

6.7. Коды Рида-Шамиера

Булевы функции: $\mathcal{F}: \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2$
АНФ: $; \oplus, 0, 1$

Ex $\mathcal{F}(x_1, x_2, x_3) = x_1 x_2 \oplus x_3 \oplus 1$ — полином Жегалкина (однозн. предст. в АНФ)

x_1	x_2	x_3	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$\deg(F) = 2$

Опн Степень булевой ф-ии — кол-во множителей в самой линейной слагающей в АНФ

Рассмотрим все бул. ф-ии \mathcal{F} от n пер-х: $\deg(\mathcal{F}) \leq r$

(мин. замыкание этого багуса)

\vdash	x_1, \dots, x_n	(n)
\wedge	$x_1 x_2, \dots, x_i x_j, \dots$	(C_n^2)
$\wedge\wedge$	$x_1 x_2 x_3, \dots$	(C_n^3)
\vdots	\vdots	\vdots
$x_1 x_2 \dots x_r, \dots$	(C_n^r)	число бул. ф-й соотв. сист.

Линии векторов знач-й: 2^n
Мощность множ-ва: $2^{1+c_1+c_2+\dots+c_n}$

Опн Код Рида-Шамиера порядка r ($\text{Rell}(r, n)$) — это мин-во всех векторов знач-й бул. ф-й от n пер-х степени $\leq r$

Его параметры: $[2^n, 1 + C_n + C_n^2 + \dots + C_n^r, 2^{n-r}]$

Он линейный (можно замкнутость проверить)

Упр Д-ть, что кодовое расст-е: 2^{n-r}

6.8. Теорема Шеннона о скорости кодирования.

Опн Пропускная способность двоичного симметричного канала связи:

$$C = 1 - H_2(p)$$

тут p — вероятность искажения бита

$$H_2(p) = p \cdot \log \frac{1}{p} + (1-p) \cdot \log \frac{1}{1-p}$$

$$p=0: H_2(p)=0 \quad C=1 \quad - \text{лучше}$$

$$p=1: H_2(p)=1 \quad C=0 \quad - \text{не имеет смысла}$$

Опн Скорость передачи информации по каналу связи: $R = \frac{k}{n}$ — длина инф. блока
— длина код. слова

Теорема Ченнаса о скорости кодирования: (1948)

Пусть R - число такое, что $0 < R \leq C$, где C - пропускная способность канала связи.

Тогда $\exists E > 0$ \exists код, исправляющий ошибки, достаточно большой длины n со скоростью кодирования R , для ктр вероятность ошибки декодир-я меньше E .



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

Но не говорилось о том, как эти коды построить.

А вот в 1962 Роберт Гамагер построил так назыв. коды, обещанные теоремой Ченнаса

Вот только не сразу обозначились, какие они замечательные, а в начале 2000х (когда хотели что нет патента и можно параллельно)
Да и вообще у них много плюсов, всем советую! Это LDPC-коды

7. Коды с малой плотностью проверок на плотность

(Locally Dense Parity Check или LDPC-коды)

Опн LDPC-код - лин. код, проверочн. си-ца ктр сильно разрежена, т.е. число единиц в к-ре много меньше числа нулей.

Опн LDPC-код **регулярный**, если число единиц в каждой строке - константа (w_r)_{row} и число единиц в каждой столбце - тоже константа (w_c)_{column}

$$H = \begin{array}{|c|c|c|c|} \hline & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot \\ \hline \end{array} \quad m$$

$$w_r \cdot m = w_c \cdot n$$

(просто общее число "1" считали)

Опн Конструктивная скорость регулярного LDPC-кода: $R = 1 - \frac{w_c}{w_r}$

Помним, что скор. кода $\frac{k}{n} \geq \frac{n-m}{n} = 1 - \frac{m}{n} = 1 - \frac{w_c}{w_r}$
 m -число строк в H , m' -число линий строк в H $k = n - m' \geq n - m$

Т.е. конструктивная скорость - это та, ктр можешь гарантировать

Теперь посмотрим конструкцию Гамагера:

Построим (w_r, w_c) регулярный LDPC-код.

Пусть

$$H_0 = \begin{pmatrix} 1 & \dots & 1 & 0 & 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 & \dots & 0 \\ \vdots & & \vdots & & & & \vdots & & & & \vdots \\ & & & & & & & & & & w_r \\ & & & & & & & & & & \end{pmatrix} \quad (\text{н. должно делиться на } w_r)$$

- это провер. си-ца для $(w_r, 1)$ регу. LDPC-кода

$$H = \begin{pmatrix} \pi_1(H_0) \\ \pi_2(H_0) \\ \vdots \\ \pi_{w_c}(H_0) \end{pmatrix}, \text{ где } \pi_i - \text{перестановка на столбцах } H_0$$

$d \rightarrow$ минимум с ростом n

Как выбрать перестановки так, чтобы гарантиру. кодовое расст-е?

откр. вопрос

Лекодирование:

Граф Таннера:

$$x = (x_1, \dots, x_n)$$

$$Hx^T = 0 \iff x - \text{коодыны}$$

Ex

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

(cor=4, w_c=2)
результат.



Лекодир-е LDPC-кода:

Пусть (x_1, \dots, x_n) - переданное код слово, (y_1, \dots, y_m) - получ. вектор.

Пусть c_1, \dots, c_m - проверки, определяемые H , т.е. ур-я на биты вектора y .

Так для примера выше:

$$c_1: y_1 + y_2 + y_3 + y_4 = 0$$

$$c_2: y_2 + y_4 + y_5 + y_6 = 0$$

$$c_3: y_1 + y_2 + y_5 + y_6 = 0$$

Алгоритм:

- 1 Вычисляем проверки для y (проверки для каково, выполняется ли она)
- 2 Изменяембит y_i на $y_i \oplus 1$, если более половины проверок, в ктр входит y_i , не выполняется и возвращаемся к шагу 1.

Продолжаем выполнять 1,2 до тех пор, пока не выполнено:

- $\text{Syn}(y) = 0$ ($Hy^T = 0$) \leftarrow тут всё хорошо!
 - $\text{Syn}(y)$ не изменилось
 - y не изменилось
- $\left. \begin{matrix} \text{тут надо что-то} \\ \text{делать с алгоритмом.} \end{matrix} \right\}$