{s_{q-1,0}}z_1^{s_{q-1,1}}\ldots z_{q-1}^{s_{q-1,q-1}}].
\end{aligned}$$

Но $(\alpha_0v_{i_1}, \ldots, \alpha_0v_{i_{s_0}})$ описва всички s_0-орки, когато v описва F^n, т.е.
$(s_{00}, \ldots, s_{0,q-1})$ описва всички $q-$орки със свойството $s_{00} + s_{01} + \cdots + s_{0,q-1} = s_0(u)$. Аналогично $(s_{i0}, s_{i1}, \ldots, s_{i,q-1})$ описва всички $q-$орки със свойството $s_{i0}+s_{i1}+\cdots+s_{i,q-1} = s_i(u)$. Следователно

$$\hat{f}(u) = \sum_{v\in\mathbb{F}^n} \mathcal{X}(u_1v_1)\mathcal{X}(u_2v_2)\ldots\mathcal{X}(u_nv_n)z_0^{s_0(v)}\ldots z_{q-1}^{s_{q-1}(v)} =$$

$$= \prod_{i=0}^{q-1}\left[\sum_{(s_{i0},\ldots,s_{iq-1})} \mathcal{X}^{s_{i0}}(\alpha_i\alpha_0)\ldots\mathcal{X}^{s_{i,q-1}}(\alpha_i\alpha_{q-1})z_0^{s_{i0}}z_1^{s_{i1}}\ldots z_{q-1}^{s_{i,q-1}}\right] =$$

$$= \left[\sum_{j=0}^{q-1}\mathcal{X}(\alpha_0\alpha_j)z_j\right]^{s_0(u)}\left[\sum_{j=0}^{q-1}\mathcal{X}(\alpha_1\alpha_j)z_j\right]^{s_1(u)}\ldots\left[\sum_{j=0}^{q-1}\mathcal{X}(\alpha_{q-1}\alpha_j)z_j\right]^{s_{q-1}(u)}.$$

Сега прилагайки лема 2.5.10 получаваме:

$$\sum_{v\in C^\perp} z_0^{s_0(v)}\dots z_{q-1}^{s_{q-1}(v)} =$$

$$\frac{1}{|C|}\sum_{u\in C}\left[\sum_{j=0}^{q-1}\mathcal{X}(\alpha_0\alpha_j)z_j\right]^{s_0(u)}\dots\left[\sum_{j=0}^{q-1}\mathcal{X}(\alpha_{q-1}\alpha_j)z_j\right]^{s_{q-1}(u)},$$

с което теоремата е доказана.

◇

Ще отбележем, че в теорема 2.5.11, $\mathcal{W}_{C^\perp}(z_0,z_1,\dots,z_{q-1})$ се получава от $\mathcal{W}_C(z_0,z_1,\dots,z_{q-1})$ чрез линейна смяна

$$\hat{z} \;=\; \mathbf{M}\begin{pmatrix} z_0 \\ \vdots \\ z_{q-1}\end{pmatrix},$$

където $\mathbf{M}$ е $q\times q$ матрица

$$\mathbf{M} \;=\; \frac{1}{|C|}\begin{pmatrix} \mathcal{X}(\alpha_0\alpha_0) & \mathcal{X}(\alpha_0\alpha_1) & \dots & \mathcal{X}(\alpha_0\alpha_{q-1}) \\ \mathcal{X}(\alpha_1\alpha_0) & \mathcal{X}(\alpha_1\alpha_1) & \dots & \mathcal{X}(\alpha_1\alpha_{q-1}) \\ \dots & \dots & \dots & \dots \\ \mathcal{X}(\alpha_{q-1}\alpha_0) & \mathcal{X}(\alpha_{q-1}\alpha_1) & \dots & \mathcal{X}(\alpha_{q-1}\alpha_{q-1})\end{pmatrix}.$$

Теорема 2.5.12 *Ако $\mathcal{C}$ е $[n,k]$ линеен код над $\mathbb{F}=GF(q)$ с хемингова тегловна функция $\mathcal{W}_C(x,y)$, то хеминговата тегловна функция на дуалния му код удовлетворява*

$$\mathcal{W}_{C^\perp}(x,y) = \frac{1}{|\mathcal{C}|}\mathcal{W}_C(x+(q-1)y,\; x-y). \tag{2.6}$$

Доказателство. В теорема 2.5.11 полагаме $z_0 = x$, $y = z_0 = \dots = z_{q-1}$. Вземайки предвид, че $\mathcal{X}(\alpha_0\alpha_j) = \mathcal{X}(0) = 1$ и $\sum_{j=1}^{q-1}\mathcal{X}(\alpha_i\alpha_j) = -\mathcal{X}(\alpha_i\alpha_0) = -1$, тъй като $\sum_{j=0}^{q-1}\mathcal{X}(\alpha_i\alpha_j) = 0$ за $i\neq 0$, получаваме

$$\sum_{j=0}^{q-1}\mathcal{X}(\alpha_0\alpha_j)z_j = x+(q-1)y.$$

$$\sum_{j=0}^{q-1}\mathcal{X}(\alpha_i\alpha_j)z_j = \mathcal{X}(\alpha_i\alpha_0)x + \left(\sum_{j=1}^{q-1}\mathcal{X}(\alpha_i\alpha_j)\right)y = \; x-y.$$

◇

Тъй като $|\mathcal{C}| = q^{Rn} = (q^R)^n$ и $\mathcal{W}_C$ е полином от степен n, то можем да запишем и

$$\mathcal{W}_{C^\perp}(x,y) = \mathcal{W}_C\left(\frac{x+(q-1)y}{q^R},\ \frac{x-y}{q^R}\right).$$

Последното се прави обикновенно при скорост на кода $R = 1/2$ при самодуални кодове.

Нека $\{A_i\}_1^n$ и $\{B_i\}_1^n$ са спектрите на кода $\mathcal{C}$ и $\mathcal{C}^\perp$ съответно. Тогава теорема 2.5.12 при полагане $x = 1$ ни дава:

$$\sum_{i=0}^{n} B_i y^i \ = \ \frac{1}{q^k}\sum_{i=0}^{n} A_i(1+(q-1)y)^{n-i}(1-y)^i. \tag{2.7}$$

От друга страна биномът на Нютон ни дава, че:

$$(1+(q-1)y)^{n-i}(1-y)^i = \sum_{m=0}^{n} P_m(i)y^m,$$

където

$$P_m(i) = \sum_{j=0}^{m}(-1)^j\binom{i}{j}\binom{n-i}{m-j}(q-1)^{m-j}.$$

Сега приравнявайки коефициентите пред еднаквите степени на y получаваме следното следствие от теорема 2.5.12.

Следствие 2.5.13 *Ако $\mathcal{C}$ е $[n,k]$ линеен код със спектър $\{A_i\}_1^n$, а $\mathcal{C}^\perp$ е дуалният му код със спектър $\{B_i\}_1^n$, то*

$$B_m = \frac{1}{q^k}\sum_{i=0}^{n} A_i P_m(i) \tag{2.8}$$

Дефиниция 2.5.14 ***Полином на Кравчук*** *наричаме полинома*

$$P_m(t) = \sum_{j=0}^{m}(-1)^j\binom{t}{j}\binom{n-t}{m-j}(q-1)^{m-j},\ m = 0,1\ldots n.$$

Ето няколко от първите полиноми на Кравчук при $q = 2$ записани по степените на t.

$$\begin{aligned}
P_0(t) &= 1\\
P_1(t) &= n-2t\\
P_2(t) &= \binom{n}{2} - 2nt + 2t^2\\
P_3(t) &= \binom{n}{3} - (n^2-n+2/3)t + 2nt^2 - \frac{4}{3}t^3
\end{aligned}$$

Ако положим в (1) $y = 1$, ще получим аналогично на (2) равенството

$$\sum_{i=0}^{n} B_i x^{n-i} \;=\; \frac{1}{q^k} \sum_{i=0}^{n} A_i (x+q-1)^{n-i}(x-1)^i. \tag{2.9}$$

Ако разменим местата на C и $C^\perp$ в горните разглеждания, ще получим съответно:

$$\sum_{i=1}^{n} A_i y^i \;=\; \frac{1}{q^{n-k}} \sum_{i=0}^{n} B_i (1+(q-1)y)^{n-i}(1-y)^i. \tag{2.10}$$

$$\sum_{i=1}^{n} A_i x^{n-i} \;=\; \frac{1}{q^{n-k}} \sum_{i=0}^{n} B_i (x+q-1)^{n-i}(x-1)^i. \tag{2.11}$$

Диференцирайки m пъти (5) и полагайки $y = 1$ за $m = 0, 1, 2, \ldots$ получаваме:

$$\frac{1}{q^k} \sum_{i=m}^{n} \binom{i}{m} A_i \;=\; \frac{1}{q^m} \sum_{i=0}^{m} (-1)^i (q-1)^{m-i} \binom{n-i}{n-m} B_i. \tag{2.12}$$

Аналогично диференцирайки m пъти, $m = 0, 1, 2, \ldots$, и замествайки $x = 1$ получаваме от (6):

$$\frac{1}{q^k} \sum_{i=m}^{n-m} \binom{n-i}{m} A_i \;=\; \frac{1}{q^m} \sum_{i=0}^{m} \binom{n-i}{n-m} B_i. \tag{2.13}$$

2.6 Допълнителни задачи към Глава 2.

Problem 2.6.1 *Докажете, че* $d(x,y) = \mathrm{wt}(x) + \mathrm{wt}(y) - 2\mathrm{wt}(x * y)$, *където* $x * y = (x_1 y_1, \ldots, x_n y_n)$.

Problem 2.6.2 *Докажете, че кодът* $\mathcal{C}$ *едновременно изправя* t *грешки и открива до* $d > t$ *грешки тогава и само тогава, когато* $d(\mathcal{C}) \geq d + t + 1$.

Problem 2.6.3 *По BSC канал с* $p = 0,1$ *се предават данни като се използва* $[6,3]$ *кодът от пример 2.2.3. Каква е вероятността за грешно декодиране на кодова дума?*

Problem 2.6.4 *Нека по BSC канал с вероятност за грешка p се предават данни като се използва блоков код $\mathcal{C}$ в режим на откриване на грешки. При откриване на грешка се изпраща заявка за препредаване на кодовата дума. Да се изрази вероятността за препредаване на кодова дума чрез p и спектъра на $\mathcal{C}$. Да се пресметне за $p = 0,01$ и за кода от пример 2.2.3.*

Problem 2.6.5 *Докажете, че ако пораждащата матрица на $\mathcal{C}$ има за стълбове всички $(q^k-1)/(q-1)$ непропорционални ненулеви вектори с дължина k, то всяка ненулева кодова дума е с тегло q^{k-1}.*

Problem 2.6.6 *За линения $[13, 5]$ двоичен код $\mathcal{C}$ е известно, че $\alpha_0 = 1$, $\alpha_1 = 13$, $\alpha_2 = 78$ и $\alpha_3 = 152$, където α_i е броят на лидерите на съседни класове по $\mathcal{C}$ с тегло i. Докажете, че $\alpha_5 \leq 2$ и $\alpha_i = 0$, за $i > 5$.*

Problem 2.6.7 *Докажете, че не съществува $[13, 6, 5]$ двоичен код.*

Problem 2.6.8 *Нека $\mathcal{C}$ е $[9, 7]$ двоичен код с $\mathbf{G} = (\mathbf{I}_7\, \mathbf{A})$, където*

$$\mathbf{A}^\tau = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Намерете спектъра на $\mathcal{C}$.

Problem 2.6.9 *Нека $\mathcal{C}$ е код с проверочна матрица $\mathbf{H} = (\mathbf{A}\, \mathbf{I})$. Докажете, че $\mathcal{C}$ е самодуален тогава и само тогава, когато $\mathbf{A}\mathbf{A}^\tau = -\mathbf{I}$.*

Problem 2.6.10 *Нека $\mathcal{C}$ е код с тегловна функция $W_{\mathcal{C}}(z) = \sum_{i=0}^{n} A_i z^i$. Докажете, че*

а) четният подкод $\mathcal{C}_0$ има тегловна функция

$$W_{\mathcal{C}_0}(z) = \frac{1}{2}[W_{\mathcal{C}}(z) + W_{\mathcal{C}}(-z)];$$

б) за удължения с проверка на четност код $\hat{\mathcal{C}}$ е в сила

$$W_{\hat{\mathcal{C}}}(z) = \frac{1}{2}[(1+z)W_{\mathcal{C}}(z) + (1-z)W_{\mathcal{C}}(-z)].$$

Глава 3

Циклични кодове

3.1 Въведение.

Дефиниция 3.1.1 *Кодът $\mathcal{C}$ над крайното поле $\mathbb{F}$ се нарича* ***цикличен****, когато всяка кодова дума след циклично завъртане остава също кодова дума, т.е. когато $(c_1, \ldots, c_n) \in C$ влече $(c_n, c_1, \ldots, c_{n-1}) \in C$.*

Например, двоичният [3,2,2] код $\mathcal{C} = \{000, 110, 011, 101\}$ е цикличен код.

Нека $\mathbb{F} = GF(q)$ е крайно поле с q елемента. Да разгледаме алгебрата $\mathcal{F}_n = \mathbb{F}[x]/(x^n - 1)$, т.е. пръстена от полиноми

$$a(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$$

от степен ненадминаваща n с операции събиране и умножение на полиноми по модул $(x^n - 1)$, явяващ се едновременно и линейно пространство над $\mathbb{F}$ (при същата операция събиране) с $\dim_{\mathbb{F}} \mathcal{F}_n = n$. Напомняме, че *идеал* $M \lhd \mathcal{F}_n$ се нарича линейно подпространство M на $\mathcal{F}_n$, удовлетворяващо следните условия: $f(x)m(x) \in \mathcal{F}_n$, за всяко $m(x) \in M$ и $f(x) \in \mathcal{F}_n$.

Както е добре известно от основния университетски курс по алгебра, идеалите в $\mathcal{F}_n$ са *главни*, т.е. съществува полином $g(x) \in M$ (наречен *пораждащ полином*) такъв, че всеки друг полином $a(x)$ на M е кратен на $g(x)$: $a(x) = t(x)g(x)$, $t(x) \in \mathcal{F}_n$. Този факт се отбелязва с $M = (g(x))$.

За да бъде определен еднозначно $g(x)$ се избира да бъде с минимална степен измежду полиномите в M и да е със старши коефициент единица.

Изображението $v = (v_0, v_1, \ldots, v_{n-1}) \Rightarrow v(x) = v_0 + v_1 x + \cdots + v_{n-1} x^{n-1}$ е очевидно изоморфизъм на $\mathbb{F}^n$ върху алгебрата $\mathcal{F}_n$ като линейни пространства. Следователно, всеки линеен код $\mathcal{C}$ над $\mathbb{F}$ с блокова дължина n може да се разглежда като линейно подпространство на алгебрата от полиноми $\mathcal{F}_n$.

Твърдение 3.1.2 *Линейният код $\mathcal{C}$ с блокова дължина n над $\mathbb{F}$ е цикличен тогава и само тогава, когато $\mathcal{C}$ е идеал в $\mathcal{F}_n$.*

Доказателство. Както отбелязахме вече кодът $\mathcal{C}$ може да се отъждествени с линейно подпространство на $\mathcal{F}_n$. От друга страна цикличното завъртане на $c(x) \in \mathcal{C}$ е еквивалентно на умножение (по модул x^n-1) на $c(x)$ с x. Следователно, $\mathcal{C}$ е цикличен тогава и само тогава, когато $x.c(x) \in C$ за всяко $c(x) \in \mathcal{C}$. Но очевидно, $xc(x) \in \mathcal{C}$ е еквивалентно с $v(x)c(x) \in \mathcal{C}$ за всяко $v(x) \in \mathcal{F}_n$. Следователно, кодът $\mathcal{C}$ е цикличен тогава и само тогава, когато е идеал на $\mathcal{F}_n$.

$\diamond$

Нека $\mathcal{C}$ е цикличен код. Съгласно току-що доказаното твърдение, той е идеал на $\mathcal{F}_n$. Както отбелязахме по-горе, съществува единствен полином $g(x) \in \mathcal{C}$ със старши коефициент 1 такъв, че $\mathcal{C} = (g(x))$. При това $g(x)$ е полином с най-ниска ненулева степен в кода и се нарича ***пораждащ полином*** на цикличния код $\mathcal{C}$.

Тъй като x^n-1 е нулевия елемент в $\mathcal{F}_n$, т.е. принадлежи на всеки идеал (цикличен код), то пораждащият полином $g(x)$ на всеки цикличен код в $\mathcal{F}_n$ дели $x^n - 1$.

Дефиниция 3.1.3 *Полиномът*

$$h(x) \ = \ (x^n - 1)/g(x)$$

се нарича ***проверочен полином*** *на цикличния код $\mathcal{C}$.*

Теорема 3.1.4 *Нека $\mathcal{C} = (g(x))$ е цикличен код с пораждащ полином $g(x) = g_0 + g_1x + \cdots + g_rx^r$. Тогава в сила са следните твърдения:*

1. Всяка кодова дума $c(x) \in \mathcal{C}$ се представя по единствен начин като произведение

$$c(x) = i(x).g(x),$$

където $i(x)$ е от степен по-малка от $k = n - r$;

2. $\dim \mathcal{C} = n - \deg g(x) = n - r = k$;

3. Кодът $\mathcal{C}$ има за пораждаща матрица $(n - r) \times n$ матрицата

$$\mathbf{G} \ = \ \begin{pmatrix} \overrightarrow{g(x)} \\ \overrightarrow{xg(x)} \\ \vdots \\ \overrightarrow{x^{n-r-1}g(x)} \end{pmatrix} = \begin{pmatrix} g_0 & g_1 & \dots & g_r & 0 & 0 & \dots & 0 \\ 0 & g_0 & \dots & g_{r-1} & g_r & 0 & \dots & 0 \\ \dots & & & & & & & \\ 0 & 0 & \dots & & & g_0 & \dots & g_r \end{pmatrix};$$

4. Ако $h(x) = h_0 + h_1 x + \cdots + h_k x^k$, $k = n - r$ *, то проверочна матрица* $\mathcal{C}$ *е следната* $r \times n$ *матрица:*

$$\mathbf{H} = \begin{pmatrix} \overleftarrow{h(x)} \\ \overleftarrow{xh(x)} \\ \vdots \\ \overleftarrow{x^{n-k-1}h(x)} \end{pmatrix} = \begin{pmatrix} 0 & 0 & \dots & 0 & h_k & \dots & h_1 & h_0 \\ 0 & 0 & \dots & h_k & h_{k-1} & \dots & h_0 & 0 \\ & & & & & & & \\ h_k & h_{k-1} & \dots & \dots & h_0 & \dots & 0 & 0 \end{pmatrix}.$$

Доказателство. 1.: $c(x) = t(x)g(x)$ в $\mathcal{F}_n$. Ако $\deg t(x) \geq n-r = k$, то

$$t(x) = q(x)h(x) + i(x), \quad \deg i(x) < k = n - r.$$

Следователно $c(x) = (q(x)h(x) + i(x))g(x)$, , т.е.

$$c(x) = q(x)\underbrace{(x^n - 1)}_{=0} + i(x)g(x) \ = \ i(x)g(x).$$

Ако съществуват $i(x)$ и $j(x)$ от степени по-малки от $k = n - r$ такива, че $c(x) = i(x)g(x) = j(x)g(x)$, то $(i(x) - j(x))g(x) = 0$, което е невъзможно.

