

# Кодове на Рид-Соломон

---

Кристиян Стоименов

13 февруари 2026 г.

Факултет по математика и информатика,  
**Ефективна реализация**  
**на математически алгоритми**  
**и концепции**



## Дефиниция

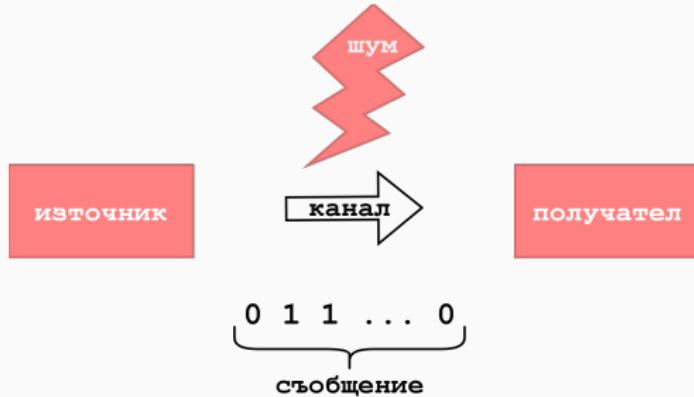
*Код на Рид-Соломон наричаме ...*



Какво е код?

---

*Шумозащитно кодиране* наричаме набор от математически средства и алгоритми, чрез които грешки, възникнали при предаване на информация по канал между два агента, биват разпознати и поправени.

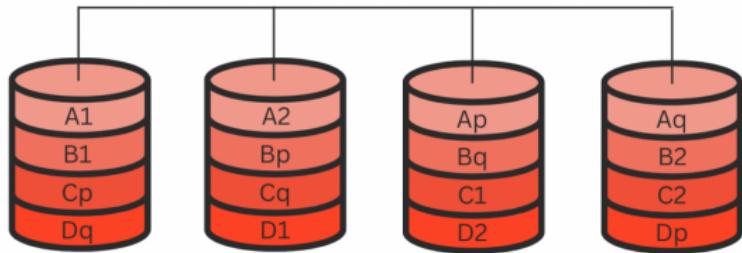


Фигура 1: Опростен модел на предаване на информация през комуникационен канал.



Фигура 2: Voyager 2 е безпилотна космическа сонда на НАСА, изстреляна през 1977 г., която изследва външните планети на Слънчевата система. Единственият апарат, посетил Уран и Нептун. Снимката е взема от [NAS26].

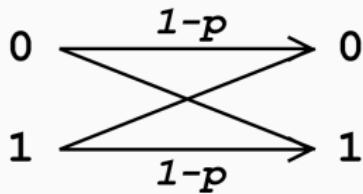
## RAID 6



Фигура 3: RAID-6 е схема за нареждане на масив от дискове с цел максимално бързодействие и едновременно с това издръжливост при грешка. [Ана20]

- Приложенията на шумозащитното кодиране са много. Някои други от тях включват CD [99] и QR кодовете [qr].
- Основния фактор, допускащ ефективността на шумозащитното кодиране е следната теоремата за шумния канал (Шанън, 1948) [Mac03]. В общо линии тя гласи, че за произволен шумен канал, съобщения могат да се предават без загуба на информация, стига преносът да не надхвърля капацитета - максималната пропускателна способност, на канала.

Какво представлява шумът?



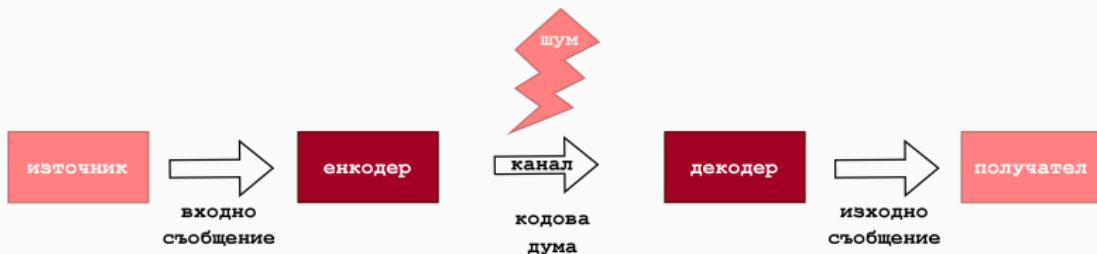
$p := P(\text{ обръщане })$



Фигура 4: Двоичен симетричен канал.