2. и 3.: Кодовите вектори $g(x)$, $xg(x), \dots, x^{k-1}g(x)$, са линейно независими и всяка кодова дума се представя като тяхна линейна комбинация съгласно твърдение 1. Следователно, те образуват базис на $\mathcal{C}$ и

$$\dim \mathcal{C} = |\{g(x),\ xg(x), \dots, x^{k-1}g(x)\}| = k = n - \deg g(x).$$

4.: Нека $c(x) \in \mathcal{C}$. Тогава $c(x)h(x) = i(x)g(x)h(x) = 0$ в $\mathcal{F}_n$. Обратно, ако $c(x) \in \mathcal{F}_n$ и $c(x)h(x) = 0$, то

$$0 = c(x)h(x) = (q(x)g(x) + r(x))h(x) = q(x)(x^n - 1) + r(x)h(x).$$

Следователно, $r(x)h(x) = 0$ като полиноми от $\mathbb{F}[x]$, тъй като $\deg r(x) < \deg g(x) = n - k$. Но това е невъзможно освен в случая $r(x) \equiv 0$, т.е. когато $c(x) = q(x)g(x) \in C$. Следователно $c(x) \in C$ точно тогава, когато $c(x)h(x) = 0$ in $\mathcal{F}_n$. Тъй като $c(x)h(x) = \sum_{j=0}^{n-1}(\sum_{i=0}^{j} c_i h_{j-i})x^j$ горното е еквивалентно на $c(x) \in C$ тогава и само тогава, когато

$$c_0 h_j + c_1 h_{j-1} + \cdots + c_j h_0 = 0, \ \ j = k, k+1, \dots, n-1$$

и $h_j = 0$ for $j > k$.

$\diamond$

Забележка 3.1.5 *Всеки полином $s(x) = g(x)u(x) \in C$, където $(s(x), h(x)) = 1$ поражда $C = (g(x))$ в $\mathcal{F}_n$, но само $g(x)$ удовлетворява 1.*

Да напомним, че *реципрочен полином* $a^*(x)$ на полинома $a(x) = a_0 + \cdots + a_k x^k$ се нарича

$$a^*(x) = x^k a(x^{-1}) = a_0 x^k + a_1 x^{k-1} + \cdots + a_k.$$

Теорема 3.1.6 *Нека $\mathcal{C} = (g(x))$ е цикличен код с проверочен полином $h(x) = (x^n - 1)/g(x)$. Тогава дуалният код $\mathcal{C}^\perp$ се поражда от реципрочния на $h(x)$ полином $h^*(x)$ и $\mathcal{C}^\perp$ е еквивалентен на кода $(h(x))$.*

Доказателство. Проверочната матрица $\mathbf{H}$ на $\mathcal{C}$ може да бъде записана във вида

$$\mathbf{H} = \begin{pmatrix} \overleftarrow{h(x)} \\ \overleftarrow{xh(x)} \\ \vdots \\ \overleftarrow{x^{r-1}h(x)} \end{pmatrix} = \begin{pmatrix} \overrightarrow{x^{n-k-1}h^*(x)} \\ \overrightarrow{x^{n-k-2}h^*(x)} \\ \vdots \\ \overrightarrow{h^*(x)} \end{pmatrix},$$

където $r = n - k = n - \deg h(x) = \deg g(x)$. Очевидно $\operatorname{rank}(\mathbf{H}) = r = \deg g(x)$. Следователно $\mathbf{H}$ поражда дуалния код $\mathcal{C}^\perp$ и тъй като $h(x)g(x) = x^n - 1$, то $1 - x^n = x^{n-k} g(x^{-1}).x^k h(x^{-1})$, т.е. $h^*(x).g^*(x) = 1 - x^n$. Следователно $h^*(x) \mid (x^n - 1)$ и тъй като $\deg h^*(x) = k = n - \dim \mathcal{C}^\perp$, то $h^*(x)$ поражда $\mathcal{C}^\perp$.
Тъй като редовете на $\mathbf{H}$ се получават чрез записване кодовите думи на $(h(x))$ в обратен ред, то $\mathcal{C}^\perp = (h^*(x))$ и $(h(x))$ са еквивалентни.

◇

Нека $x^n - 1 = p_1(x) \ldots p_t(x)$ е разлагането на $x^n - 1$ в произведение на неразложими над основното поле F множители. Ако $(n, q) = 1$ тези множители са различни, а ако $(n, q) = p^s$, то $x^n - 1 = (x^{n_1} - 1)^{p^s}$, където $(n_1, q) = 1$. Оттук нататък ще предполагаме, че $(n, q) = 1$.

Дефиниция 3.1.7 *Цикличният код $(p_i(x))$ се нарича* ***максимален*** *и ще го отбелязваме с M_i^+. Съответно кодът породен от $(x^n - 1)/p_i(x)$ се нарича* ***минимален или още неразложим****.*

Названията на горните циклични кодове се обуславят от това, че те са съответно максимален и минимален идеал в $\mathcal{F}_n$.

Пример 3.1.1 Нека $\mathbb{F} = GF(2)$, $n = 7$. Тогава

$$x^7 - 1 = (x+1)(x^3 + x + 1)(x^3 + x^2 + 1),$$

т.е. $x^7 - 1 = (x+1)p(x)p^*(x)$, $p(x) = 1 + x + x^3$. Да разгледаме $C = (p(x))$. Проверочният му полином е

$$h(x) = (x+1)p^*(x) = x^4 + x^2 + x + 1$$

и следователно

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \quad \text{и} \quad \mathbf{H} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Лесно се вижда, че горните матрицаи са еквивалентни на пораждащата и проверочната матрица на [7, 4, 3] кода на Хеминг.

Скъсени циклични кодове. Скъсените циклични кодове се получават като се вземат кодовите думи имащи нули в последните, например s последователни позиции и отстраним тези нули. Полученият скъсен код е подмножество от полиноми на C от степен $\leq n - 1 - s$.

Упражнение 3.1.1 *Докажете следната теорема:*

Теорема 3.1.8 *Кодът C^* е скъсен цикличен код тогава и само тогава, когато C^* е идеал в $\mathbb{F}[x]/f(x)$ за някой полином $f(x)$.*

Кодиране на циклични кодове.

1. Един очевиден начин за кодиране на циклични кодове дават дефиницията им и Теорема 3.1.4. Нека $\mathcal{C}$ е $[n, k]$ цикличен код с пораждащ полином $g(x)$ от степен $\deg g(x) = r = n - k$. Всяко съобщение от k елемента може да се представи като полином $i(x) = i_0 + i_1 x + \cdots + i_{k-1}x^{k-1}$. Кодерът изобразява $i(x)$ в кодовата дума

$$c(x) = i(x)g(x).$$

Този метод, обаче, не е систематичен - няма съвкупност от k символа на кодовата дума, които да представят $i(x)$.

2. Сега ще опишем **систематичен метод за кодиране**.

Информационният блок $(i_0, i_1, \ldots, i_{k-1})$ се представя с полинома $x^r i(x) = i_0 x^r + \cdots + i_{k-1}x^{n-1}$, $r = n - r$. След това $x^r i(x)$ се разделя на $g(x)$: $x^r i(x) = q(x).g(x) + r(x)$, където $\deg r(x) < r$. Информационният блок се кодира с $c(x) = -r(x) + x^r i(x)$.

Упражнение 3.1.2 *Покажете, че описания алгоритъм за кодиране съответства на кодиране с пораждаща матрица*

$$\mathbf{G} = (\mathbf{A}|\,\mathbf{I}_k).$$

Редовете на $\mathbf{A}$ са остатъците на $x^r, x^{r+1}, \ldots, x^{n-1}$ при деление с $g(x)$. Проверочната матрица в такъв случай е $\mathbf{H} = (\mathbf{I}_r|\,\mathbf{A}^\tau)$ и изчисляването на синдрома $v\mathbf{H}^\tau$ по същество е кодиране на получения по канала вектор $v(x)$, т.е. синдромът $s(x)$ е остатъка от деление на $v(x) = q(x).g(x) + s(x)$ на пораждащия полином.

Вторият от описаните алгоритми се прилага без изменение и за скъсени циклични кодове, тъй като скъсяването отстранява само информационни символи.

3.2 Нули на код. БЧХ и Рид-Соломон кодове.

Както предположихме по-горе n и q са взаимно-прости: $(n, q) = 1$. Тогава $x^n - 1 = p_1(x) \ldots p_k(x)$ се разлага в произведение на различни неразложими полиноми и следователно всички корени на $x^n - 1$ са различни.

Нека m е мултипликативния порядък на q по модул n, т.е. m е най-малкото естествено число, такова че $n \mid (q^m - 1)$. Тогава разширението $GF(q^m)$ на $\mathbb{F} = GF(q)$ е поле на разлагане на $x^n - 1$.

Както е известно, корените на $x^n - 1$ образуват циклична група $\{1, \beta, \beta^2, \ldots, \beta^{n-1}\}$, където β е примитивен $n-$ти корен на единицата, например $\beta = \alpha^{\frac{q^m-1}{n}}$, където α е примитивен елемент на полето $GF(q^m)$.

Да напомним, че ако β^i е корен на неразложимия полином $p(x)$, то всички други корени на $p(x)$ имат вида β^{iq^s}. Следователно съществува взаимно-еднозначно съответствие между неразложимите над $\mathbb{F} = GF(q)$ делители на $x^n - 1$ и циклотомичните класове C_i относно q по модул n :

$$C_i = \{i, iq, \ldots, iq^{m_{i-1}}\},$$

където $iq^{m_i} \equiv i \pmod n$ и m_i е най-малкото естествено число с това свойство. Обикновенно минималното число в класа се взема като негов представител.

Дефиниция 3.2.1 *Нека $\mathcal{C} = (g(x))$ е цикличен код с пораждащ полином $g(x)$, където*

$$g(x) = \prod_{j \in K} (x - \beta^j), \quad K \subset \{0, 1, \ldots, n-1\}$$

(K се явява обединение от циклотомични класове).
Множеството $\{\beta^j | j \in K\}$ се нарича ***множество от нули на кода*** *$\mathcal{C}$. Допълнението му, т.е.*

$$\{\beta^j | j \notin K\}$$

се нарича множество от ***ненули****.*

Твърдение 3.2.2 *Векторът $c(x) \in \mathcal{C}$ тогава и само тогава, когато $c(\beta^j) = 0$ за всяко $j \in K$, т.е.*

$$C = (g(x)) \ = \ \{c(x) | c(\beta^j) = 0, \ j \in K\}.$$

Доказателство. Ако $c(x) \in \mathcal{C} \quad \Rightarrow \quad g(x)|c(x) \ \Rightarrow \ c(\beta^j) = 0.$
Обратно, ако $c(\beta^j) = 0$, то $g(x)|c(x)$.
$\diamond$

Твърдение 3.2.3 *γ е нула на $\mathcal{C}^\perp$ тогава и само тогава, когато γ^{-1} е ненула на $\mathcal{C}$.*

Доказателството оставяме за упражнение на читателя.

Ако $\{\alpha_1, \alpha_2, \ldots, \alpha_s\}$ е множеството от нули на $\mathcal{C}$, то $\mathcal{C} \equiv \{c \in \mathbb{F}^n \mid c\mathbf{H}^\tau = 0\}$, където

$$\mathbf{H} \ = \ \begin{pmatrix} 1 & \alpha_1 & \alpha_1^2 & \ldots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \ldots & \alpha_2^{n-1} \\ . & . & . & . & . \\ 1 & \alpha_s & \alpha_s^2 & \ldots & \alpha_s^{n-1} \end{pmatrix}.$$

Очевидно, достатъчно е да се вземат само тези α_j, които са корени на различни неразложими делители на $g(x)$, т.е. ние можем да зачеркнем редовете с α_j, които са q^λ степени на други α_i.

Теорема 3.2.4 (БЧХ граница) *Нека $\mathcal{C}$ е цикличен код с пораждащ полином $g(x)$ такъв, че за някои цели числа $b \geq 0$ and $\delta \geq 1$*

$$g(\beta^b) = g(\beta^{b+1}) = \cdots = g(\beta^{b+\delta-2}) = 0,$$

т.е. кодът има редица от $\delta - 1$ последователни степени на β като нули. Тогава минималното разстояние на кода е поне δ.

Доказателство. Ако $c = (c_0, c_1, \ldots, c_{n-1}) \in C$

$$c(\beta^b) = c(\beta^{b+1}) = \cdots = c(\beta^{b+\delta-2}) = 0$$

Следователно $c\mathbf{H}^\tau = 0$, където

$$\mathbf{H} = \begin{pmatrix} 1 & \beta^b & \beta^{2b} & \dots & \beta^{(n-1)b} \\ 1 & \beta^{b+1} & \beta^{2(b+1)} & \dots & \beta^{(n-1)(b+1)} \\ . & . & . & . & . \\ 1 & \beta^{b+\delta-2} & \beta^{2(b+\delta-2)} & \dots & \beta^{(n-1)(b+\delta-2)} \end{pmatrix}.$$

Ако $wt(w) = w$, то някои w стълба на $\mathbf{H}$ трябва да бъдат линейно зависими. Следователно, детерминантата образувана от тези стълбове и първите w реда трябва да бъде равна на нула. Обаче, това е невъзможно защото тя се представя като произведение на ненулев елемент от $GF(q^m)$ и детерминанта на Вандермонт от ред w.

$\diamond$

Кодове БЧХ (BCH: Bose-Chaudhudri-Hocquenghem codes.)

Дефиниция 3.2.5 *Цикличният код с блокова дължина n над $\mathbb{F} = GF(q)$ се нарича* ***БЧХ код с конструктивно разстояние*** *δ, ако за някое цяло число $b \geq 0$ пораждащият му полином $g(x)$ е най-малкото общо кратно от минималните полиноми на $\beta^b, \beta^{b+1}, \dots, \beta^{b+\delta-2}$, където β е примитивен $n-ти$ корен на единицата. Ако $n = q^m - 1$ БЧХ кода се нарича* ***примитивен****. Освен това най-често се избира $b = 1$ (****БЧХ код в тесен смисъл****)*

Пример 3.2.1 *БЧХ кодове поправящи две грешки.*

Нека $\mathbb{F} = GF(2)$, $n = 2^m - 1$ и α е примитивен елемент на $GF(2^m)$. Да разгледаме БЧХ код в тесен смисъл с конструктивно разстояние 5, т.е. с нули $\alpha, \alpha^2, \alpha^3, \alpha^4$. Тогава

$$g(x) = l.c.m.[M_1(x), M_3(x)] = M_1(x)M_3(x),$$

където $M_i(x) = irr_F\alpha^i$ е неразложимия полином на α^i. Тъй като $3 \neq 2^l \bmod n$, то $M_1(x) \neq M_3(x)$ и циклотомичните класове C_1 и C_3 са различни, но $|C_1| = |C_2| = m$. $\dim \mathcal{C} = n - \deg g(x) = 2^m - 1 - 2m$. Следователно, $\mathcal{C}$ има параметри $[2^m - 1,\ 2^m - 1 - 2m,\ d]$, като БЧХ границата дава $d \geq 5$. Сега да раазгледаме следната матрица:

$$\hat{\mathbf{H}} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \dots & \alpha^{n-1} \\ 1 & \alpha^3 & \alpha^6 & \dots & \alpha^{3(n-1)} \end{pmatrix}.$$

Тогава $c = (c_0, \ldots, c_{n-1})$ е от кода $\mathcal{C}$ тогава и само тогава, когато $c\hat{\mathbf{H}}^\tau = 0$. Ако заместим в $\hat{\mathbf{H}}$ всеки елемент от $GF(2^m)$ със съответната му m-орка от координатите му даден базис над $GF(2)$, ще получим $2m \times n$ матрица $\mathbf{H}$. Очевидно $c\mathbf{H}^\tau = 0$ тогава и само тогава, когато $c\hat{\mathbf{H}}^\tau = 0$. Следователно, $\mathbf{H}$ е проверочна матрица за $\mathcal{C}$.

Пример 3.2.2 *Кодът на Хеминг като цикличен.*

Твърдение 3.2.6 *Нека $n = \frac{q^m-1}{q-1}$, α е примитивен елемент на крайното поле $GF(q^m)$ и $\beta = \alpha^{q-1}$. q-ичният цикличен код*

$$\mathcal{C} \;=\; \{c(x) \in \mathcal{F}_n \mid c(\beta) = 0\}$$

има размерност $n - m$. Минималното му разстояние е 3 тогава и само тогава, когато $(m, q-1) = 1$.

Доказателство. Кодът C се поражда от $m(x) = irr_F\beta$. Мултипликативният порядък q по модул $n > q^{m-1}$ е поне m, а $q^m \equiv 1 (mod\ n)$. В такъв случай той е точно m, т.е. $\deg m(x) = m$. Следователно $\dim C = n - \deg m(x) = n - m$.

Кодът C се състои от всички вектори $c = (c_0, \ldots, c_{n-1})$ с $cH^t = 0$, където

$$H \;=\; \begin{pmatrix} 1 & \beta & \beta^2 & \ldots & \beta^{n-1} \end{pmatrix}.$$

Минималното разстояние е 3, точно когато всеки два стълба са линейно независими, т.е. тогава и само тогава, когато $\beta^{i-j} \notin F$, което е еквивалентно с $(\beta^{i-j})^{q-1} \neq 1$ за всеки $i \neq j$. С други думи тогава и само тогава, когато n не дели $(i-j)(q-1)$ за всеки $i \neq j$. Последното, обаче, е в сила, точно когато $(n, q-1) = 1$ и тъй като $n = q^{m-1} + \cdots + q + 1 = (q^{m-1} - 1) + \cdots + (q-1) + m$, то е еквивалентно с $(m, q-1) = 1$.

$\diamond$

Да отбележим, че в двоичния случай $(m, 1) = 1$, т.е. в двоичния случай кодът на Хеминг е еквивалентен на цикличен. Най-малките параметри, за които недвоичен код на Хеминг не може да бъде представен като цикличен са $q = 4,\ n = 21,\ k = 18 (m = 3)$.

Кодове на Рид-Соломон (Reed-Solomon).

Дефиниция 3.2.7 ***Код на Рид-Соломон*** *над* $\mathbb{F} = GF(q)$ *се нарича БЧХ код с дължина* $N = q - 1$.

Нека α е примитивен елемент на $\mathbb{F} = GF(q)$. Тогава кодът на Рид-Соломон с конструктивно разстояние δ има пораждащ полином

$$g(x) = (x - \alpha^b)(x - \alpha^{b+1}) \dots (x - \alpha^{b+\delta-2}).$$

Тъй като $\deg g(x) = \delta - 1$ (и блоковата дължина е $N = q-1$), то размерността му е $K = N - \delta + 1$. Съгласно БЧХ границата, минималното разстояние $D \geq \delta$. Но $\delta = N - K + 1$ и границата на Синглинтон $D = N - K + 1$. Следователно $\delta = D$ и всеки Рид-Соломон код е MDS код с параметри

$$[N,\ K,\ N - K + 1].$$

3.3 Декодиране на циклични кодове.

Нека $\mathcal{C} = (g(x))$ е цикличен код над $\mathbb{F} = GF(q)$ изправящ до t грешки. Нека след предаване по канала на кодовата дума $c(x)$ е получен вектор $v(x) = c(x) + e(x)$, където $e(x) = e_{i_1}x^{i_1} + e_{i_2}x^{i_2} + \dots + e_{i_w}x^{i_w}$. Синдромът $s(x) = [v(x)]_{g(x)} = [e(x)]_{g(x)}$ определя еднозначно $e(x)$, ако броят на грешките $w \leq t$. С $[f(x)]_{g(x)}$ бележим остатъка на $f(x)$ по модул $g(x)$.

3.3.1 Декодер на Мегит.

Описваният по-долу декодер е алгоритъм за декодиране на произволни циклични кодове и се основава единствено на свойството цикличност.

Идея: Вместо да помним синдромите за всички $e(x)$ с тегло ненадминаващо t запомняме само техните циклични представители, например тези с $e_{n-1} \neq 0$.

Алгоритъм:

1. При получаване на $v(x)$ изчислява $s(x) = [v(x)]_{g(x)}$;
2. Сравнява с таблицата на синромите:
   -ако $s(x)$ е в таблицата, декодерът определя $e(x)$;
   -ако не, преминава към точка 3.

3. Изчислява $[xs(x)]_{g(x)}$ и преминава към точка 2. Ако процедурата е повторена n пъти и няма синдром в таблицата, то декодерът дава информация, че са станали повече от t грешки.

Обосновката на алгоритъма се дава със следната лема и следствието от нея.

Лема 3.3.1 *Ако* $v(x) \equiv s(x) \ (mod\ g(x))$ *в* $\mathbb{F}[x]$, *то за всеки полином* $f(x) \in \mathbb{F}[x]$ *е в сила* $[f(x)v(x)]_{x^n-1} \equiv f(x)s(x) \ (mod\ g(x))$.

Доказателство. Умножавайки даденото сравнение по $f(x)$ получаваме
$f(x)v(x) \equiv f(x)s(x) \ (mod\ g(x))$ в $\mathbb{F}[x]$. От друга страна

$$f(x)v(x) = (x^n-1)q(x)+[f(x)v(x)]_{x^n-1} = (q(x)h(x))g(x)+[f(x)v(x)]_{x^n-1}.$$

Следователно $[f(x)v(x)]_{x^n-1} \equiv f(x)s(x) \ (mod\ g(x))$. ◇

Следствие 3.3.2 *Ако* $s(x) = [v(x)]_{g(x)}$ *е синдром на* $v(x)$, *то* $xv(x)$ *има синдром* $[xs(x)]_{g(x)}$.

Модифициран декодер на Мегит.

Вместо декодерът по $s(x)$ да дава като резултат $e(x)$ той определя само дали $e_{n-1} \neq 0$. В такъв случай не е необходимо да съхраняваме цикличните представители с $e_{n-1} \neq 0$, а само синдромите им. След това поправя $v(x)$ в $v'(x) = v(x) - e_{n-1}x^{n-1}$ и за него повтаря процедурата. Да отбележим, че $v'(x) = c(x) + e'(x)$, където $e'(x)$ има тегло с единица по-малко от това на $e(x)$. Процесът продължава докато се получи нулев синдром.

Ако вместо синдрома $[v(x)]_{g(x)}$ се вземе $s(x) = [x^r v(x)]_{g(x)}$, където $r = n-k$ е броя на проверочните символи, то реализацията на декодера ще има следните предимства:

1. За изчисляване на синдрома може да се използва устройството за кодиране;

2. Няма нужда да се изчислява синдромът $s'(x)$, а само се променя старшият разряд, тъй като

$$s'(x) = [x^r(v(x) - e_{n-1}x^{n-1})]_{g(x)} = [x^r v(x) - e_{n-1}x^{n+r-1}]_{g(x)} =$$
$$= s(x) - [e_{n-1}x^{r-1}]_{g(x)} = s(x) - e_{n-1}x^{r-1}.$$

3.3.2 Декодиране чрез нулите на кода.

Нека пораждащият полином $g(x)$ на цикличния код $\mathcal{C}$ има за нули $\beta^{k_1}, \beta^{k_2}, \ldots, \beta^{k_r}$, където β е $n-$ти примитивен корен на единицата и $\beta \in GF(q^m)$, като m е мултипликативния порядък на q по модул n. За синдрома $s(x)$ на получения вектор $v(x)$ имаме

$$\begin{aligned} s(\beta^{k_j}) = v(\beta^{k_j}) = e(\beta^{k_j}) &= e_{i_1}\beta^{i_1 k_j} + e_{i_2}\beta^{i_2 k_j} + \cdots + e_{i_w}\beta^{i_w k_j} \\ &= E_1 X_1^{k_j} + E_2 X_2^{k_j} + \cdots + E_w X_w^{k_j}, \end{aligned}$$

където сме положили $X_l = \beta^{i_l}$ и $E_l = e_{i_l}$. Да означим с S_k следните суми:

$$S_k = s(\beta^k) = E_1 X_1^k + E_2 X_2^k + \cdots + E_w X_w^k,$$

които ще наричаме ***обобщени степенни сборове***. Под ***полином на локаторите на грешките*** ще разбираме полинома

$$\sigma(x) = (x - X_1)(x - X_2)\ldots(x - X_w) = x^w - \sigma_1 x^{w-1} + \sigma_2 x^{w-2} - \cdots + (-1)^w \sigma_w.$$

Много често вместо $\sigma(x)$ се използва неговия реципрочен

$$\Lambda(x) = \prod_{i=1}^{w}(1 - X_i x) = 1 - \sigma_1 x + \sigma_2 x^2 - \cdots + (-1)^w \sigma_w x^w,$$

който също се нарича полином на локаторите на грешките.

Тъй като X_i са корени на $\sigma(x)$, то

$$X_i^w - \sigma_1 X^{w-1} + \sigma_2 X^{w-2} - \cdots + (-1)^w \sigma_w = 0.$$

Полагайки $a_i = (-1)^{i-1}\sigma_i,\ i = 1, \ldots, w$ горното равенство добива вида

$$X_i^w - a_1 X^{w-1} - a_2 X^{w-2} - \cdots - a_w = 0 \tag{3.1}$$

Умножавайки (3.1) по $E_i X_i^j,\ j = 0, 1, 2, \ldots, n-1$, и събирайки получените равенства за $i = 1, 2, \ldots, w$ получаваме

$$S_{j+w} = a_1 S_{j+w-1} + a_2 S_{j+w-2} + \cdots + a_w S_j,\ \ j = 0, 1.2, \ldots, n-1. \tag{3.2}$$

Следователно редицата $\{S_j\}$ удовлетворява линейната рекурентна връзка зададена с (3.2).

И така задачата за декодиране, т.е. за намиране на позициите i_j и стойностите e_{i_j} на грешките, се свежда до следната задача:

Знаейки редицата $S_{k_1}, S_{k_2}, \ldots, S_{k_r}$ ***да се намери полином*** $\sigma(x)$ ***от възможно най-ниска степен*** w ***така, че редицата да удовлетворява (3.2), т.е. да се намери рекурентна връзка от възможно най-ниска степен за тази редица.***

Такъв подход съответства на декодиране по максимално правдоподобие.

Горната задача играе ключова роля в алгоритъма изложен по-долу (представлява изпълнението на точка 2). Според това как тя се решава може да разграничим три метода за декодиране:
- алгоритъм на Питерсон-Горенщайн-Цирлер;
- алгоритъм на Евклид;
- алгоритъм на Берлекемп-Месси.

В най-общ план алгоритъмът за декодиране изглежда така:
Алгоритъм за декодиране:

1. По получения вектор $v(x)$ определяме $S_b, S_{b+1}, \ldots, S_{b+\delta-2}$ като $S_j = v(\beta^j)$,
   $j = b, b+1, \ldots, b+\delta-2$.

2. Намираме полинома на локаторите $\sigma(x)$.

3. Намираме корените $X_1, X_2, \ldots, X_w$ на полинома на локаторите.

4. От $X_j = \beta^{i_j}$ определяме позициите на грешките $i_1, \ldots, i_w$.

5. Решаваме линейната система

$$\begin{pmatrix} X_1^b & X_2^b & \ldots & X_w^b \\ \vdots & \vdots & \vdots & \\ X_1^{b+w-1} & X_2^{b+w-1} & \ldots & X_w^{b+w-1} \end{pmatrix} \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_w \end{pmatrix} = \begin{pmatrix} S_b \\ S_{b+1} \\ \vdots \\ S_{b+w-1} \end{pmatrix}$$

   за да определим стойностите на грешките $e_{i_1}, \ldots, e_{i_w}$.

6. Накрая определяме $c(x) = v(x) - e(x)$.

Алгоритъм на Питерсон-Горенщайн-Цирлер за декодиране на БЧХ кодове.

Нека $\mathcal{C}$ е $q-$ичен БЧХ код с блокова дължина n зададен с множеството от нули $\beta^b, \beta^{b+1}, \ldots, \beta^{b+\delta-2}$, където $\delta = 2t+1$ или $2t+2$, т.е C е конструиран

да изправя до t грешки. В такъв случай $\delta - 1$ последователни члена на редицата $\{S_j\}$ от обобщени степенни сборове са известни и равенства (3.2) показват, че

$$\begin{pmatrix} S_b & S_{b+1} & \dots & S_{b+w-1} \\ S_{b+1} & S_{b+2} & \dots & S_{b+w} \\ \vdots & \vdots & \vdots & \vdots \\ S_{b+\delta-2-w} & S_{b+\delta-1-w} & \dots & S_{b+\delta-3} \end{pmatrix} \begin{pmatrix} a_w \\ a_{w-1} \\ \vdots \\ a_1 \end{pmatrix} = \begin{pmatrix} S_{b+w} \\ S_{b+w+1} \\ \vdots \\ S_{b+\delta-2} \end{pmatrix}. \tag{3.3}$$

Матрицата на (3.3) има $\delta - 1 - w \geq w$ реда за всяко $w \leq t$, тъй като $\delta \geq 2t + 1 \geq 2w + 1$.

Да разгледаме системата

$$\begin{pmatrix} S_b & S_{b+1} & \dots & S_{b+r-1} \\ S_{b+1} & S_{b+2} & \dots & S_{b+r} \\ \vdots & \vdots & \vdots & \vdots \\ S_{b+\delta-2-r} & S_{b+\delta-1-r} & \dots & S_{b+\delta-3} \end{pmatrix} \begin{pmatrix} x_r \\ x_{r-1} \\ \vdots \\ x_1 \end{pmatrix} = \begin{pmatrix} S_{b+r} \\ S_{b+r+1} \\ \vdots \\ S_{b+\delta-2} \end{pmatrix} \tag{3.4}$$

и с $\mathbf{A}_{pq}$ означим матрицата

$$\mathbf{A}_{pq} = \begin{pmatrix} S_b & S_{b+1} & \dots & S_{b+q} \\ S_{b+1} & S_{b+2} & \dots & S_{b+q+1} \\ \vdots & \vdots & \vdots & \vdots \\ S_{b+p} & S_{b+p+1} & \dots & S_{b+p+q} \end{pmatrix}.$$

Ако системата (3.4) има решение за някое $r < w$, това ще означава, че има два или повече вектора на разстояние ненадминаващо $w \leq t = \lfloor(\delta-1)/2\rfloor$, което е невъзможно. Следователно (3.4) е несъвместима за $r < w$, а при $r = w$ има единствено решение $(a_w, \dots, a_1)$ определено от (3.3). Това означава, че

$$\operatorname{rank}(\mathbf{A}_{\delta-r-2,r-1}) < \operatorname{rank}(\mathbf{A}_{\delta-r-2,r}), \quad \text{за} \quad r < w,$$

$$\operatorname{rank}(\mathbf{A}_{\delta-w-2,w-1}) = \operatorname{rank}(\mathbf{A}_{\delta-w-2,w}) = w.$$

Ако $w < r \leq t$, то матрицата и разширената матрица на (3.4) са от вида $\mathbf{A}_{pq}$ с $p \geq w$ и $q \geq w$ и използвайки (3.2) с елементарни преобразувания по редове и стълбове се трансформират в

$$\mathbf{A_{pq}} \sim \left(\begin{array}{c|c} \mathbf{A}_{w-1,w-1} & O \\ \hline O & O \end{array}\right) = \left(\begin{array}{c|c} WDW^\tau & O \\ \hline O & O \end{array}\right),$$

където $W = W(X_1, \ldots, X_w)$ е матрицата на Вандермонт от $X_1, \ldots, X_w$, а D е диагоналната $w \times w$ матрица с елементи $E_j X_j^b$ по диагонала, т.е.

$$D = \begin{pmatrix} E_1 X_1^b & & & \\ & E_2 X_2^b & & \\ & & \ddots & \\ & & & E_w X_w^b \end{pmatrix}.$$

Следователно и двете матрици имат един и същ ранг w, което показва, че системата е съвместима като при това рангът не нараства за $r \geq w$. Очевидно $(0, \ldots, 0, a_w, \ldots, a_1)$ е едно частно решение на (3.4), а

$$(a_w, \ldots, a_1, -1, 0, \ldots, 0);\ (0, a_w, \ldots, a_1, -1, \ldots, 0); \ldots; (0, \ldots, 0, a_w, \ldots, a_1, -1)$$

образуват базис на пространството от решения на съответната хомогенна система. Следователно произволно решение на (3.4) е сума на частното решение с тяхна линейна комбинация. Но всяка такава $r-$орка води до локаторен полином от вида

$$f(x)(a_w + a_{w-1}x + \cdots + a_1 x^{w-1} - x^w),$$

където $f(x)$ е произволен унитарен полином от степен $r - w$. Следователно всяко решение на системата води до същите позиции на грешките.

Направените бележки ни позволяват да формулираме по-подробно действията по точка 2 от алгоритъма.

Алгоритъм на Питерсон-Горенщайн-Цирлер:

2.1. Определяме ранга w на матрицата

$$\begin{pmatrix} S_b & S_{b+1} & \ldots & S_{b+t} \\ S_{b+1} & S_{b+2} & \ldots & S_{b+t+1} \\ \vdots & \vdots & \vdots & \vdots \\ S_{b+\delta-t-2} & S_{b+\delta-r} & \ldots & S_{b+\delta-2} \end{pmatrix}$$

(при $\delta = 2t + 2$ рангът може да бъде и $t + 1$ - в този случай са станали повече от t грешки и не можем да декодираме правилно.)

2.2. Решаваме системата (3.4) или по точно еквивалентната ѝ

$$\begin{pmatrix} S_b & S_{b+1} & \ldots & S_{b+w-1} \\ S_{b+1} & S_{b+2} & \ldots & S_{b+r} \\ \vdots & \vdots & \vdots & \vdots \\ S_{b+w-1} & S_{b+w} & \ldots & S_{b+2w-2} \end{pmatrix} \begin{pmatrix} x_w \\ x_{w-1} \\ \vdots \\ x_1 \end{pmatrix} = \begin{pmatrix} S_{b+w} \\ S_{b+w+1} \\ \vdots \\ S_{b+2w-1} \end{pmatrix}.$$

2.3. По намереното решение $(a_w, \ldots, a_1)$ построяваме полинома на локаторите

$$x^w - a_1 x^{w-1} - a_2 x^{w-2} - \cdots - a_w.$$

Пример 3.3.1 Нека $\mathcal{C}$ е $[15, 5, 7]$ БЧХ код над $GF(2)$ с нули $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6$, където α е примитивен елемент на $GF(2^4)$ с неразложим полином x^4+x+1. Елементите на полето в мултипликативно и адитивно представяне са дадени в таблица 3.3.1

Степен на α	Полином от α
0	0000
1	1000
α	0100
α^2	0010
α^3	0001
α^4	1100
α^5	0110
α^6	0011
α^7	1101
α^8	1010
α^9	0101
α^{10}	1110
α^{11}	0111
α^{12}	1111
α^{13}	1011
α^{14}	1001

Таблица 3.1: Елементите на полето $GF(16)$.

Пораждащият полином на $\mathcal{C}$ е

$$g(x) = (x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1) = x^{10}+x^8+x^5+x^4+x^2+x+1$$

и кодът изправя три грешки, т.е. $t = 3$.

Нека е получен векторът $v(x) = x^{10} + x^8 + x^7 + x^5 + x^4 + x + 1 = g(x) + x^2 + x^7$.

1. Пресмятаме (ползвайки таблица 3.3.1) $S_j = v(\alpha^j)$:

$$S_1 = \alpha^2 + \alpha^7 = \alpha^{12}, \quad S_2 = \alpha^4 + \alpha^{14} = \alpha^9, \quad S_3 = \alpha^6 + \alpha^6 = 0,$$
$$S_4 = \alpha^8 + \alpha^{13} = \alpha^3, \quad S_5 = \alpha^{10} + \alpha^5 = 1, \quad S_6 = \alpha^{12} + \alpha^{12} = 0.$$

2. Определяме ранга на матрицата

$$\begin{pmatrix} S_1 & S_2 & S_3 & S_4 \\ S_2 & S_3 & S_4 & S_5 \\ S_3 & S_4 & S_5 & S_6 \end{pmatrix} = \begin{pmatrix} \alpha^{12} & \alpha^9 & 0 & \alpha^3 \\ \alpha^9 & 0 & \alpha^3 & 1 \\ 0 & \alpha^3 & 1 & 0 \end{pmatrix}.$$

Лесно се вижда, че е 2. Следователно $w = 2$, т.е. станали са две грешки.

3. Решаваме системата

$$\begin{pmatrix} \alpha^{12} & \alpha^9 \\ \alpha^9 & 0 \end{pmatrix} \begin{pmatrix} x_2 \\ x_1 \end{pmatrix} = \begin{pmatrix} 0 \\ \alpha^3 \end{pmatrix}$$

Решението е (α^9, α^{12}). Следователно

$$\sigma(x) = x^2 + \alpha^{12}x + \alpha^9.$$

4. Намираме корените на $\sigma(x)$. Те са

$$X_1 = \alpha^2$$

и $X_2 = \alpha^7$. Следователно векторът грешка е

$$e(x) = x^2 + x^7,$$

тъй като кодът е двоичен и няма нужда да изчисляваме стойностите на грешките - те са единици.

5. Изпратената кодова дума е $g(x)$.

3.4 Декодиране с алгоритъм на Евклид.