*Шумозащитен код*, или за по-кратко само *код*, наричаме систематичен подход за съпоставяне между съобщение и кодова дума. Това съпоставяне се разглежда в две посоки:

- Посоката „съобщение  $\mapsto$  кодова дума“ се нарича *кодиране*. Кодирането изменя съобщението, добавяйки *проверочни символи*<sup>1</sup>, такива че входното съобщение да бъде възстановено в случай на промяна по време на предаване.
- Обратната, „кодова дума  $\mapsto$  съобщение“, се нарича *декодиране* и именно тя е същината на един шумозащитен код. Декодирането използва въведените от кодирането *проверочни символи*, за да поправи възникнали грешки от предаването, *ако свойствата на шумозащитния код позволяват това*.



Фигура 5: Опростен модел на предаване на информация през комуникационен канал, използвайки шумозащитен код.

<sup>1</sup>на англ. redundancy

## Код с повторение

Нека разгледаме един елементарен пример за шумозащитен код, за да илюстрираме по-ясно модела на предаване на информация през комуникационен канал.

Следва да фиксираме няколко основни параметъра на нашия код - дължина на съобщението и брой проверочни символа. От тях получаваме директно и максималния брой грешки, които шумозащитният код е способен да възстанови.

Нека за нашите цели съобщението се състои от 16 бита, а броят на проверочните символи е 32 бита. Така кодовата дума става за дължина 48 бита.

Използваме следния алгоритъм за кодиране. Ефектът от него е, че всеки бит от съобщението се предава точно по три пъти, т.е. повтаряме съобщението три пъти. Оттам и името на този код.

---

```
1 u64 encode(u16 msg)
2 {
3     u64 codeword=msg|msg<<16|msg<<32;
4     return codeword;
5 }
```

---

Декодирането също става очевидно - за всеки от битовете в получената кодова дума избираме стойност за съобщението, съответстваща на модата от стойностите на трите копия.

```
1  u16 decode(u64 codeword)
2  {
3      u16 msg=0;
4      for (int i=0; i<16; ++i
5      {
6          int l1=1<<i, l2=1<<(16+i), l3=1<<(32+i);
7          int b1=!!(codeword&l1), b2=!!(codeword&l2),
8              b3=!!(codeword&l3);
9          msg |= b1+b2+b3>>1<<i;
10     }
11     return msg;
12 }
```

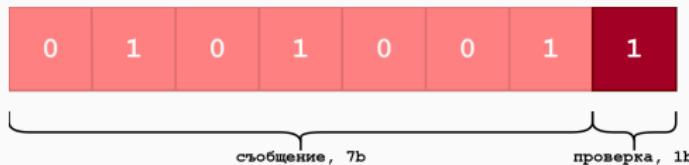
## Основни понятия

---

Разграничаваме *откриване* от *поправяне* на грешка. В първия случай имаме единствено информация, че полученото съобщение се различава от изпратеното. Във втория случай знаем не само това, ами и кои точно битове са били промени.

Друг вариант за грешка е обаче някои от битовете да бъдат пропуснати, а не обърнати. Също така възникват и други особености при бройни системи, които не са двоичната - в такива случаи не е достатъчно да знаем къде е възникнала грешката, тъй като има повече от една възможна друга стойност.

Прост пример за това е т.нар. *проверка по четност*. Нейната идея е следната. Ако имаме съобщение, съставено от например 7 двоични символа, то проверка по четност реализираме, добавяйки символ с такава стойност, че броя на всички единици сред съобщението да бъде четен. Очевидно в случая можем да забележим единствено нечетен брой грешки, а да поправим, не можем да нито една от тях.



Фигура 6: Прост пример за проверка по четност.

## Лінейни блокови кодове

---

Нека разгледаме какво точно представлява един шумозащитен код.

Оттук настетне считаме, че източник съставя съобщения използвайки символи от крайно поле  $\mathbb{F}$  с мощност  $q$ . Тъй като полетата, които ни интересуват са крайни, обичайно ги бележим с  $GF(q)$ .

### Дефиниция

$(n, k)$  блоков код  $\mathcal{C}$  над азбука от  $q$  символа наричаме множество от  $q^k$  вектора, всеки от които има дължина  $n$ .  
Тези вектори наричаме кодови думи.

На всеки блоков код съответства енкодер, който съпоставя на съобщението  $m \in \mathbb{F}^k$  съответна кодова дума  $c \in \mathcal{C}$ .

Често кодовете, които ни интересуват имат следното свойство.

### Дефиниция

Нека блоковия  $(n, k)$  код  $\mathcal{C}$  над  $GF(q)$  образува линейно подпространство на линейното пространство от наредените  $n$ -торки  $\mathbb{F}_q^n$ . Тогава за  $\mathcal{C}$  казваме, че е **линеен** блоков код.