Нека (запазвайки означенията от предния параграф) за получения вектор $v(x)$ са известни $2t$ последователни синдрома: $S_b, S_{b+1}, \ldots, S_{b+2t-1}$ и положим

$$S(x) = S_b + S_{b+1}x + \ldots + S_{b+2t-1}x^{2t-1}.$$

Тогава

$$S(x) = \sum_{j=0}^{2t-1} \left(\sum_{i=1}^{w} E_i X_i^{b+j} \right) x^j = \sum_{i=1}^{w} \left(E_i X_i^b + E_i X_i^{b+1} x + \cdots + E_i X_i^{b+2t-1} x^{2t-1} \right)$$
$$= \sum_{i=1}^{w} E_i X_i^b \left(1 + X_i x + \cdots + X_i^{2t-1} x^{2t-1} \right) = \sum_{i=1}^{w} E_i X_i^b \frac{1 - X_i^{2t} x^{2t}}{1 - X_i x}.$$

Нека

$$\Lambda(x) = \prod_{i=1}^{w} (1 - X_i x) = 1 + a_1 x + a_2 x^2 + \cdots + a_w x^w$$

е реципрочния локаторен полином. Тогава е изпълнено

$$\begin{aligned} S(x)\Lambda(x) &= \sum_{i=1}^{w} E_i X_i^b \left(1 - X_i^{2t} x^{2t} \right) \frac{\Lambda(x)}{1 - X_i x} \\ &= \sum_{i=1}^{w} E_i X_i^b \prod_{i \neq j}^{w} (1 - X_j x) - \sum_{i=1}^{w} E_i X_i^{b+2t} \prod_{i \neq j}^{w} (1 - X_j x).x^{2t} \\ &= \omega(x) - u(x).x^{2t}. \end{aligned}$$

Следователно

$$S(x)\Lambda(x) + u(x).x^{2t} = \omega(x), \tag{3.5}$$

където

$$\begin{aligned} \omega(x) &\stackrel{\text{def}}{=} \sum_{i=1}^{w} E_i X_i^b \prod_{i \neq j}^{w} (1 - X_j x), \qquad \deg \omega(x) = w - 1 < t, \\ u(x) &\stackrel{\text{def}}{=} \sum_{i=1}^{w} E_i X_i^{b+2t} \prod_{i \neq j}^{w} (1 - X_j x), \qquad \deg u(x) = w - 1 < t, \end{aligned} \tag{3.6}$$

Да отбележим, че $(u(x), \Lambda(x)) = 1$.

Теорема 3.4.1 *Ако са известни $\Lambda(x)$ и $\omega(x)$, то стойностите на грешките се намират по формулата:*

$$E_k = -\frac{\omega(X_k^{-1})}{X_k^{b-1} \Lambda'(X_k^{-1})}.$$

Доказателство. От определението за реципрочния локаторен полином получаваме

$$\Lambda'(x) = \sum_{j=1}^{w}(-X_j)\prod_{i\neq j}^{w}(1 - X_i x).$$

Следователно

$$\Lambda'(X_k^{-1}) = -X_k \prod_{i\neq j}^{w}\left(1 - \frac{X_i}{X_k}\right).$$

Аналогично се получава

$$\omega(X_k^{-1}) = E_k \prod_{i\neq k}^{w}\left(1 - \frac{X_i}{X_k}\right),$$

откъдето получаваме и формулата. ◇

Лема 3.4.2 *Ако съществува представяне*

$$a(x)x^{2t} + b(x)S(x) = \omega(x), \tag{3.7}$$

с $\deg\omega(x) < t$ *и* $\deg b(x) \leq t$, *то то е пропорционално на единственото такова представяне с* $(a(x), b(x)) = 1$.

Доказателство. Да отбележим първо, че ако съществува представянето (3.7), то трябва да е изпълнено $\deg a(x) < t$. Нека $(x^{2t}, S(x)) = x^d$. Тогава $(x^{2t-d}, s(x)) = 1$, където $s(x) = S(x)/x^d$ и (3.7) се записва във вида

$$a(x)x^{2t-d} + b(x)s(x) = \omega'(x), \tag{3.8}$$

където $\omega'(x) = \omega(x)/x^d$ и $\deg\omega'(x) < t - d$.

Да предположим, че съществуват две различни представяния от вида (3.8):

$$\begin{aligned} &a_1(x)x^{2t-d} + b_1(x)s(x) = \omega_1(x), \qquad \deg\omega_1(x) < t-d, \quad \deg b_1(x) \leq t,\\ &a_2(x)x^{2t-d} + b_2(x)s(x) = \omega_2(x), \qquad \deg\omega_2(x) < t-d, \quad \deg b_2(x) \leq t. \end{aligned}$$

Нека $\omega(x) = (\omega_1(x), \omega_2(x))$ и положим $f(x) = \omega_2(x)/\omega(x)$ и $g(x) = \omega_1(x)/\omega(x)$. Очевидно $(f(x), g(x)) = 1$. Умножавайки първото от

горните равенства с $f(x)$, а второто с $g(x)$ и изваждайки ги, получаваме

$$x^{2t-d}\left[f(x)a_1(x) - g(x)a_2(x)\right] + s(x)\left[f(x)b_1(x) - g(x)b_2(x)\right] = 0.$$

Тъй като $(x^{2t-d}, s(x)) = 1$, то можем да заключим, че

$$\begin{aligned} f(x)a_1(x) &\equiv g(x)a_2(x) \quad (\mod s(x)) \\ f(x)b_1(x) &\equiv g(x)b_2(x) \quad (\mod x^{2t-d}). \end{aligned}$$

Но степените на $f(x)b_1(x)$ и $g(x)b_2(x)$ са строго по-малки от $(t-d)+t = 2t-d$. Следователно

$$f(x)b_1(x) = g(x)b_2(x) \quad \text{и} \quad f(x)a_1(x) = g(x)a_2(x).$$

Но тогава

$$f(x) \mid b_2(x), \quad g(x) \mid b_1(x) \quad \text{и} \quad f(x) \mid a_2(x), \quad g(x) \mid a_1(x).$$

Следователно,

$$\begin{aligned} b_1(x) &= g(x)b(x), \qquad b_2(x) = f(x)b(x) \\ a_1(x) &= g(x)a(x), \qquad a_2(x) = f(x)a(x). \end{aligned}$$

Последното показва, че и двете равенства са пропорционални на

$$a(x)x^{2t-d} + b(x)s(x) = \omega(x)$$

с коефициенти на пропорционалност съответно $g(x)$ и $f(x)$. При това $\deg\omega(x) < t-d$ и $\deg b(x) \leq t$. Ако $a(x)$ и $b(x)$ не са взаимнопрости, като разделим на най-големия им общ делител ще получим равенство от искания тип.

◇

Ще покажем, че $a(x), b(x)$ и $\omega(x)$ винаги съществуват и се получават на даден етап от изпълнението на алгоритъма за намиране на най-голям общ делител. Но преди да пристъпим към доказателството на този резултат да си припомним самия алгоритъм и някои факти свързани с него.

За да намерим най-голям общ делител на полиномите $f(x)$ и $g(x)$ извършваме следните последователни деления с остатък:

$$\begin{array}{rcll} f(x) & = & g(x)q_1(x) + r_1(x), & \deg r_1 < \deg g \\ g(x) & = & r_1(x)q_2(x) + r_2(x), & \deg r_2 < \deg r_1 \\ r_1(x) & = & r_2(x)q_3(x) + r_3(x), & \deg r_3 < \deg r_2 \\ \cdots & \cdots & \cdots\cdots\cdots\cdots\cdots\cdots & \cdots\cdots\cdots\cdots\cdots \\ r_{n-3}(x) & = & r_{n-2}(x)q_{n-1}(x) + r_{n-1}(x), & \deg r_{n-1} < \deg r_{n-2} \\ r_{n-2}(x) & = & r_{n-1}(x)q_n(x). & \end{array}$$

Тъй като степените на остатъците строго намаляват $\deg r_1 > \deg r_2 > \cdots > \deg r_{n-1}$, то съществува номер n, такъв че $r_n(x) = 0$. Последният ненулев остатък (по-точно полиномът със старши коефициент 1 пропорционален на него) е най-голям общ делител $(f(x), g(x))$.

Да положим

$$\begin{aligned} a_0(x) = 0, \quad a_1(x) = 1, \quad a_j(x) &\stackrel{\text{def}}{=} a_{j-2}(x) - q_j(x)a_{j-1}(x) \\ b_0(x) = 1, \quad b_1(x) = -q_1(x), \quad b_j(x) &\stackrel{\text{def}}{=} b_{j-2}(x) - q_j(x)b_{j-1}(x). \end{aligned}$$

Ще ползваме и означението $[a(x)/b(x)]$ за частното на два полинома $a(x)$ и $b(x)$, т.е. при това означение $q_j(x) = [r_{j-2}(x)/r_{j-1}(x)]$.

Лема 3.4.3 *В сила са следните свойства:*

(1) $r_j(x) = f(x)a_j(x) + g(x)b_j(x)$;

(2) $a_{j-1}(x)b_j(x) - a_j(x)b_{j-1}(x) = (-1)^j$;

(3) $r_{j-1}(x)a_j(x) - r_j(x)a_{j-1}(x) = (-1)^j g(x)$;

(4) $r_{j-1}(x)b_j(x) - r_j(x)b_{j-1}(x) = (-1)^j f(x)$.

> ***Доказателство.*** Равенствата могат лесно да се докажат с метода на математическата индукция. Директната проверка показва, че са в сила за $j = 1, 2$. Предполагаме, че твърденията са верни за стойности $< j$ и ще покажем валидността им за j. Проверката ще извършим само за (2), като оставяме за читателя останалите случаи.
>
> $$\begin{aligned} & a_{j-1}(x)b_j(x) - a_j(x)b_{j-1}(x) \\ &= a_{j-1}(x)\left(b_{j-2}(x) - q_j(x)b_{j-1}(x)\right) - \left(a_{j-2}(x) - q_j(x)a_{j-1}(x)\right)b_{j-1}(x) \\ &= -\left[a_{j-2}(x)b_{j-1}(x) - a_{j-1}(x)b_{j-2}(x)\right] = -(-1)^{j-1} = (-1)^j. \end{aligned}$$
>
> $\diamond$

При $j = n - 1$ получаваме полиномите $a(x)$ и $b(x)$ с помощта, на които се представя най-големият общ делител $d(x) = f(x)a(x) + g(x)b(x)$.

Реализация на алгоритъма:
Данни: $f(x)$ и $g(x)$ полиноми, $\deg f \geq \deg g$.
Резултат: $d(x) = (f(x), g(x))$, $a(x)$, $b(x)$ полиноми.
Променливи: $A = (A_1, A_2, A_3)$, $B = (B_1, B_2, B_3)$ и $C = (C_1, C_2, C_3)$ са три

масива, които ще се изменят в процеса на изпълнение на програмата; $q(x)$ е полином.
$A := (f(x), 1, 0),\ B := (g(x), 0, 1),\ C := (1, 0, 0).$
while $C_1 \neq 0$ do
$q(x) := [A_1/B_1],\ C := A - qB, \quad A := B, \quad B := C$
else
$d(x) := A_1,\ a(x) := A_2,\ b(x) := A_3.$

Теорема 3.4.4 *Съществуват еднозначно определени полиноми* $u(x), \Lambda(x)$ *и* $\omega(x)$, *такива че*

$$u(x).x^{2t} + \Lambda(x)S(x) = \omega(x), \tag{3.9}$$

$\deg \omega(x) < t,\ \deg \Lambda(x) \leq t,\ u\ (u(x), \Lambda(x)) = 1.$

Доказателство. Прилагаме алгоритъма на Евклид за полиномите x^{2t} и $S(x)$ докато получим $r_j(x)$ със степен $\deg r_j(x) < t$. Тогава стойностите на C_2 и C_3 ще бъдат съответно $u(x)$ и $\Lambda(x)$. От $(1, -q_1(x)) = 1$ и равенство (2) на лема 3.4.3 получаваме $(u(x), \Lambda(x)) = 1$.

Единствеността следва от лема 3.4.2.

◇

Алгоритъм за декодиране.

Точка 2 от алгоритъма изглежда така:
Данни: x^{2t} и $S(x)$.
Резултат: $u(x),\ \Lambda(x)$.
Променливи: $A = (A_1, A_2, A_3),\ B = (B_1, B_2, B_3)$ и $C = (C_1, C_2, C_3)$ са три масива, които ще се изменят в процеса на изпълнение на програмата; $q(x)$ е полином.
$A := (x^{2t}, 1, 0),\ B := (S(x), 0, 1),\ C := (x^t, 0, 0).$
while $\deg C_1 \geq t$ do
$q(x) := [A_1/B_1],\ C := A - qB, \quad A := B, \quad B := C$
else
$\omega(x) := B_1,\ u(x) := B_2,\ \Lambda(x) := B_3.$
Точка 5 добива вида:

$$E_k = -\frac{\omega(X_k^{-1})}{X_k^{b-1}\Lambda'(X_k^{-1})}. \tag{3.10}$$

Пример 3.4.1 Да разгледаме кода от пример 3.3.1 и предположим, че е получен същият вектор $v(x) = g(x) + x^2 + x^7$. Тогава

$$S(x) = \alpha^{12} + \alpha^9 x + \alpha^3 x^3 + x^4.$$

Пресмятанията си по алгоритъма на Евклид записваме за удобство в таблица с четири стълба. Първите три колони на всеки три последователни реда представляват текущите стойности на тройките A, B, C (началното състояние на регистъра C не го записваме), а последният стълб (от втория ред нататък) съдържа текущото състояние на частното q.

x^6	1	0	$q(x)$
$S(x)$	0	1	$x^2 + \alpha^3 x - \alpha^6$
$x + (\alpha^3 - 1)$	1	$-(x^2 + \alpha^3 x - \alpha^6)$	

Следователно $\omega(x) = x + (\alpha^3 - 1) = x + \alpha^{14}$; $u(x) = 1$; $\Lambda(x) = x^2 + \alpha^3 x - \alpha^6$. За корените на $\Lambda(x)$ получаваме $X_1 = \alpha^{13}$, $X_2 = \alpha^8$. Следователно, локаторите на грешките са α^2 и α^7, т.е. грешките са пред x^2 и x^7.
За стойността на грешките получаваме съответно $E_1 = 1$, $E_2 = 1$.

3.5 Допълнителни задачи към Глава 3.

Problem 3.5.1 *Докажете, че ако $g(x) = g_0 + g_1 x + \cdots + g_r x^r$ е пораждащ полином на цикличен код, то свободният му член $g_0 \neq 0$.*

Problem 3.5.2 *Нека $\mathcal{C} = (g(x))$ е цикличен код над $GF(q)$ с блокова дължина n, $(n, q) = 1$. Докажете, че векторът $\mathbf{1} = (1, 1, \ldots, 1) \in \mathcal{C}$ тогава и само тогава, когато $x - 1$ не дели $g(x)$.*

Problem 3.5.3 *Нека $g(x) = m_1(x)$ е пораждащ полином на $[2^r - 1,\ 2^r - r - 1, 3]$ кода на Хеминг $\mathcal{H}_r$. Покажете, че $((x-1)g(x))$ е $[2^r - 1,\ 2^r - r - 2, 4]$ код, изправящ единични грешки и всяка конфигурация от две съседни грешки, т.е. $e(x) = x^i + x^{i+1}$.*

Problem 3.5.4 *Нека $\mathcal{C}$ е цикличен код с блокова дължина n. Докажете, че изображението $\varphi :\ \mathcal{F}_n \longrightarrow \mathcal{F}_n$ дефинирано с $\varphi :\ x^i \to x^{is}$, където $(s, n) = 1$, изпраща $\mathcal{C}$ в еквивалентен на него цикличен код.*

Problem 3.5.5 *Покажете, че броят на елементите $|C_i|$ на циклотомичния клас C_i относно q по модул n съвпада с броя на различните циклични премествания на представянето на i в $q-$ична бройна система.*

Problem 3.5.6 *Ако $\mathcal{C}$ е БЧХ код в тесен смисъл ($b = 1$) с дължина $n = \delta a$ и конструктивно разстояние δ, то минималното му разстояние $d(\mathcal{C}) = \delta$.*

Problem 3.5.7 *Двоичните БЧХ кодове в тесен смисъл и конструктивни разстояния $\delta = 2t$ и $\delta = 2t + 1$е съвпадат и имат размерност $k \geq n - mt$.*

Problem 3.5.8 *Нека α е примитивен елемент на $GF(2^4)$ с неразложим полином $m_1(x) = x^4 + x + 1$. Докажете, че*

$$x^{15} + 1 = (x + 1)m_1(x)m_3(x)m_5(x)m_7(x),$$

където

$$\begin{aligned} m_3(x) = \operatorname{irr}\alpha^3 = 1 + x + x^2 + x^3 + x^4, &\quad C_3 = \{3, 6, 12, 9\}; \\ m_5(x) = \operatorname{irr}\alpha^5 = 1 + x + x^2, &\quad C_5 = \{5, 10\}; \\ m_7(x) = \operatorname{irr}\alpha^7 = \operatorname{irr}\alpha^{-1} = 1 + x^3 + x^4, &\quad C_7 = \{7, 14, 13, 11\}. \end{aligned}$$

Определете параметрите на кода $\mathcal{C} = (m_1(x)m_7(x))$.

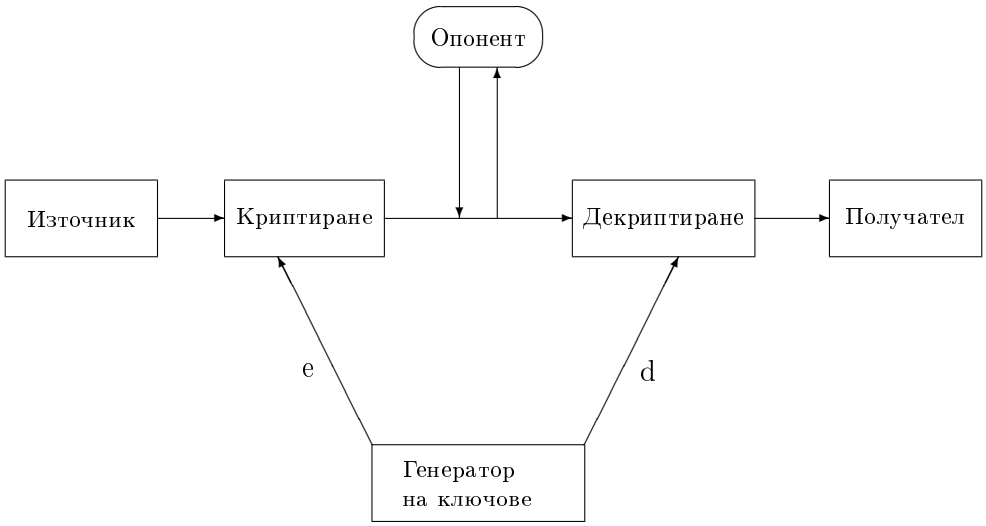
Problem 3.5.9 *Нека кодът $\mathcal{C} = (m_1(x)m_5(x)m_7(x))$ (виж предната задача) се използва за защита на предаваните данни. Намерете изпратената кодова дума, ако е получен векторът $v(x) = x^2 + x^7 + x^{12}$.*

Глава 4

Криптография

4.1 Защита на данни - същност и цели.

Блок-диаграмата на фигура 4.1 представя абстрактен модел на комуникационна система с наличие на опонент.



Фигура 4.1: Блок-диаграма на комуникационна система с опонент.

Опонентът (противникът) неправомерно включил се към обмена на ин-

формация между легитимните участници може да бъде:

• *пасивен*, когато се опитва да определи ключовете (e, d) или поне да разкрие съобщението m с подслушване на канала. Пасивното следене на канала може да бъде използвано и за *анализ на трафика*, т.е за наблюдение и анализ на източника, получателя, времето за комуникация и количеството информация, която се обменя.

• *активен (tampering)*, когато използва твърде разнообразни форми на противодействие като:

- блокиране на информационния поток,

- записване на прихванатата информация и излъчването ѝ по-късно по време на фалшива комуникационна сесия,

- промяна съдържанието на информацията чрез заличаване, вмъкване и/или разместване на части от нея и други такива.

Това ясно показва, че защитата на информацията поражда много и най-разнообразни проблеми. Тя се осъществява на три нива:

• *физическо (хардуерно)* - преди всичко информацията трябва да бъде физически налична и достъпна. По принцип този кръг проблеми се решават с изграждане на отказоустойчиви устройства за съхранение и обработка на информация, комуникационни възли и софтуерни продукти за изграждане и управление на отказоустойчиви компютърни системи.

• *теоретично*: защита при съхранение и предаване на информацията от подслушване, разрушаване или подправяне, както и от грешки, породени от естествено зашумяване на комуникационните канали. Криптографията и теорията на кодирането са теоретичните основи за решаване на такъв кръг задачи. Към това ниво ще причислим и защитата на използваните операционни системи, макар че частично този проблем се отнася и към предното ниво.

• *организационни и законодателни мерки*, които подпомагат и регламентират горните нива и аспекти на защитата на данни.

Съвременната криптография се привлича все по-широко за успешно решаване на редица проблеми, нововъзникнали или добре познати, но поставени в нова светлина от Информационното общество. Задачите, които трябва да се решават на второто ниво (с помощта на съвременната криптография) включват:

• ***Неприкосновеност и поверителност на информацията (privacy and confidentiality)***: Постигането на тази цел осигурява информацията да остане секретна и недостъпна за всички, освен за тези, които са оторизирани да си служат с нея.

• Осигуряване ***цялост на данните (data integrity)***, с което се цели изпратената информацията да не бъде подправяна или подменяна в процеса на предаване или обработка.

• Осъществяване на ***електронен подпис (signature)***: Целта е реализацията на електронен аналог на класическия подпис върху писмени документи. В електронния вариант получателят на даден електронен документ също трябва да е в състояние да убеди трета независима страна ("съдия"), че документът, и то точно в същия вид, действително е изпратен от страната, която го е подписала.

• ***Неотменимост (неотричаемост) (nonrepudiation)***: Осигурява, че никоя от страните в комуникационния процес не може да отрече реално осъществени свои действия (например, че е изпратила дадено съобщение) и/или поети ангажименти. Това е особено важна цел както от правна така и от комерсиална гледна точка.

• ***Идентификация (identification or entity authentication)***: Задачата е легално включените в обмена на информация страни (лица, компютърни терминали, кредитни карти и др.) да се убедят взаимно, че наистина комуникират една с друга, т.е. че отсрещната страна е наистина тази, за която се представя.

• ***Автентичност на съобщението (message authentication)***: Получателят на съобщението трябва да се убеди дали подателят му е наистина този, който е заявен в съобщението като такъв. Тази задача е известна още като data origin authentication.

• ***Авторизация (authorization)***: задача, която трябва да осигури прехвърляне правото на друга страна да бъде или да извърши нещо.

• ***Контрол на достъпа (access control)***: Да ограничи в определени рамки правата и възможностите, които легитимен ползвател на една компютърна система или мрежа има.

• ***Управление на ключовете (key management)***: фактор от жизнено значение за сигурността. Управлението на ключовете обхваща всички аспекти на боравенето с тях, като се започне от генерирането им до евентуалното им разрушаване. Най-голямата сложност е при съхраняването и разпространението на ключовете. Тези проблеми се решават отново с криптографски алгоритми.

• ***Анонимност (anonymity)***: Да се прикрие идентичността на определен участник в даден процес.

• ***Едновременен обмен (simultaneous exchange)***: В ситуация на много участници някакво желано свойство (качество) е в сила, докато нещо друго ценно (например подписа на отсрешната страна) не се получи.

• ***Разпределение на секрета (sharing schemes)***: В ситуация на многостранно сътрудничество дадено свойство (съглашение) действа докато броят на противниците му в групата не надвиши определен праг. Такъв тип задачи възникват при съхранение и управление на ключовете.

Криптографията, която осигурява методите и средствата за защита, заедно с ***Криптоанализа***, които има за задача разбиването на шифрите, формират науката ***Криптология***. Навлизането на радиото като средство за комуникация в армията и дипломатическите структури даде съществен тласък в развитието на криптологията и превръщането ѝ в наука, тъй като улесни достъпа до предаваната информация и направи неизползваеми много от традиционните мерки за защита. Но за начало на теория на кодирането и съвременната криптология се считат основополагащите в теория на информацията статии на Клод Шенон от 1948 година. Независимо, че решават различни задачи от защитата на информация, теорията на кодирането и криптографията използват един и същ математически апарат и се допълват. След 1979 година, когато МакЕлис предложи система за публичен ключ основана на кодове изправящи грешки, са публикувани много статии експлоатиращи идеи и конструкции от теория на кодирането за целите на криптографията.

Всеки механизъм, процес или структура опиращи се на достиженията на съвременната криптография привличат и използват нейните базисни средства (методи, алгоритми и др.), наричани най-общо ***криптографски примитиви***. В следващите параграфи ще дадем основаващи се на теорията на вероятностите и теорията на сложността на алгоритми математически критерии за оценка на тяхната ефективност и полезност, а сега ще изложим някои критерии, по които те трябва да се оценяват от практическа гледна точка.

Още в 1883 г. Керкхоф формулира съвкупност от изисквания за една криптосистема, наричани **Kerckhoffs' desiderata**:

1. да бъде, ако не теоретически, то поне практически неразбиваема;

2. компрометирането на един неин детайл да не създава проблеми на кореспондиращите;

3. ключът трябва да се запомня лесно и без да се записва, както и лесно да се сменя;

4. криптограмите трябва да могат да се изпращат с телеграф;

5. апаратурата за криптиране трябва да бъде портативна и да позволява само един човек да работи с нея;

6. системата трябва да бъде лесна, да не изисква знанието на голямо количество правила, нито умствено пренапрежение.

Тези препоръки са актуални и днес, а ето и някои съвременни практически критерии:

• ***ниво на секретност***.

Определя се от работния фактор (т.е. необходимите време и изчислителни ресурси) за компрометиране на криптографския механизъм при най-добрата известна в момента на оценката атака, както и предполагаемите в близко бъдеще развитие на компютърните технологии и методи за криптоанализ.

• ***функционалност***.

Как се съгласува с други примитиви при изграждане на механизми, каква и колко съществена е ролята му. За решаване на какви криптографски задачи даденият примитив е по-подходящ и ефективен.

• ***методи за оперирane***.

Оценява се поведението по отношение на сигурността при различни начини на прилагане, свързване с други примитиви, вида на входните данни и др.

• ***производителност и бързина***.

Например скоростта на криптиране измерена в бит/сек. Към критериите от този тип спадат, например, необходимото време да се създаде електронен подпис (ЕП), времето за проверката му, времето, необходимо за генериране на основните параметри на системата и ключовете. Оценяването винаги се прави за фиксирана форма на оперирane.

• ***внедримост***.

Оценява каква е сложността при хардуерна и/или софтуерна реализация на съответния криптографски механизъм. Небходимите ресурси (дисково пространство, памет и др) за съхраняване на ЕП както и всички данни и/или резултати получени по време на създаване на електрония подпис, издаване на сертификат и др.

Интересна от практическа гледна точка представлява така наречената "Ad hoc security". При нея се прави оценка на ресурсите на потенциалния противник и те се сравняват с ресурсите необходими за разбиване на криптосистемата.

4.2 Основни понятия. Видове криптосистеми.

Дефиниция 4.2.1 *Под* ***криптосистема*** *се разбира съвкупността от три множества* $\mathbf{M}$, $\mathbf{C}$ *и* $\mathbf{K}$ *от редици, съответно над азбуки* $\boldsymbol{A_1}$, $\boldsymbol{A_2}$ *и* $\boldsymbol{A_3}$, *и двойка* $(\boldsymbol{E}, \boldsymbol{D})$ *множества от изображения* $\boldsymbol{E} = \{\mathcal{E}_e : \mathbf{M} \to \mathbf{C} \mid e \in K\}$

и $\boldsymbol{D = \{\mathcal{D}_d : \mathbf{C} \to \mathbf{M} \mid d \in K\}}$, *които притежават свойството, че за всяко* $\boldsymbol{e \in K}$ *(криптиращ ключ) съществува единствен* $\boldsymbol{d \in K}$ *(декриптиращ ключ) така, че*

$$\boldsymbol{\mathcal{D}_d(\mathcal{E}_e(m)) = m,} \quad \textit{за всяко} \quad \boldsymbol{m \in \mathbf{M}.}$$

Множествата $\mathbf{M}$, $\mathbf{C}$ *и* $\mathbf{K}$ *се наричат съответно* ***пространство от съобщенията (plaintext) message space), пространство от шифротекста (ciphertext space)*** *и* ***пространство от ключовете (key space)****, а елементите на* $\mathbf{E}$ *и* $\mathbf{D}$ ***криптиращи*** *и* ***декриптиращи*** *трансформации.*

Забележка 4.2.2 *Понятието криптографски алгоритъм в литературата се употребява двусмислено: в широк смисъл като синоним на криптосистема, а в тесен за означаване на криптиращата и декриптираща трансформации.*

Криптосистемите се делят на две групи:

• ***Симетрични***: $e = d$ или "изчислително лесно" може да се извлече d от e.

Примери на такива криптосистеми са AES, DES, RC4, Stream ciphers и др.

• ***Асиметрични*** (наричани още *двуключови, с публичен ключ)*:
когато е "изчислително невъзможно" да се определи d, знаейки само e. Ключът e използван за шифриране може да бъде направен публично известен, докато този за дешифриране d трябва да се държи в тайна.

Примери на такива криптосистеми са RSA, Elliptic curve cryptography (ECC), тези на Rabin, ElGamal и McEliece.

Понятието "изчислително невъзможно" трябва да се разглежда в контекста на Теория на сложността на алгоритми (theory of complexity). "Невъзможно/лесно" означава, че не съществува/съществува алгоритъм, който дава решение на проблема за време, зависещо полиномиално от параметъра (параметрите) на проблема. Когато такъв алгоритъм не съществува, се казва още, че задачата е в класа $\mathbf{NP}$. Прецизирането на тези понятия излиза извън рамките на тези лекции и ние ще се опрем на интуитивния им смисъл. На читатели, които искат да се запознаят по-подробно с горните понятия и съпътстващите ги проблеми препоръчваме да прочетат някоя книга по теория на сложност на алгоритмите (например [5]).

Дефиниция 4.2.3 *Нека* X *и* Y *са произволни множества.* ***Еднопосочна функция (one-way function)*** *се нарича функция* $f : X \to Y$, *такава че е "лесно" да се пресметне* $f(x)$ *за всяко* $x \in X$, *докато за случайно* $y \in Im(f)$ *е "изчислително невъзможно" да се намери* x, *такова че* $f(x) = y$.

Пример 4.2.1 Нека g е примитивен корен по модул простото число p. Функцията $f: Z_p^* \to Z_p^*$ дефинирана с

$$f(x) = g^x \pmod{p}$$

е пример за еднопосочна функция. Ето и един числов пример: $p = 17$ и $f(x) = 3^x \pmod{17}$).

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$f(x)$	3	9	10	13	5	15	11	16	14	8	7	4	12	2	6	1

Изчисляването на $y = f(x)$ не представлява трудност, но намирането на x за произволно $y \in \mathbb{Z}_p^*$, дори за не големи p, вече създава проблеми. Както отбелязахме по-горе, в настоящото изложение няма да разглеждаме сложността на тази операция, но препоръчваме на читателя да направи горната таблица за простото число $p = 53993$ (твърде "нищожно" от криптографска гледна точка), за да добие представа за сложността на проблема.

При системите с публичен ключ се използва най-често следният подклас от еднопосочни функции.

Дефиниция 4.2.4 ***Еднопосочна функция със секрет (Trapdoor one-way)*** *се нарича еднопосочна функция $f: X \to Y$ със свойството, че при зададена допълнителна (тайна) информация (trapdoor information) за всяко $y \in Im(f)$ става "възможно" да се намери x, така че $f(x) = y$.*

Към основните градивни елементи, без които не може да се построи една система за защита на данни, трябва да добавим и следните криптографски примитиви:

- управление, създаване, съхранение и разпределение на ключове
- хеш-функции
- генератори на случайни числа

Някои от тях, както и двата основни вида (симетрични и асиметрични) алгоритми за криптиране ще разгледаме по долу.

4.3 Видове криптоатаки. Критерии за сигурност.

Дефиниция 4.3.1 *Една криптосистема се нарича разбита или компрометирана (breakable/compromised), ако трета страна е в състояние систематично без да знае предварително двойката ключове (e, d), да разкрива от шифротекста съобщението за приемливо време.*

Съществуват два основни подхода за оценка сигурността на един криптографски механизъм: единият се основава на теорията на информацията, а другият на теорията на изчислителната сложност.

Първият, предложен още от Клод Шенон, предполага, че опонентът разполага с неограничени време и изчислителни ресурси.

Дефиниция 4.3.2 *(Шенон) Една криптографска система се нарича* ***съвършено сигурна****, ако*

$$\Pr(m|c) = \Pr(m),$$

където $\Pr(m|c)$ *е вероятността да е изпратено съобщението* m *при условие, че е получено* $c = \mathcal{E}_k(m)$.

Съгласно тази дефиниция, една криптосистема е съвършенно сигурна, ако независимо, че криптоаналитикът познава алгоритъма и може да наблюдава неограничено количество шифротекст, вероятността да открие съобщението остава равна на вероятността на избор на това съобщение от пространството на всички възможни съобшения. Шенон доказва, че за да се постигне съвършена сигурност пространството от ключове трябва да е равномощно с пространството от съобщения M и ключът k да бъде избиран случайно с вероятност, притежаваща равномерно разпределение. Пример за такава реална система е използваната по "горещата линия" свързваща Москва и Вашингтон през Втората световна война.

Ако един и същи ключ се използва за криптиране на повече от едно съобщение, системата не може да бъде съвършенно сигурна и тогава се говори за безусловно сигурна.

Дефиниция 4.3.3 *Една криптосистема се нарича* ***безусловно сигурна****, ако ентропията на ключа*

$$H(k|c) = \sum_c \Pr(c) \sum_{k \in K} -\Pr(k|c) \log_2 \Pr(k|c)$$

не клони към нула независимо, че дължината на съобщението с нараства неограничено.

Предположението, че опонентът разполага с неограничени време и изчислителни ресурси, е нещо твърде далеч от реалността. На практика ефект от разбиването на един криптографски механизъм има само, ако това се случи в рамките на определен период от време. Освен това, колкото и мощни да са изчислителните възможности на опонента, те не са неограничени и разходът на компютърно време означава разход на финансови средства.

Неформално казано, по-добре е една криптосистема да се счита за сигурна, ако е невъзможно на практика опонент, който знае криптиращия и декриптиращия алгоритъм, но не знае ключа (ключовете), да извлече от шифротекста съобщението m или някаква част от него.

Дефиниция 4.3.4 *Една криптосистема се нарича* ***изчислително сигурна****, ако е доказано, че задачата за разбиването (компрометирането) ѝ е еквивалентна с решаване на* **NP***-пълна задача.*

При криптоанализ на устойчивостта на дадена криптосистема или криптографски механизъм се предполага, че опонентът разполага с пълна информация за използваните алгоритми (например криптиращата и декриптиращата функции $\mathcal{E}$ и $\mathcal{D}$, хеш-функциите и др.) и тяхната взаимовръзка. Предполага се, че същият разполага или може да се сдобие с най-разнообразна статистическа информация за шифротекста и явния текст.

Възможните атаки се класифицират в следните четири групи:

• ***Атака чрез криптограма (Ciphertext-only attack***): На опонента са известни само отделни блокове (евентуално значително количество) шифротекст: $c_1, c_2 \ldots, c_k$ и по тях трябва да разкрие ключовете (e, d) или поне съответстващите им съобщения $m_1, m_2, \ldots, m_k$;

• ***Атака чрез известен открит текст (Known plaintext attack***): Опонентът разполага с открития текст $m_1, \ldots, m_k$ (с достатъчна дължина), както и със съответните им парчета шифрован текст $c_1 = \mathcal{E}_e(m_1), \ldots, c_k = \mathcal{E}_e(m_k)$. Задачата е същата - да открие ключовете;

• ***Атака чрез избран открит текст (Chosen-plaintext attack***): Опонентът е в състояние да осигури произволен брой избрани от него съобщения да бъда криптирани, т.е. той разполага с произволен брой подбрани от него двойки (m_i, c_i), $c_i = \mathcal{E}_e(m_i)$. Задачата му е да открие ключа (ключовете);

• ***Атака чрез подбрана криптограма (Chosen-ciphertext attack***): Аналогична е на горната атака с тази разлика, че опонентът може да подбира парчета шифротекст (а не открития текст) и да осигурявя разкриване на съответните му парчета открит текст.

Последните две атаки са известни като *атака с избран текст (chosen-text attack).* ***Понастоящем криптоаналитиците считат, че само криптосистеми, които са устойчиви на този тип атака трябва да се използат в практиката***

Атаки срещу криптографски примитиви и механизми:

• прехващане на ключове

• impersonation: представяне за друг потребител

• речник: атаката е приложима, когато съобщенията са от определен тип (дължина, включват определени знаци и др.) и представлява подлагане на всички възможни съобщения на криптографското преобразование. Използва се главно за разкриване на пароли, но е приложима за хеш-функции, в някои случаи на използване на асиметрични алгоритми и т.н.

• forward search attack: предварително разглеждане на всички възможности и пресмятане на всички очаквани стойности. Прилага се например срещу хеш-функции

• повтаряне на сесия (записана предварително и повторена в друго време)

4.4 Класически симетрични криптосистеми.

Предимства на симетричните криптосистеми:

- Голяма скорост на криптиране и декриптиране както при хардуерна, така и при софтуерна реализация.

- Лесни за реализация.

- Използват се като съставни части на най-разнообразни криптографски примитиви.

Недостатъци:

- Секретният ключ е само един и всяка от страните, участващи в процеса, може да го компрометира случайно или целенасочено.

- При комуникация по двойки в мрежа от n участника са необходими $n(n-1)/2$ ключа.

- Изисква често смяна на ключа (при всяка сесия), което поражда проблеми с разпространението на ключовете.

- Не са удобни за използване в механизми за електронен подпис, защото изискват много големи ключове за проверяващата трансформация и въвличане на Трета доверена страна (TTP).

Най-употребяваните симетрични криптосистеми са **блоковите криптосистеми (block ciphers)**.