Използвайки аксиомите на линейно пространство, получаваме начин за „конструиране“ на нови кодови думи. Например, ако  $c_1, c_2 \in \mathcal{C}$ , то  $c_1 + c_2 \in \mathcal{C}$ .

Третата основна характеристика на блоковите кодове освен дължината на съобщението и дължината на кодовата дума е неговото минимално разстояние. Именно то определя максималният брой грешки, които кодът може да поправи. За целта ни е необходима следната метрика.

## Дефиниция

*Разстояние на Хеминг* между две кодови думи  $a, b \in \mathcal{C}$ , където  $\mathcal{C}$  е  $(n, k)$  код, наричаме

$$d(a, b) := |\{ i \mid a_i \neq b_i \}|$$

## Дефиниция

Минимално разстояние на  $\mathcal{C}$  наричаме

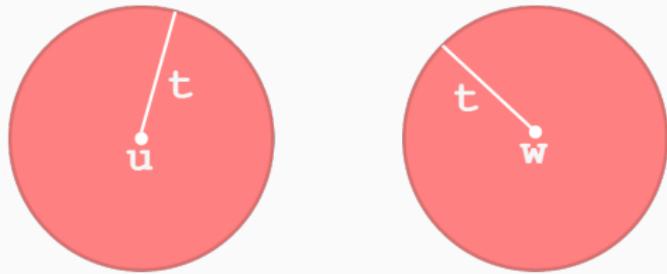
$$d(\mathcal{C}) := \min\{ d(a, b) \mid a, b \in \mathcal{C}, a \neq b \}$$

Стойността

$$t = \left\lfloor \frac{d - 1}{2} \right\rfloor$$

наричаме радиус на опаковка и съвпада с максималния брой грешки, които могат да бъдат поправени от кода  $\mathcal{C}$ .

( $n, k$ ) код  $\mathcal{C}$ , при който  $d := d(\mathcal{C})$ , бележим като  $(n, k, d)$  код.



Фигура 7: Абстрактно представяне на кодови думи заедно с радиус на опаковка.

Оттук насетне ще разглеждаме линейни блокови кодове. Нека  $\mathcal{C}$  е  $(n, k, d)$  линеен код. От линейната алгебра ни е известно понятието *базис* и знаем също така, че произволен елемент  $c \in \mathcal{C}$  може да се представи (по единствен начин) като *линейна комбинация* на базисните вектори. Нека  $g_0, \dots, g_{k-1}$  образува базис на  $\mathcal{C}$ . Тогава за произволна кодова дума  $c \in \mathcal{C}$  е вярно, че

$$c = \sum_{i=0}^{k-1} m_i g_i$$

Или, записвайки вектори  $g_i$  като редове на една матрица  $G$

$$G = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix}$$

и съобщението като вектор-ред

$$m = [m_0 \cdots m_{k-1}]$$

кодовата дума, съответстваща на съобщението  $m$  получаваме като

$$c = mG$$

Матрицата  $G$  се нарича **пораждаща** за кода  $\mathcal{C}$ . Чрез нея директно получаваме алгоритъм за кодиране - а именно чрез умножение.

Нека за илюстрация разгледаме един от първите използвани шумозащитни кодове - (7, 4) кодът на Хеминг. Чрез него 4-битови съобщения се кодират в 7-битови кодови думи, използвайки 3 проверочни бита. Една негова пораждаща матрица е

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

Тогава съобщението  $m = [1001]$  кодираме по следния начин

$$c = mG = [1100101]$$

Използвайки метриката разстояние по Хеминг схемата на декодиране се нарича *метод на максималното правдоподобие*. Имаме следната

### Теорема

За двоичен симетричен канал с вероятност за грешка  $p < \frac{1}{2}$  декодирането по метода на максималното правдоподобие е еквивалентно на декодиране до най-близката кодова дума.

Сега ще разгледаме как работи това декодиране чрез таблица на Слепян.

Преди да посочим алгоритъма, ни необходима е следната

## Дефиниция

Нека  $\mathcal{C}$  е двоичен  $(n, k)$  код. *Съседен клас* на кода  $\mathcal{C}$ , определен от вектора  $y$  наричаме множеството

$$y + \mathcal{C} := \{ y + c \mid c \in \mathcal{C} \}$$

*Лидер* на съседен клас  $y + \mathcal{C}$  наричаме елемента

$$\hat{c} := \min_{c \in y + \mathcal{C}} \text{wt}(c)$$

където  $\text{wt}(c)$  е *теглото* на кодовата дума  $c$ , т.е. броят на ненулевите битове из двоичната кодова дума.

Нека разгледаме сега алгоритъма за декодиране на получена дума.

Нека източникът е изпратил кодовата дума  $c$ , а поради шума на канала се е добавила някаква грешка  $e$ . До декодера достига (в общия случай) некодова дума  $y = c + e$ . Тогава е вярно, че

$$y + c = (c + e) + c = (c + c) + e = e$$

тъй като в  $GF(2)$  събирането съвпада с операцията изключващо или, а то има свойството  $\forall a (a \oplus a = 0)$ .

Оттук сме уверени, че получената дума  $y$  и грешката  $e$  принадлежат на един и същи съседен клас, както и, че всички възможни грешки при получена дума  $y$  са именно думите от този съседен клас.

$0$	$c_1$	$c_2$	.....	$c_{M-1}$	
$e_1$	$e_1 + c_1$	$e_1 + c_2$	.....	$e_1 + c_{M-1}$	съседни класове с лидери с тегло $\leq t$
.....	.....	.....	.....	.....	
$e_s$	$e_s + c_1$	$e_s + c_2$	.....	$e_s + c_{M-1}$	съседни класове с лидери с тегло $\leq t$
.....	.....	.....	.....	.....	
$e_{s+1}$	$e_{s+1} + c_1$	$e_{s+1} + c_2$	.....	$e_{s+1} + c_{M-1}$	съседни класове с лидери с тегло $> t$
.....	.....	.....	.....	.....	
$e_N$	$e_N + c_1$	$e_N + c_2$	.....	$e_N + c_{M-1}$	

Таблица 1: Стандартна таблица на Слепян

Нека разгледаме една примерна стандартна таблица на Слепян.

Както забелязвате, този метод е крайно непрактичен по множество причини. Съществуват подходи за неговото опростяване - най-вече за намаляване на таблицата и едновременно с това броя проверки, които са необходими (декодиране чрез синдроми), но за да получим качествено по-добри резултати, са необходими различни свойства на кодовете, които използваме.

Така достигаме до цикличните кодове.

## Циклични кодове

---

## Дефиниция

Нека имаме вектор  $b = [b_0, b_1, \dots, b_{n-1}]$ . Циклично преместване на  $b$  наричаме вектор  $b'$ , получен като координатите са разместени така, че втората е записана на първо място, третата на втората и т.н.. Първата е записана на последно място.

$$b' := [b_1, b_2, \dots, b_0]$$

## Дефиниция

Линеен блоков код, който заедно с всяка своя кодова дума  $c = [c_0, \dots, c_{n-1}]$  съдържа още и неговото циклично завъртане  $c' = [c_1, \dots, c_0]$ , се нарича **цикличен код**.

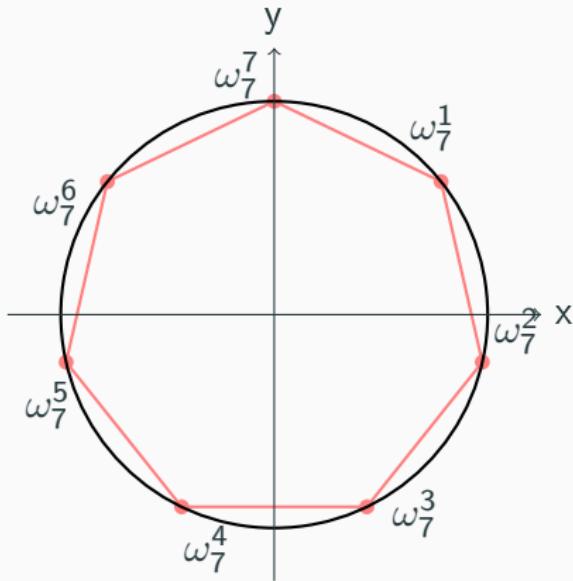
При работа с циклични кодове е удачно да имаме дефинирани не само операциите от линейното пространство, но още и **умножение на вектори** над полето  $GF(q)$ .

Поради тази причина се използва следното съответствие

$$c = [c_0, \dots, c_{n-1}] \mapsto c(x) = c_0 + c_1x + \dots + c_{n-1}x^{n-1}$$

и още по-конкретно израза

$$c(\omega) = c_0 + c_1\omega + \dots + c_{n-1}\omega^{n-1} \quad \text{където } \omega^n = 1$$



Фигура 8: Решения на уравнението  $x = \sqrt[7]{1}$ . Забележка: Решенията са точно елементите на цикличната група  $\mathbb{C}_7$ .