Таблица 4.1:

символ	число	символ	число	символ	число	символ	число
0	00	м	23	Е	46	Ю	69
1	01	н	24	Ж	47	Я	70
2	02	о	25	З	48	,	71
3	03	п	26	И	49	.	72
4	04	р	27	Й	50	-	73
5	05	с	28	К	51	:	74
6	06	т	29	Л	52	;	75
7	07	у	30	М	53	"	76
8	08	ф	31	Н	54	(	77
9	09	х	32	О	55	)	78
интервал	10	ц	33	П	56	I	79
а	11	ч	34	Р	57	V	80
б	12	ш	35	С	58	*	81
в	13	щ	36	Т	59	=	82
г	14	ъ	37	У	60	+	83
д	15	ь	38	Ф	61	/	84
е	16	ю	39	Х	62	^	85
ж	17	я	40	Ц	63	!	86
з	18	А	41	Ч	64	?	87
и	19	Б	42	Ш	65	$	88
й	20	В	43	Щ	66		
к	21	Г	44	Ъ	67		
л	22	Д	45	Ь	68		

Дефиниция 4.4.1 *Криптосистема, чието пространство от съобщенията* $\mathbf{M}$ *се състои от редици* $m = (m_1, \ldots, m_n)$ *с дължина* n*, т.е. разделя открития текст на блокове от по* n *символа на азбуката, се нарича* ***блоков шифър с дължина*** n*.*

Повечето добре известни симетрични криптосистеми са блокови. Такива са и следните класически криптосистеми:

Заместващи шифри (Substitution ciphers)

Дефиниция 4.4.2 *Критосистема, която попада в една от следващите категории, се нарича* ***заместващ шифър (субституция)****:*

Проста субституция *Нека е дадена блокова криптосистема с дължина* n *и пространство от ключовете* $\mathbf{K}$*, състоящо се от пермутации*

на азбуката. Криптирането с ключа $e \in \mathbf{K}$ се извършва по правилото

$$\mathcal{E}_e(m) = (e(m_1), e(m_2), \ldots, e(m_n)).$$

Хомофоник (homophonic) *Всеки символ a от азбуката се изобразява $a \longrightarrow H(a)$ в множество от редици с дължина t, като $H(a) \cap H(b) = \emptyset$, за $a \neq b$.*

Многоазбукови (polyalphabetic) *За разлика от простата субституция, ключовете $e = (\pi_1, \ldots, \pi_n)$ представляват наредени $n-$орки от пермутации на азбуката и*

$$\mathcal{E}_e(m) = (\pi_1(m_1), \pi_2(m_2), \ldots, \pi_n(m_n)).$$

Разместващи (пермутационни) шифри (Transposition ciphers)

Дефиниция 4.4.3 *Нека $\mathbf{M}$ е пространството от съобщения на блоков шифър с дължина n. Криптосистемата се нарича **разместваща (пермутационна)**, ако всеки ключ e еднозначно определя пермутация $\pi \in S_n$ на $\{1, 2, \ldots, n\}$, такава че*

$$\mathcal{E}_e(m) = (m_{\pi(1)}, \ldots, m_{\pi(n)}).$$

Ще илюстрираме заместващите шифри с една криптосистема основана на афинната трансформация

$$\begin{pmatrix} x \\ y \end{pmatrix} \Longrightarrow \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}.$$

В нашия случай ще изберем

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 2 & 13 \\ 17 & 22 \end{pmatrix} \quad \text{и} \quad \mathbf{b} = \begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} -7 \\ 5 \end{pmatrix},$$

които ще представляват ключа. На елементите на матриците ще гледаме като на елементи на $\mathbb{Z}_{89}$, т.е. действията ще ги извършваме по модул 89. Съответствието, при което на буквите и другите текстови символи се съпоставят числа по модул 89 се задава с таблица 4.1.

Да криптираме текста: "Теория на числата". На него му съответства редицата от двуцифрени числа 59 16 25 27 19 40 10 24 11 10 34 19 28 22 11 29 11 10. Накрая допълваме текста с интервал (10) за да станат числата четен брой. Първите две числа се преобразуват в

$$\begin{pmatrix} 2 & 13 \\ 17 & 22 \end{pmatrix} \begin{pmatrix} 59 \\ 16 \end{pmatrix} + \begin{pmatrix} -7 \\ 5 \end{pmatrix} = \begin{pmatrix} 52 \\ 25 \end{pmatrix}$$

Извършвайки последователно пресмятанията получаваме 52 25 38 45 17 51 58 80 56 56 41 22 68 75 36 29 56 56 Следователно шифротекстът е

ЛоьДжКСVППАлЬ;щтПП

Декриптирането се извършва като се приложи обратната трансформация:

$$\begin{pmatrix} x \\ y \end{pmatrix} \Longrightarrow \mathbf{A^{-1}} \begin{pmatrix} x \\ y \end{pmatrix} - \mathbf{A^{-1}b} = \begin{pmatrix} 22 & -13 \\ -17 & 2 \end{pmatrix} \left[\begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} -7 \\ 5 \end{pmatrix} \right]$$

Разгледаният тип криптосистеми са предложени в 1931 от Хил (Hill). Поради линейността си те нямат криптографска стойност днес, макар че все още се използват (алгоритми използващи същия принцип) в някои комуникационни системи за да увеличат ентропията на предаваната информация.

Друг много използван тип симетрични криптосистеми са ***поточните шифри (stream ciphers)***. При тях криптирането представлява прибавяне по модул 2 на бинарна редица към бинарния запис на открития текст.

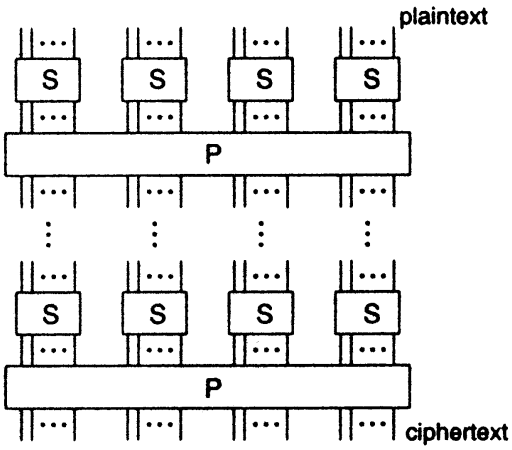
4.5 Съвременни блокови шифри. DES и AES.

Използваните днес блокови шифри принадлежат на класа от така наречените ***итеративни шифри от тип на Фейстел (Feistel)***, които ще разгледаме в този параграф.

Дефиниция 4.5.1 *Под произведение на две или повече криптосистеми с множества от криптиращи трансформации $\mathcal{E}_i$ се разбира криптосистема с криптиращи трансформации, явяващи се композиции $\mathcal{E}_1\dot{\mathcal{E}}_2\ldots$ от криптиращите функции на изходните шифри.*

Когато горните криптосистеми са субституции или пермутационни, произведението им се нарича *субституционно-пермутационна (SP) мрежа*. Прилагането на поредната трансформация $\mathcal{E}_i$ се нарича *round (рунд, цикъл, итерация)*.

Дефиниция 4.5.2 *Итеративен блоков шифър се нарича блоков шифър с някаква вътрешна трансформация - итерационна функция (round function), която се изпълнява многократно (r пъти). От ключа с дължина k се конструират r на брой итерационни ключа e_i за всяко прилагане на итерационната функция.*



Фигура 4.2: Субституционно-пермутационна (SP) мрежа

Дефиниция 4.5.3 *Фейстел шифър се нарича итеративен шифър, изобразяващ открит текст* (L_0, R_0) *от два* $t-$*битови подблока в шифротекст* (L_r, R_r), $r \geq 1$ *чрез* r *итерации. За всяко* $1 \leq i \leq r$, $r-$*тата итерация изобразява* (L_{i-1}, R_{i-1}) *в* (L_i, R_i) *посредством итерационния ключ* k_i *по правилото:* $L_i = R_{i-1}$, $R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$, *където* f *е итеративната функция.*

Най-често $r \geq 3$ и е четно.

Блоковите шифри се прилагат в следните модификации (форми на оперирane):

Electronic Code Book (ECB) mode

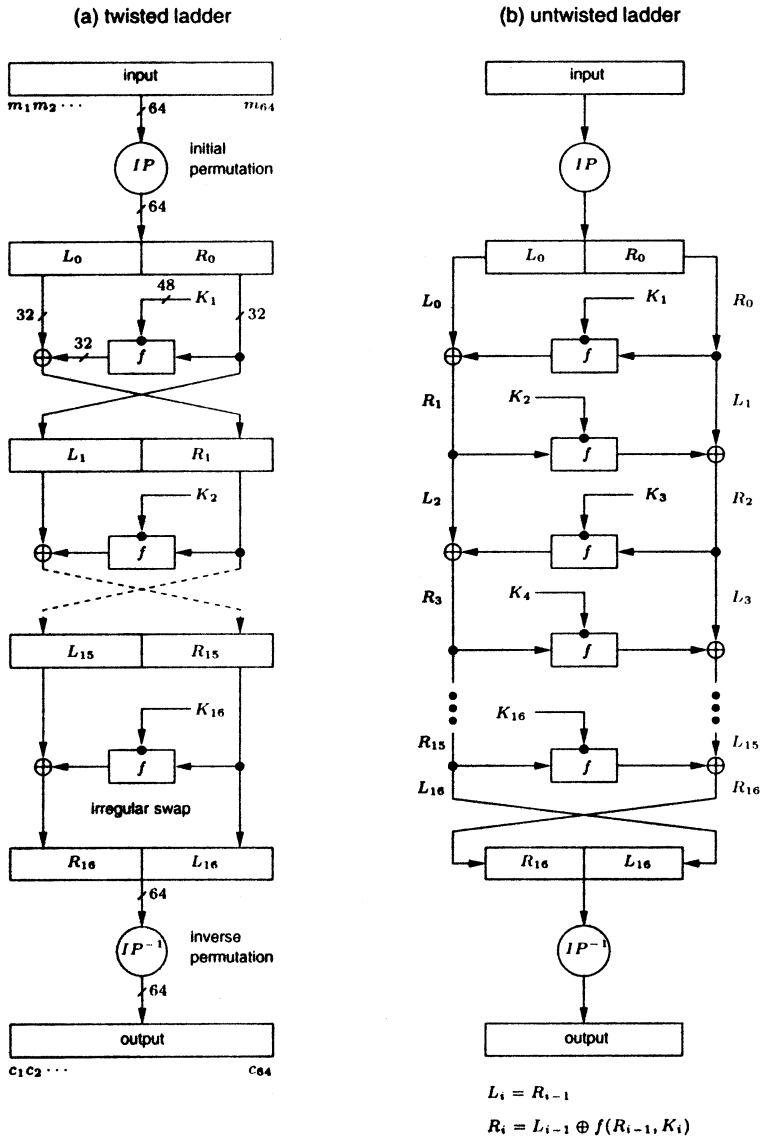
Всеки блок се криптира независимо. Тази форма на оперирane е поддатлива на атаката "речник" и не се препоръчва.

Cipher Block Chaining (CBC).

Преди да се криптира, всеки блок от битове се събира по модул 2 (XOR) с резултата от криптирането на предишния ход. За първия блок се ползва начален (зададен предварително) блок c_0. Широко се използва, в това число за осигуряване цялостта на данните (MDC - виж дефиниция 4.7.2) и за автентификация на съобщението (т.е. хеширане с ключ - MAC модификация, виж дефиниция 4.7.2)).

Cipher Feedback (CFB) and Output Feedback (OFB)

Прилага се, когато се използва n-блоков шифър, но откритият текст трябва да се обработва на блокове с дължина r. В изместващ регистър I с дъл-



Фигура 4.3: Блокова схема на DES

жина n се задава начална стойност. Съдържанието на I се криптира чрез ключа и левите му r бита се събират с постъпващия блок от открития текст. Полученият шифротекст c_j се вкарва в регистъра I чрез изместване вляво за да се използва при следващата итерация.

При OFB модификацията обработката е аналогична с тая разлика, че регистърът I се зарежда с резултата от криптирането (блок с дължина n), т.е. откритият текст не влияе върху стойността на регистъра I.

Ето и някои от най-използваните блокови шифри:

DES - Data Encryption Standard:
Създаден (основно от IBM) през 1975, той е одобрен като федерален стандарт за САЩ (Federal Information Processing Standard) през 1977. Не е патентно защитен и се превръща в световен стандарт. Известен е още под името Data Encryption Algorithm (DEA). Трансформира блок от 64 бита открит текст в блок от 64 бита, като използва ключ с дължина 56 бита. Описан е в FIPS 46 [6]. Използва се за осигуряване на конфиденциалност, цялост на данните и аутентификация на съобщението. Реализира се във всичките форми на оперираne: ECB, CBC, CFB, OFB (виж FIPS 81 [6], ISO 8372, ISO 10116).

DES е Фейстел шифър, за който $f(R_{i-1}, k_i) = P[S(E(R_{i-1}) \oplus k_i)]$, където P е пермутация, E е функция, която превръща 32 бита в 48 бита и S е функцията осъществявана от 8 така наречени S-box. Те осигуряват нелинейността. Всеки от тях трансформира 6 бита в 4 бита и така като изход се получава отново редица от 32 бита. Фигура 4.3 дава обща представа за извършваните итерации.

В последните години с развитието на компютърните технологии и повишаване бързодействието на компютрите DES стана ненадежден. С мощни компютри при пълно изчерпване е достигнато до ключа за часове. Затова се препоръчва при приложения, които изискват запазване на секретността повече от няколко часа, да се използва 3-DES или други криптосистеми с по-дълъг ключ.

AES - Advanced Encryption Standard:
Във връзка с възможността DES да стане разбиваем поради развитието на компютърните технологии National Institute of Standards and Technology-САЩ (NIST) обяви процедура за намиране на наследник на DES - блоков шифър, който да може да работи с ключове с дължини 128, 192 и 256. До крайния срок юни 1998 г. са постъпили 15 предложения, които удовлетворяват изискванията на NIST. След публично объждане NIST избра 5 финалиста: Twofish (Bruce Schneier et al.), RC-6 (RSA Lab.), Mars (IBM), Serpent (UK, Israel, Norway), Rijndael. Процедурата по избор на нов стандарт на NIST

приключи през есента на 2000 г. с избора на Rijndael за заместник на DES и понастоящем е официален стандарт на NIST. Автори на алгоритъма са двамата белгийски криптографи Joane Daemen и Vincent Rijmen. Описанието му може да се намери на сайта на NIST (http//:csrc.nist.gov/CryptoToolkit/aes). AES е, разбира се, итеративен, субституционно-пермутационен алгоритъм.

3-DES - Triple Data Encryption Standard:
Представлява трикратно последователно криптиране в един от следните два варианта:
- $\mathcal{E}(m) = \mathcal{E}_3(\mathcal{E}_2(\mathcal{E}_1(m)))$, където $\mathcal{E}_i,\ i = 1,2,3$, е DES алгоритъм с ключ e_i.
- $\mathcal{E}(m) = \mathcal{E}_1(\mathcal{D}_2(\mathcal{E}_1(m)))$, където $\mathcal{E}_i,\ i = 1,2$ е DES алгоритъм с ключ e_i и $\mathcal{D}_2 = \mathcal{E}_2^{-1}$, т.е. представлява криптиране - декриптиране (с друг ключ) - криптиране. Този вариант е по-популярен, защото при $e_1 = e_2$ се получава простият DES.

В първия случай се използва ключ с дължина 168 бита (3 ключа по 56), а във втория случай - ключ с дължина 112 (възможна е и употреба със 168 битов ключ).

IDEA - The International Data Encryption Algorithm от James Massey и Xuejia Lai, 1990.:
Трансформира блок от 64 бита в 64 бита чрез 128 битов ключ. Изпълняват се 8 еднотипни итерации и една изходна. Всяка итерация използва 6 подключа от по 16 бита. IDEA е три пъти по бързэ от 3-DES и малко по-сигурен. Реализиран е в PGP протокола за защита на електронна поща. Патентован е в Европа, САЩ и Япония. Работи във всички форми на оперирane и освен за засекретяване може да се ползва за цялост на данните и аутентификация.

Други блокови шифри са RC2 (Ronald Rivest, 1996; описан е в RFC 2268 [9]), RC5 (1994, използва се в IPsec.), FEAL, SAFER и др.

Забележка 4.5.4 *Симетричните алгоритми предполагат доверие между комуникиращите страни (споделят общ ключ) и затова, както отбелязахме, не са подходящи за създаване на електронен подпис. Те могат да се ползват за аутентификация на съобщението, т.е. когато адресантът се нуждае само от увереност, че полученото съобщение (криптирано или открито) наистина е пратено, и то в този вид, от автора. Това се постига с добавка на няколко байта (обикновено 4) към съобщението (MAC). Ако електронното изявление не е предмет на възможни последващи оспорвания е възможно да не се подлага на асиметричен алгоритъм. Описаното действие може да се причисли към обикновения електронен подпис съгласно терминологията на ЗЕДЕП [3]. Но в този случай не може да се докаже автентичността на изявлението пред трета страна (съдия).*

4.6 Асиметрични криптосистеми.

Предимства:

- Двойката ключове (частен и публичен) могат да бъдат ползвани многократно и през дълъг период (дори няколко години).

- В контраст на симетричния случай, при мрежа от много участници са необходими толкова двойки ключове, колкото са участниците.

- Позволяват създаване на ефективни схеми за електронен подпис с проверяваща функция, изискваща ключ с доста по-малки размери от съответните, построени на симетрични алгоритми.

Недостатъци:

- Относително по-бавни в сравнение със симетричните.

- При използване за криптиране ключът им е много по-дълъг от този на симетричните.

- Дължината на добавения електронен подпис е относително голяма.

Към асиметричните криптосистеми спадат RSA, RSA(Rabin-Williams modification), ElGamal, Elliptic Curves Cryptography и др.

Криптосистема RSA:

Носи името на създателите си: Rivest, Shamir, Adleman. Описанието ѝ като стандарт се дава с PKCS #1 [8], ISO 11166. Сигурността на RSA се основава на трудността да се разложи едно естествено число n на прости множители. За достатъчно големи n съвременните математически методи и компютърна техника не могат да се справят с тази задача.

Нека $n = pq$ е произведение на две големи прости числа и e, $1 < e < \varphi(n)$, е случайно число, такова че $(e, \varphi(n)) = 1$ ($\varphi(n)$ е функцията на Ойлер). RSA ползва два ключа:

ПУБЛИЧЕН КЛЮЧ: (n, e), състоящ се от модул n и експонента e;
ЧАСТЕН КЛЮЧ: d, $1 < d < \varphi(n)$, такова че $ed \equiv 1 \pmod{\varphi(n)}$.
КРИПТИРАНЕ: Съобщението се превръща в поредица от числа $m_1, m_2, \ldots$, всяко от които се криптира по правилото:

$$m_i \longrightarrow c_i \equiv m_i^e \pmod{n}.$$

Поредицата $c_1, c_2, \ldots$ представлява шифротекста.
ДЕКРИПТИРАНЕ: Откритият текст се получава с

$$m_i \equiv c_i^d \pmod{n}.$$

Наистина $c_i^d \equiv m_i^{ed} \pmod{n}$. Но от $ed \equiv 1 \pmod{p-1}$ и теоремата на Ферма следва, че $m_i^{ed} \equiv m_i \pmod{p}$. Аналогично $m_i^{ed} \equiv m_i \pmod{q}$. Тъй като p и q са различни прости числа, то $m_i^{ed} \equiv m_i \pmod{pq}$.

Криптосистемата RSA се използва най-вече за създаване на електронен подпис, при разпространяване и съхраняване на ключове за симетрични алгоритми и за аутентификация. Понастоящем модулът n трябва да бъде число поне от 768 бита (в двоичен запис). При приложения, които изискват по-дълъг срок на надежност (месеци или години, както е при електронния подпис), ***трябва да се ползват модули с дължина поне 1024 бита***. Числата p и q също трябва да удовлетворяват някои ограничения, които са описани в стандартизационните документи.