Цикличните кодове, понеже са линейни, имат своя пораждаща матрица. Но освен нея те имат и пораждащ полином -

### Дефиниция

*Пораждащ полином*  $g(x)$  на цикличен код  $\mathcal{C}$  се нарича полиномът от най-ниска степен, съответстващ на ненулева кодова дума  $g \in \mathcal{C}$ . Бележим  $\mathcal{C} = \langle g \rangle$ .

Той е особен, поради свойствата, следващи от следната

## Теорема

Ако имаме цикличен код  $\mathcal{C} = \langle g \rangle$ , то:

- полиномът  $g(x)$  дели  $x^n - 1$  без остатък;
- вектор с е кодова дума ТСТК  $g(x)$  дели  $c(x)$ ;
- ако  $\deg(g) = r$ , то  $\mathcal{C}$  е  $(n, n - r)$  код, а пораждаща матрица  $G$  изглежда така -

$$G = \begin{pmatrix} g_0 & g_1 & \dots & g_r & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & \dots & g_r & 0 & \dots & 0 \\ \dots & \dots \\ 0 & 0 & \dots & 0 & g_0 & g_1 & \dots & g_r \end{pmatrix}$$

Можем да видим един примерен цикличен код, който се поражда от полинома  $x^3 + x + 1$ , и има кодови думи с дължина 7 бита.

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

Ха! Изненада! Това отново е  $(7, 4)$  кодът на Хеминг! 😊

Сега можем да посочим как работи кодирането на съобщение, използвайки цикличен код.

Нека имаме  $\mathcal{C} = \langle g \rangle$ . Входното съобщение представяме чрез информационен полином

$$i(x) = i_0 + \cdots + i_{k-1}x^{k-1}$$

Имаме два подхода на кодиране, определени от това дали съобщението е част от получената кодова дума, или е „разбъркано“. Първият се нарича систематичен и именно него ще обясним тук. В него информационната последователност образува старшите членове на кодовата дума, т.е.

$$c(x) = x^{n-k}i(x) + t(x)$$

Знаейки, че  $g(x) \mid c(x)$ , можем да използваме теоремата на Евклид -

$$x^{n-k} i(x) = g(x)q(x) + r(x)$$

и тогава

$$c(x) = x^{n-k} i(x) + t(x) = g(x)q(x) + (r(x) + t(x))$$

където  $r(x) + t(x)$  е от степен по-малка от  $n - k$  и се дели на  $g(x)$ , откъдето  $r(x) = -t(x)$ .

Нека разгледаме един пример. Отново ще разгледаме познатия  $(7, 4)$  код на Хеминг. За него имаме, че  $\mathcal{C} = \langle g \rangle$  при  $g(x) = x^3 + x + 1$ . Да вземем например съобщението 1001. На него съпоставяме информационен полином

$$i(x) = 1 + 0x^1 + 0x^2 + 1x^3 = x^3 + 1$$

Кодовата дума получаваме като

$$c(x) = x^3 i(x) + t(x) = x^6 + x^4 + x^3 + t(x)$$

Търсим  $t(x)$ , разделяйки  $x^3 i(x)$  на  $g(x)$ , т.e.  $x^6 + x^4 + x^3$  на  $x^3 + x + 1$ . Получаваме  $q(x) = x^3$  и  $r(x) = 0$ .

## Какво е код на Рид-Соломон?

---



**Как се използват Рид-Соломон  
кодовете?**

---



Какви приложения имат  
Рид-Соломон кодовете?

---



**Благодаря за вниманието!**

## Литература

---

- [99] ***IEC 60908:1999 – Compact Disc Digital Audio System.*** Defines Cross-Interleaved Reed–Solomon Coding (CIRC) for error correction in audio CDs. International Electrotechnical Commission, 1999. URL: <https://standards.iteh.ai/catalog/standards/clc/7e75f508-8dea-4477-8151-dfac03511623/en-60908-1999>.
- [Ana20] Anadoxin. ***Reed–Solomon Error Recovery in RAID-6.*** <https://anadoxin.org/blog/error-recovery-in-raid6.html>. достъпено на 02.02.2026. 2020.
- [Mac03] David J. C. MacKay. ***Information Theory, Inference, and Learning Algorithms.*** <https://www.inference.org.uk/iitprnn/book.pdf> достъпено на 02.02.2026. Cambridge University Press, 2003. ISBN: 0-521-64298-1.
- [NAS26] NASA. ***Voyager 2 – NASA Science.*** <https://science.nasa.gov/mission/voyager/voyager-2/>. достъпено на 02.02.2026. 2026.