Експонентата и частният ключ не е препоръчително да бъдат много малки, защото при определени обстоятелства може да се разкрие съобщението m. Има реализации, при които $e = 3$, но в тези случаи, особено за малки m, сигурността е слаба. В тези случаи към съобщението се добавя случайна информация, за да се избегне малко m. Ако вместо малки експоненти се вземат такива, които имат малко единици в двоичното им представяне, криптирането също ще изисква по-малко ресурси.

Вариант на RSA е криптосистемата на Rabin-Williams. Сигурността ѝ се основава на трудността на разлагане на прости множители и на намирането квадратен корен по модул съставно n.

Пример 4.6.1 Нека $p = 73$ и $q = 109$. Тогава $n = pq = 7957$, а $\varphi(n) = 7776$. Да изберем за експонента $e = 17$. Публичният ключ ще бъде $(7776, 17)$. Тъй като $17 \cdot 7489 \equiv 1 \pmod{7776}$, то частният ключ е $d = 7489$.
Да криптираме съобщението "Кой е Ойлер?". За целта всеки два последователни текстови символа ще разглеждаме като двузначни числа в 89-ична бройна система, т.е. от Ко|й |е |Ой|ле|р? получаваме $51 \cdot 89 + 25 \mid 20 \cdot 89 + 10 \mid \ldots \mid 27 \cdot 89 + 87$. Следователно на нашия текст съответства съвкупност от 6 числа: 4564 | 1790 | 1434 | 4915 | 1974 | 2490. Сега всяко от тези числа трябва да подигнем на степен 17 по модул 7957. Да забележим, че $a^{17} = (((a^2)^2)^2)^2 \cdot a$, т.е. повдигането може да извършим с последователни повдигания в квадрат и едно умножение. След изпълнение на операциите получаваме

$$4564^{17} \equiv 7859,\ 1790^{17} \equiv 3552,\ \ldots,\ 2490^{17} \equiv 2911 \pmod{7957}.$$

И така, шифротекстът е набора от шесте числа

$$7859 \mid 3552 \mid 1885 \mid 2339 \mid 1557 \mid 2911.$$

Ако всяко от тях представим в 89-ична бройна система и използваме таблица 4.1 ще получим текстови запис на шифротекста:

$$\text{р\$} \mid \text{*ю} \mid \text{ ек} \mid \text{оп} \mid \text{Гж} \mid \text{Цх}.$$

Трябва да отбележим, че $88 \cdot 89 + 88 = 7920 < 7957$, то след повдигане в степен по модул n може да се получи число, което е с 3 цифри в 89-ична бройна система. Затова, ако шифротекстът ще се преобразува в текстова форма, то някакво разделяне (както горе) трябва да се направи. Тази възможност също създава неудобства при използване на RSA. Затова обикновено се ползва за електронен подпис или криптиране на кратки съобщения (например парола или ключ за симетрична криптосистема), така че да не се налага разбиване на текста на блокове.

Криптосистема на ElGamal:
Вариант на тази ситема се използва като асиметричен алгоритъм в механизма за електронен подпис DSA. Сигурността ѝ се основава върху трудността на намиране на дискретен логаритъм в $\mathbb{Z}_p^\star$. ***Публичният ключ*** на даден потребител U представлява тройка (p, g, E), където g е примитивен корен по модул p и $E = g^a$. ***Частният ключ*** е a : $1 \leq a \leq p-2$. Ако група ползватели имат едни и същи p и g, то публичният ключ, който се записва в регистъра, е само $E = g^a$. Процедурата по криптирането е следната:

Потребител S, който иска да изпрати съобщение m на U избира произволно число k и изпраща двойката $(g^k,\, mE^k)$.

За да декриптира съобщението, U изчислява $D = (g^k)^a = E^k$ и след това намира $mE^kD^{-1} \equiv mE^kE^{-k} \equiv m \pmod{p}$.

Простото число p трябва да е с дължина поне 768 бита, като за по-дългосрочна сигурност се препоръчва дължина поне 1024 бита. Като недостатък на криптосистемата на ElGamal може да се посочи, че шифротекста е два пъти по-дълъг от откритията текст.

При варианта, фиксиран в DSA (FIPS 186), цикличната група не е $\mathbb{Z}_p^\star$, а нейна подгрупа от ред простото число q с дължина 160 бита и делящо $p-1$. Затова публичният ключ включва и q.

Криптосистемата на ElGamal естествено се обобщава като $\mathbb{Z}_p^\star$ се замени с произволна крайна циклична група от достатъчно голям ред. Частен случай на обобщената криптосистема на ElGamal е криптоситемата, основана на ***елиптични криви*** (ECC).

Елиптичните криви се дефинират като множество от точки (x, y), удовлетворяващи уравнение $F(x, y) = 0$, където $F(x, y)$ е полином от дадена степен на две променливи с коефициенти от крайно поле. В стандарта IEEE P1363 се визират само два типа криви от трета степен: едната за поле с p елемента, $GF(p)$, p просто число, а другата за поле $GF(2^m)$. Базисна операция в ECC е събирането на точки и в частност намиране на $k-$кратно на точка: kP. Минималното естествено k, за което kP е безкрайната точка, се нарича *ред* на точката P. Избира се точка G (базисна точка) с ред просто число r, такова че r^2 не дели реда на кривата. Двойката ключове се дефинира както следва:

- частният ключ е естествено число s по-малко от r;
- публичният ключ е точка $W = sG$.

Сигурността на ECC се базира на невъможността по дадени G и sG да се намери s, т.е на задачата за намиране на дискретен логаритъм в адитивния вариант, зададен чрез елиптична крива. Следващата таблица показва съотношението между дължините на използваните ключове за RSA и ECC при приблизително еднаква сигурност.

RSA	ECC
768	132
1 024	163
2 048	210
21 000	600

ECC е много подходяща за реализация в мобилни устройства и smart карти, където съществуват естествени ограничения върху изчислителните ресурси. В тези случаи използването на относително по-къси ключове и по-ефективни (по отношение разход на ресурси) алгоритми, без да се загубва сигурност, е съществено.

Дължината на ключа, която се препоръчва е над 160 бита.

Освен с IEEE P1363 ECC е описана с ANSI X9.62 и X9.63. Информация за ECC може да се намери и на сайта на NIST.

4.7 Хеш-функции.

Дефиниция 4.7.1 ***Хеш-функция (hash function)*** *се нарича еднопосочна функция h (виж §4.2), която изобразява редица (от битове) с произволна дължина в редица с фиксирана дължина n (например, $n = 64, 128, 160, 512$)*

и притежава свойството, че е свободна от колизии, т.е. изчислително невъзможно е да се намерят две различни съобщения m и m', така че $h(m) = h(m')$.

От дадената дефиниция е ясно, че по резултата от прилагането ѝ практически е невъзможно (виж определението на еднопосочност) да се определи съобщението, което е хеширано.

Основните приложения на хеш-функциите са за проверка цялостта на данните и създаване на кратка добавка, която да бъде подписана при реализация на електронен подпис, както и при някои идентификационни протоколи.

Препоръчително е да се използват хеш-функции с $n = 128$ или 160 бита, за да са свободни от колизии и същевременно n да не е твърде голямо в сравнение с подписвания електронен документ, за да не се забавя обработката му.

Дефиниция 4.7.2 *Хеш-функции, които не използват секретен ключ и се прилагат само за проверка цялостта на данните се наричат* ***modification detection codes (MDCs)****. Хеш-функция, която въвлича използване на секретен ключ, се наричат* ***message authentication code (MAC)****. Употребява се за аутентификация на източника на информацията.*

Много хеш-функции са итеративни процеси. Информацията се дели на блокове $\{x_i\}$ от по t бита

$$H_0 = initial\ value, \qquad H_i = f(H_{i-1}, x_i), \quad \text{for } \ 1 \leq i \leq r$$

и изходната трансформация е

$$h(x) = g(H_r),$$

където f и g са зададени функции.

Примери на хеш-функции са MD4, MD5, SHA-1, RIPEMD-160. Достатъчно информация за хеш-функциите може да се намери в ISO/IEC 10118.

MDC хеш-функции

MD4 и **MD5**:

MD4 и MD5 (message digest) алгоритми са създадени за софтеурна реализация върху 32-битови компютри. Описани са в RFC 1320 и RFC 1321 [9]. MD5 е подобрен вариант на MD4. За MD4, обаче, са намерени колизии след 2^{20} проверки. Затова MD4 не би трябвало да се ползва за отговорни

цели, особено, ако подписваният текст не е с ясно изразено съдържание, което трудно би се изменило незабелязано при случайна промяна.

SHA-1:

Създадена е от NIST и NSA (National Security Agency) за 32 битови компютри. Дава като резултат 160 бита хеш-стойност. Включена е като функционална част от процедурата по пораждане на ключ за механизма за електронен подпис DSA (ключ за алгоритъма на ElGamal). Понастоящем са стандартизирани и се прилагат версиите SHA-224, SHA-256, SHA-384 и SHA-512. Описанието им може да се намери във FIPS 180-2 [6].

RIPEMD-160:

Разработен е на базата на MD4, като са взети предвид опита от прилагането и резултати от анализа на предишните хеш-функции.

Следната таблица дава сравнение между изредените горе хеш-функции:

Наименование	Дължина	Итерации × бр. стъпки за итер.	Относ. скорост
MD4	128	3×16	1.00
MD5	128	4×16	0.68
RIPEMD-128	128	4×16 удвоено в паралел	0.39
SHA-1	160	4×20	0.28
RIPEMD-160	160	5×16 удвоено в паралел	0.24

DES in MDC mode:

Използването на блоков шифър (в частност и DES) като хеш-функция е описано в ISO 10118-2. Тази реализация дава като резултат 128 бита хеш-стойност и е известна като MDC-2 (вариант е MDC-4). При такова използването на DES (блоков шифър) за хеширане, например в схема за електронен подпис, ключът вече не е таен. Той трябва да се предостави на проверяващия.

MAC хеш-функции

При размяна на електронни изявления е съществено получателят (адресантът) да е уверен, че авторът на съобщението е наистина този, от чието име го получава. Дори когато изявлението не е секретно, неговата автентификация е съществена. Един начин тя да се осъществи е като се добави хеш-стойност (от m бита), получена чрез хеширане с ключ, който се предполага, че е известен само на автора и адресанта. Хеш-функцията може да бъде блоков шифър работещ в режим CBC.

DES в CBC модификация:

Описана е в ISO 9797. В края на итерациите може като опция да се изпълни операция, аналогична на тази в 3-DES.

Много често MAC дължината m е по-малка от блоковата дължина, например 4 байта, т.е. 32 бита, или дори 3 байта. Числото m не трябва да е много малко, за да не се увеличи много вероятността (2^{-m}) опонентът да отгатне MAC. Също така разликата $n-m$ не трябва да е малка, за да не се редуцира неприемливо пространтвото от 2^{n-m} двоични редици (ключовете, съответстващи на даден MAC), които опонентът би трябвало да провери. Препоръчително е, когато дължината на ключа е 56 бита, всеки ключ да се използва само веднъж или да се използва двоен ключ (112 бита - в този случай и дължината на "подписа" става 8, а не 4 байта).

Message Authenticator Algorithm (MAA):

Предложен е в 1993 г. и е ориентиран към софтуерна реализация на 32-битови компютри. Входните данни са съобщение с дължина кратна на 32 и 64-битов ключ. Като изход се получава 32-битова хеш-стойност. Описан е в ISO 8731-2.

MD5-MAC:

Представлява хеширане с ключ, използващо хеш-функцията MD5. Предполага ключ от 128 бита (ако е по-къс, се допълва чрез каскадиране). От ключа и фиксирани константи се получават входните параметри за MD5.

4.8 Криптографски протоколи.

Съществен клас примитиви, които се изграждат от последователно прилагане на описаните по-горе, са криптографските протоколи.

Дефиниция 4.8.1 ***Криптографски протокол*** *е разпределен алгоритъм, дефиниран чрез последователност от стъпки, детайлно описващи необходимите действия, които двама или повече участника (страни) трябва да осъществят, за да постигнат определени цели по отношение на сигурността.*

Протоколите играят ключова роля в криптографията и имат съществен принос за постигане на нейните цели. За илюстрация тук ще разгледаме протокола на Diffie-Hellman, който се ползва в популярния софтуерен протукт за трансфер на файлове на SSH.

Протоколът на Diffie-Hellman (DH) е прост и ефективен метод за договаряне между две страни на сесийния ключ за последващия обмен на информация. По този начин се решава елегантно един критичен за сигурността на системата проблем - разпространението на ключовете.

Нека A и B са двама потребители (устройства), които трябва да се договорят за някаква секретна информация. Например, това може да бъде ключът за симетричния алгоритъм за криптиране, който те ще ползват в последващ обмен на информация. В основната версия на DH протоколът два параметъра са фиксирани и обикновенно са публични:

- голямо просто число p (1024, дори 2048 бита),

- "пораждащ елемент" g, където $1 < g < p-1$ и $g^k \not\equiv 1 (\mod p)$ за $0 < k < p-1$.

DH протоколът се състои от следните стъпки:

1. A избира случайно (и секретно) число $a: \ 1 < a < p-1$, пресмята и изпраща на B числото $x = g^a$;

2. аналогично B поражда $b: \ 1 < b < p-1$, и изпраща на A числото $y = g^b$;

3. B изчислява ключа $K = x^b = g^{ab}$

4. A изчислява ключа $K = y^a = g^{ab}$

Пресметнатият ключ K е сесийния ключ, който се използва в блоковия шифър (например AES). Числата a и b са секретни, не се изпращат по мрежата и се генерират непосредствено преди сесията. След пресмятането на K те се унищожават.

Друг важен пример на двустранен протокол е така наречения ***протокол с нулева информация (zero-knowledge (ZK))***, предложен от Goldwasser, Micali and Rackoff [7]. Едната страна се нарича *заявител*, а другата *проверяващ*. ZK протоколът позволява на заявителя A да докаже на проверяващия B истинността на някакво твърдение, например факта, че знае някакъв секрет, без да разкрива каквато и да е информация за самото твърдение. Някои протоколи от този тип са в основата на механизми за автентификация.

4.9 Механизми за създаване на електронен подпис.

4.9.1 Преследвани цели и проблеми, които трябва да се решават.

Бързината и гъвкавостта при предаване и обработка на информацията са основни предимства на съвременните информационни технологии, но тази отворена и леснодостъпна за широк кръг субекти информационна среда изостря особенно силно проблема за достоверността на данните. Електронният подпис е измежду най-значимите постижения на модерната криптог-

рафия, които допринасят за решаването на този проблем. Сферата му на приложение включва:

• официални комуникации с държавни и обществени институции (изпращане на данъчни декларации, обмен на административни и правни документи);

• сключване на договори в отворена информационна мрежа (покупки и продажби, електронни транзакции и др.);

• електронно подписване за идентификационни и авторизационни цели (при включване в компютърна система, идентификация на сървери и др.);

• в затворени системи (например в корпоративен Интранет);

• електронно подписване за лични цели.

Една схема (механизъм) за създаване на електронен подпис (МСЕП) трябва да позволява дадено съобщение, заявление, контракт или друго такова, подписано от един участник в комуникационния процес, да бъде покъсно проверено от всеки друг. Нещо повече, механизмът трябва да позволява арбитриране в случай на противоречия (диспут) между автора и получателя. При това, процесът на проверка трябва да може да бъде повтарян многократно и в продължение на дълъг период (например няколко години), докато е в сила валидността на направеното изявление (контракт и др.). Механизмът на поставяне на електронния подпис трябва да изключва всяка възможност за отхвърляне на неговата валидност, подправяне и/или друга форма на измама.

При създаването на електронния подпис са възможни най-различни нюанси. Съдържанието на подписания документ може да бъде явно или да е достъпно само за адресата на подписаното електронно изявление. В някои ситуации подписващият документа не е необходимо или не трябва да знае съдържанието на подписания от него документ. Такъв е случаят, когато дадена организация (например за обслужване на разплащания, сертификация на ключове и др.) трябва само да регистрира във времето дадено електронно изявление и да го придвижи.

Механизмите за създаване на ЕП могат да се групират основно в две групи:

- ЕП чрез създаване на добавка (digital signature with appendix) ISO 14888, PKCS#1 [8]. Това е основния и най-използван механизъм за електронен подпис. Към електронното изявление (в явен или криптиран вид) се добавя определена порция информация, която представлява подписа. Съществено при този механизъм е, че за проверка на подписа е необходимо самото електронно изявление (или информация получена от него чрез определено преобразуване). Този тип механизъм е изобразен по-долу на

фигура 4.4.

- ЕП с разкриване на съобщението (signature scheme with message recovery) ISO/IEC 9796-1, ISO/IEC 9796-2. В този случай не е необходимо да се знае съобщението предварително за да се направи проверка на ЕП. Използва се само за подписване на много кратки електронни изявления. Не е визиран от ЗЕДЕП.

4.9.2 Структура и функционално описание.

Всеки механизъм за създаване на електронен подпис включва *алгоритъм за генериране на подписа* и *проверяващ автентичността му алгоритъм.* Алгоритъмът за генериране на подписа, заедно с методите за привеждане на съобщението в подходящ за подписване вид, формират *процеса на цифрово подписване*, а проверяващият алгоритъм заедно със средствата за разкриване на съобщението - *процеса на проверка на цифровия подпис.*

Формализираното описание на механизма, по който един участник U в компютърна комуникационна система подписва своите съобщения (електронни изявления) включва:

• пространство от съобщенията $\mathbf{M}$ и пространство от *подписите* $\mathbf{S}$

• трансформация $\mathcal{S}_U : \mathbf{M} \to \mathbf{S}$, която е известна само на U

• проверяваща трансформация $\mathcal{V}_U : \mathbf{M} \times \mathbf{S} \to \{true, false\}$,

като са в сила следните свойства:

• $s \in \mathbf{S}$ е валиден подпис тогава и само тогава, когато $\mathcal{V}_U(m, s) = true$

• изчислително невъзможно е за всеки друг освен U да намери за всяко $m \in \mathbf{M}$ такова $s \in \mathbf{S}$, че $\mathcal{V}_U(m, s) = true$.

Подписване:

1. Пресмятане на $s = \mathcal{S}_U(m)$.
2. Изпращане на (m, s).

Процедура за проверка:

1. Получаване на $\mathcal{V}_U$ (от U или от регистър).
2. Пресмяне на $u = \mathcal{V}_U(m, s)$.
3. Приемане на подписа при $u = true$ и отхвърляне в противния случай.

На практика МСЕП трябва да притежава следните свойства:

- лесно да се създава ЕП, т.е. $\mathcal{S}_U$ да бъде достатъчно бърза и удобна за прилагане;

- лесна проверка на ЕП, т.е. $\mathcal{V}_U$ да бъде достатъчно бърза и удобна за прилагане;

- механизмът да остава сигурен поне за периода, през който подписа е валиден, т.е. времето, необходимо за разбиване на системата, да надхвърля

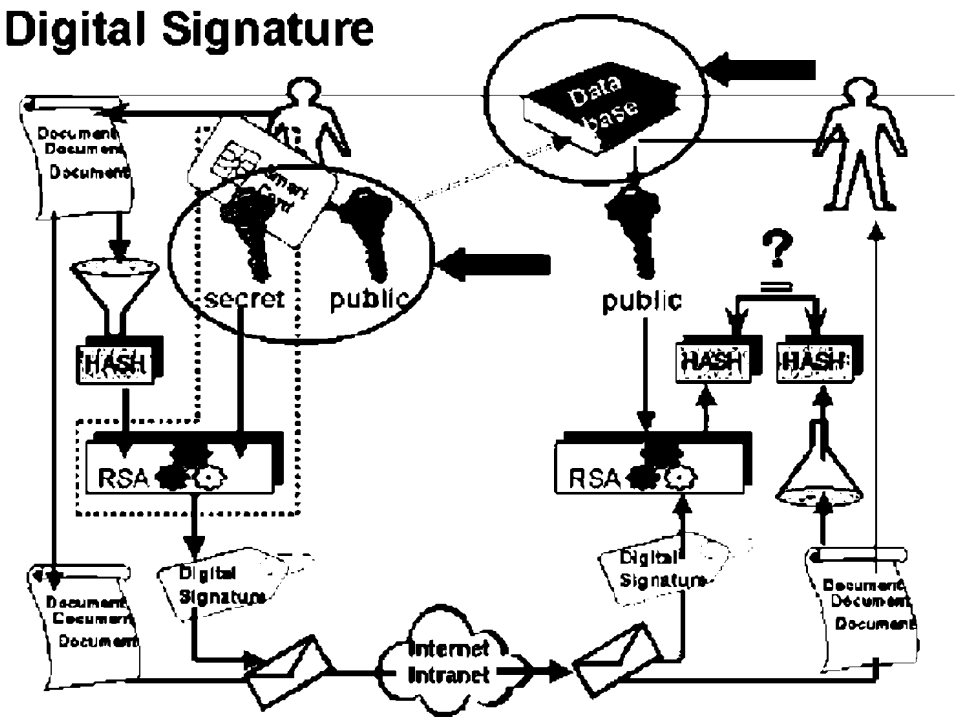
този период. Понятието “сигурен” включва практическата невъзможност друг освен автора на подписа да създаде смислени съобщения, които подложени заедно с подписа на проверяващата функция $\mathcal{V}_U$, да дават резултат *true*.

Трябва изрично да отбележим, че необходимостта от електронен подпис предполага две взаимно подозиращи се и/или готови да го оспорят страни. Затова не е възможно използването на общ ключ при реализацията на $\mathcal{S}_U$ и проверяващата трансформация $\mathcal{V}_U$. Това предопределя използването на асиметрични криптографски алгоритми (както е показано във фигура 4.4) с два ключа - частен и публичен. В литературата съществуват описания на МСЕП на базата на симетричен алгоритъм, но те изискват въвличане на TTP в процесите на генериране и проверка на подписа и то в твърде тромави процедури. Затова тези схеми не са и залегнали в международните документи.

Трансформацията $\mathcal{S}_U$ най-често е сложна функция, представляваща композиция (последователно прилагане) на хеш-функция h и криптографска трансформация $\mathcal{E}$. Фигура 4.4 изобразява блоковата структура на механизъм за електронен подпис с такава трансформация $\mathcal{S}_U$. При него съобщението (“електронното изявление"по термините на ЗЕДЕП) се предава в явен вид. Възможни са, обаче, и реализации, при които то е зашифровано. Подходящи за тази цел са симетричните алгоритми, защото са по-бързи и удобни за реализация. Освен това засекретяване на текста на съобщението предполага желание за това от двете страни (или нормативни изисквания), което предполага и сътрудничество между страните за грижливо съхранение на общия ключ.

Вторият тип механизъм за електронен подпис, който предполага извличане на съобщението от самия подпис, се използва само за много кратки съобщения (парола, ключ, идентификационен номер), тъй като прилагането на асиметричен алгоритъм за криптиране на дълги съобщения е бавен процес, изискващ много ресурси. Този механизъм въвлича функция, която създава излишък (разширява съобщението), за да не бъде всеки възможен подпис в действителност подпис на истинско съобщение, т.е. броят на възможните съобщения да бъде много по-малък от този на възможните подписи.
Този тип електронен подпис не е визиран от ЗЕДЕП, но би могъл да се ползва при разпространение на секретни ключове (за симетрични криптосистеми) или подписване на сертификати.

В някои случаи е необходимо да се подпише дадено съобщение, но без да се разкрива неговото съдържание. В тези случаи се прилага така нарече-



Фигура 4.4: Блокова схема на система за електронен подпис

ното ***сляпо подписване (blind signature)***. Този механизъм е реализиран в разпространения протокол електронни разплащания "Digicash". Ето накратко описанието на този механизъм:

Да предположим, че A иска B (частно лице, овластен субект, TTP или ДУУ и др.) да подпише неговия документ M, но без съдържанието му да стане известно на B. Нека e и d са, съответно, публичния и частен ключ на B. Тогава

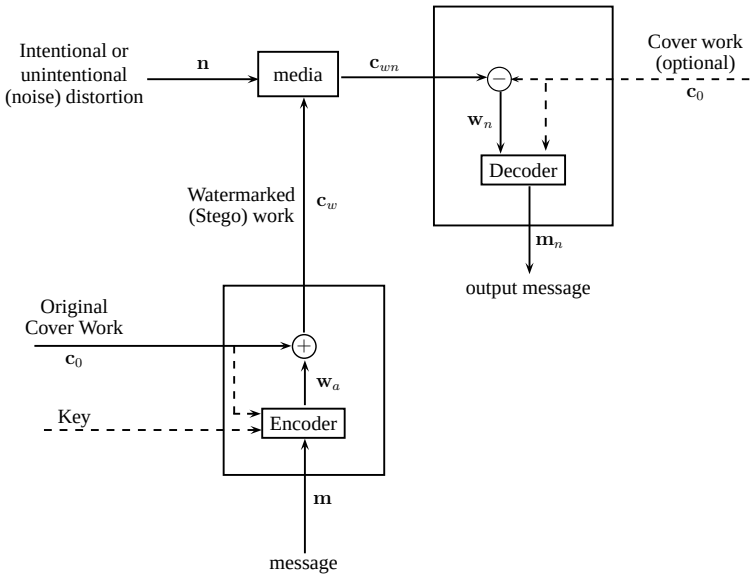
1. A генерира случайно число R, изчислява $M_1 = R^e M (mod\ n)$ и го изпраща на B.

2. Използвайки своя частен ключ, d, B пресмята $S = M_1^d = R^{ed}.M^d = R.M^d$ по модул n и го връща на A.

3. При получаването на S, A умножава S с R^{-1} за да намери M^d, което представлява подписания документ.

В литературата са описани и реализирани в специфични приложения

и редица други механизми от тип електронен подпис. Един такъв механизъм е така наречения *fail-stop* подпис. Той позволява да се открива, ако е направена фалшификация и съответния публичен ключ да се изважда веднага от употреба. Друг тип е така наречения *undeniable signature scheme*, който изисква сътрудничеството на подписващия за проверка на подписа. Реализира се например, когато подписващият осъществява достъп до нещо (например, сеиф) в дадено време. Целта е никой да не може да твърди, че този достъп е направен, когато не е осъществяван или във време различно от истинското. Тези и други подобни механизми излизат от рамките на нашите цели и няма да ги разглеждаме подробно.



Фигура 4.5: Watermarking (steganography) block diagram.

4.10 Стеганография и дигитален воден знак.

4.10.1 Основни дефиниции и свойства.

Дигиталният воден знак (digital watermarking) и стеганографията (steganography) са две много близки направления в съвременните информационни технологии. Те водят до важни практически приложения за защита на интелектуалната собственост и предаване на секретна информация, което обуславя интензивните изследвания в тези направления в последните години. И при двете се разработват методи за вмъкване на информацията (може да я считаме за поредица от битове) в дигитална медия представляваща изображения или звук посредством нейното изменение, но крайните цели леко се различават. Целта на стеганографията е да направи промяната незабелижима не само визуално (слухово), но и при изследване със специализиран софтуер, докато тази на дигиталният воден знак е да разкрива и възпрепядства пиратството, както и да доказва авторски права при необходимост.

Фигура 4.5 визуализира процеса на влагане на дигитален воден знак или стеганографска информациа в дигитален медиен обект. Пунктираните линии показват незадължителните части от този процес.

Вэзможните приложения на дигиталния воден знак са :

- **Проследяване на разпространението (Broadcast monitoring)**: За проследяване кога и къде в медиите е излъчван даден дигитален материал.
- **Идентификация на собственика на дигиталния материал (Owner identification)**
- **Подтвърждаване на авторство (Proof of ownership)**: Водният знак служи за доказване на авторство пред административни и съдебни инстанции.
- **Проследяване на разпространението (Transaction tracking)**: За идентифицирани на легални потребители, които обаче разпространяват нелегално придобития медиен продукт.
- **Аитентификация на съдържанието (Content authentication)**: Изпълнява ролята на електронен подпис, с който се установява дали има подмяна на съдържанието.
- **Контрол върху копирането (Copy control)**: Използва се от копиращите устройства за инфоране дали даденият медиен продект може да се копира.

Най-важните характеристика на дигиталния воден знак са:

- **Устойчивост (Robustness)**: Възможността водният знак да бъде разпознат след стандартните процедури по обработка на дигиталния обект: принтиране и сканиране, компресиране със загуба, геометрични преобразувания, прекарване през цифрови филтри и др. Тук не се включват преднамерените атаки базиращи се на наличие на информация за използвания алгоритъм.
- **Сигурност (Security of a watermark)**: Включва устойчивостта на преднамерени атаки като неоторизирани изтривания или вмъквания на информация.

Изискванията към стеганотрафските техники са аналогични с тази разлика, че те трябва да се оценяват от гледна точка на целите на стеганографията.

- **Незабележимост (Imperceptibility)** Наличието на скрита информация (модефицирана на дигиталния обект е неуловимо при гледане или слушане.

- **Капацитет (Capacity)**: максималният брой битове, които могат да се скрият в дадения дигитален обект, така че вероятността да бъдат открити от противниковата страна да е пренебрежима.
- **Ефективност (Embedding efficiency)**: Дефинира се като съотношението на броя на скритите битове към променените пиксели или брой скрити битове за секунда звук.
- **Статистическа неоткриваемост (Statistical undetectability)**: Вероятността да бъде открито наличието на стеганографска информация в даден дигитален обект базирайки се на статистически анализ на на обекта.
- **Сигурност (Security)**: Устойчивост на активни или вирусни ятаки.
- **False alarm rate**: Вероятността стеганографският алгоритъм да даде информация за скрита информация, когато де факто такава няма.

Информацията може да се вмъква чрез промяна в следните две области на дигиталния обект.

- **Пространствената област (Spatial domain)**: Това е терминът, който се ползва за означаване на стандартната форма на дигиталния обект, която представя медийния продукт. Например, черно-белите изображения се представят с матрица от цели числа в интервала [0, 255].
- **Честотна област (Frequency domain)** Терминът използван за дигиталния обект получен след прилагане върху оригинала дискретно косиносуво, Фуриерово или Адамарово преобразование, уейвлет или други подобни трансформации. Съобщението се вмъква в трансформирания обект и след това се подлага на обратната трансформация.

Процесът на вмъкване на информация в дигитални обекти на математически език се описва като изображение (***embedding process***)

$$\mathcal{E} : \left| \begin{array}{lcl} (\mathfrak{P}, \mathfrak{T}, \mathfrak{K}) & \rightarrow & \mathfrak{P} \\ (\mathbf{c}_0, \mathbf{m}, \mathbf{k}) & \rightarrow & \mathbf{c}_m = \mathcal{E}(\mathbf{c}_0, \mathbf{m}, \mathbf{k}) \end{array} \right. ,$$

където $\mathbf{c}_0$ е обработвания обект (*cover object*), който е елемент на $\mathfrak{P}$ - множеството от възможните дигитални обекти (например изображения), $\mathbf{m} \in \mathfrak{T}$ е съобщението предвидено за влагане в $\mathbf{c}_0$, а $\mathbf{k}$ е случайно избран ключ от пространството от ключовете $\mathfrak{K}$.

Съответно, процесът на извличане на скритата информация се състои в прилагане на ***оценъчна функция (decision function)*** $\mathcal{D}$ към полученото, евентуално повредено копие $\mathbf{c}_{mn}$ на изпратения модефициран дигитален обект $\mathbf{c}_m$:

$$\mathbf{m}' = \mathcal{D}(\mathbf{c}_{mn}), \qquad \text{или} \quad \mathbf{m}' = \mathcal{D}(\mathbf{c}_{mn}, \mathbf{c}_0), \quad \text{ако } \mathcal{D} \text{ изисква оргиналът.}$$

Оценъчната функция се основава на фиксирана метрика (наричана ***оценъчна метрика (detection metric)***) в пространството на обектите. Повредите по предадения дигитален обект могат да се интерпретират като адитивен гаусов шум $\mathbf{n}$, т.е. $\mathbf{c}_{mn} = \mathbf{c}_m + \mathbf{n}$. Но много често на практика $\mathbf{c}_{mn} = \mathbf{c}_m$.

В практическите реализации процесът на вмъкване е последователно прилагане на две изображения. Първото, $\mathcal{M}$, трансформира изображението в матрица над Z_q - пръстена от остатъци по модул q, или над крайно поле $GF(q)$ с q елемента. Множеството от такива матрици ще го означаваме с $\mathfrak{M}$, т.е.,

$$\mathcal{M} : \ (\mathfrak{P}, \mathfrak{T}, \mathfrak{K}) \ \rightarrow \ \mathfrak{M}.$$

Преобразуването $\mathcal{M}$ зависи от дигиталния обект и от избрания ключ. След внимателен статистически анализ някои от пикселите се маркират като "мокри (wet)" и не се променят. Информацията се влага в останалите, наричани "сухи (dry)" пиксели. В каква последователност се влагат информационнитe битове зависи от избора на ключа.

Втората трансформация наричана най-често ***embedding function*** се базира на лазлични математически методи. Тук ще опишем алгоритъм ползващ кодове изправящи грешки. За втората трандформация ще запазим означението $\mathcal{E}()$.

Методът "Синдромно влагане (Syndrome Embedding)"

Този метод, известен още като *матрично вмъкване (matrix embedding)*, е метод, при който двете кореспондиращи страни се договарят за проверочна матрица $\mathbf{H}$ на линеен код, а секретното съобщение се извлича като редица от синдроми (относно $\mathbf{H}$) извлечени от обработения дигитален обект. В този случай $\mathcal{M}$ трансформира дигиталния обект, който всъщност е матрица от 8, 16, или 12 (за CD audio) битови естествени числа (т.е., елементи на $\mathbb{Z}_{2^b}$, $b = 8, 12, 16$) в $n \times N$ матрица $\mathbf{D}$ над $\mathbb{F}_q$ или $\mathbb{Z}_q$.

Нека $\mathbf{H}$ е $r \times n$ проверочна матрица на линеен код над $\mathbb{Z}_q$, или крайно поле $\mathbb{F}_q$ от q елемента. Съобщението (евентуално криптирано) също се трансформира в $r \times N$ матрица $\mathbf{m}$ над $\mathbb{F}_q$.

Algorithm:

1. Изчисляваме $\mathbf{S} = \mathbf{m} - \mathbf{HD}$, т.е. $\mathbf{m} = \mathbf{S} + \mathbf{HD}$

2. Намираме $n \times N$ матрица $\mathbf{E}$, такава че $\mathbf{S} = \mathbf{HE}$
3. Пресмятаме $\mathbf{V} = \mathbf{D} + \mathbf{E}$
4. Конструираме дигитално изображение $\mathbf{c}_m$, такова че $\mathcal{M}(\mathbf{c}_m, \mathbf{k}) = \mathbf{V}$ и изпращаме $\mathbf{c}_m$.

Получателят

1. Определя $\mathbf{V} = \mathcal{M}(\mathbf{c}_w, \mathbf{k})$
2. Пресмята $\mathbf{HV} = \mathbf{HD} + \mathbf{HE} = \mathbf{HD} + \mathbf{S} = \mathbf{m}$

Ефективността при синдромното влагане използвайки код с $r \times n$ проверочна матрица е

$$E_f = \frac{r \log_2 q}{n} \quad \text{bits/pixel}$$

Пример 4.10.1 Нека $\mathbf{H}$ е проверочна матрица на $[8, 4]$ разширен двоичен код на Хеминг:

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

Множеството от синдромите се състои от всички двоични вектори с дължина 4, т.е. $\mathbf{Synd} = \mathbb{Z}_2^4$:

$$\mathbf{Synd} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1 \end{pmatrix}$$

Матрицата от вектор-грешките съответстващи на горните синдроми е

$$\mathbf{Es} = \begin{pmatrix} 0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1 \end{pmatrix}$$

Да отбележим, че поради направения избор на $\mathbf{H}$ номерът на всеки стълб в $\mathbf{Synd}$ (и на съответстващият стълб на $\mathbf{Es}$) съвпада със стойността му като десетично число плюс единица, т.е.

$$j = (8, 4, 2, 1)\mathbf{Synd}(j) + 1.$$

Разбира се за декодиране можем да използваме всеки алгоритъм за декодиране на циклични кодове (с подходящ избор на $\mathbf{H}$), но те ще бъдат с по-голяма сложност.

Обработваното изображение $\mathbf{I}$ ($a \times b$ матрица над $\mathbb{Z}_{256}$) се трансформира в $\mathbf{D} \equiv \mathbf{I} \pmod 4$. След това тази матрица се преобразува в $8 \times ab/4$ двоична матрица (вцъкването на информацията става в двата най-малко значими бита). Съобщението се трансформира в $4 \times ab/4$ двоична матрица (с допълване с нули, ако е необходимо). След това се изпълнява описаният по-горе алгоритъм. i-тият стълб на $\mathbf{E}$ е $(8, 4, 2, 1)\mathbf{S}(i) + 1$ стълб на $\mathbf{Es}$.

Библиография

[1] Денев, Й., Р. Павлов, Я. Деметрович, *Дискретна математика*, Наука и изкуство, София, 1984.

[2] Манев, К., *Увод в дискретна математика*, КЛМН, София, 2003.

[3] "Закон за електронния документ и електронния подпис", ДВ бр. 34 от 6.04.2001.

[4] Питерсон, У., Э. Уэлдон, *Коды, исправляющие ошибки,* МИР, Москва, 1976.

[5] Cormen, Th. et al., *Introduction to Algorithms*, MIT Press, 2nd edition, 2001.

[6] FIPS-xxx, http://csrc.nist.gov/publications/fips/index.html и http://csrc.nist.gov/CryptoToolkit/

[7] Goldwasser S.,S.Micali and C.Rackoff, The knowledge complexity of interactive proof-systems, SIAM J. Comput. v.18 (1989), no.1, 186-208.

[8] PKCS 1. "The public key cryptography standards - Part 1: RSA encryption standard", version 1.5, 1993, and version 2.0, 1998, *http://www.rsa.com*.

[9] RFC xxxx, http://www.ietf.org

[10] Shannon, C.E., "A mathematical theory of communication", Bell Syst. Tech. J. 27 (1948), 379-423 и 623-656.

[11] Shannon, C.E., "Communication theory of secrecy system", Bell Syst. Tech. J. 28 (1949), 657-715.

[12] Sklar, B., "A structured overview of digital communications", IEEE Commun. Mag. (1983), August, p. 6.