13.12.20. Дискр. матем. ДЗ от семинара 15.

① $\|f\| = 2^{n/2}$

$\max\limits_f L(f) = ?$ ответ: $2^{n/2} \le \max\limits_f L(f) \le 3 \cdot 2^{n/2}$

Решение: нижняя оценка: по теореме 11.4: $|F| = C_{2^n}^{2^{n/2}}$ — она применима, т.к. $n+1 = o\left(2^{\frac{n}{2}}\right)$

$$\max L(f) \ge \frac{\log C_{2^n}^{2^{n/2}}}{\log\log C_{2^n}^{2^{n/2}}} \ge \frac{2^{n/2}\log\left(\frac{2^n}{2^{n/2}}\right)}{\log\left(2^{n/2}\cdot\log\left(\frac{2^n}{2^{n/2}}\right)\right)} = \frac{2^{n/2}\cdot\frac{n}{2}}{\frac{n}{2}+\log\left(\frac{n}{2}\right)} \sim 2^{n/2}$$

$C_n^k = \frac{n(n-1)\dots(n-k+1)}{k!} = \frac{n}{k}\cdot\frac{n-1}{k-1}\dots\frac{n-k+1}{1} \ge \left(\frac{n}{k}\right)^k$

Верхняя оценка: (получим правильный порядок)

Представим f в виде СДНФ:

$f = \bigvee\limits_{i=1}^{2^{n/2}} K_i(x_1 \dots x_n)$, где K_i — эл. конъюнкция.

Запишем их в табличку (там $2^{n/2}$ строк и n столбцов) (⇒ можно за $n\cdot 2^{n/2}$ операций — можно хотим меньше)

K_1	$x_1^{\alpha_{11}}$	$x_2^{\alpha_{12}}$	…	$x_n^{\alpha_{1n}}$
K_2				
⋮				
K_i				
⋮				
$K_{2^{n/2}}$	$x_1^{\alpha_{2^{n/2},1}}$		…	$x_n^{\alpha_{2^{n/2},n}}$

Табличку $2^{n/2}\times n$ — можно реализовать за $2^{n/2}\cdot n$ операций, но мы хотим за меньше.

Сделаем за $\text{const}\cdot 2^{n/2}$.

Будем собирать $K_i(x_1 \dots x_n)$ из двух частей: $K_i(x_1 \dots x_n) = K_i(x_1 \dots x_{\frac{n}{2}}) \& K_i(x_{\frac{n}{2}+1} \dots x_n)$.

Каждую из половинок по лемме 11.1 на стр. 191 можем реализовать за $2^{\frac{n}{2}} + O\left(2^{\frac{n}{2}}\cdot\frac{n}{2}\right)$.

⇒ нам нужно $2^{\frac{n}{2}+1} + 2^{n/2}$, чтобы реализовать оба куска в каждой строчке.

За счёт того, что n может быть нечётное, и один кусок будет длиннее, нам потребуется $\le 3\cdot 2^{n/2}$ операций на реализацию обоих кусков.

Потом ещё в каждой строчке нужна одна конъюнкция для склейки двух кусков, и ещё $2^{n/2}$ внешних дизъюнкций. ⇒ всего $(3+1+1)2^{n/2} = 5\cdot 2^{n/2}$.

На самом деле, если делить не на два куска, а на три — то можно константу чуть-чуть уменьшить: нам надо $3\cdot 2^{n/3}$ операций на 3 куска, ещё по 2 конъюнкции на каждую строку, и ещё $2^{n/2}$ дизъюнкций

⇒ всего $3\cdot 2^{n/3} + 2\cdot 2^{n/2} + 2^{n/2} \sim 3\cdot 2^{n/2}$ операций.

Зам. Можно попробовать через операторы посчитать.

У нас много нулей и мало единиц среди значений ф-ции f.

Применим к f лин. оператор $L: \{0,1\}^n \to \{0,1\}^m$ (m — плохой вариант).

Он склеит некоторые нули и единицы.

определим на образе новую ф-цию g,

где $g(y) = \begin{cases} 0, & \text{если для } y \text{ нет прообраза } x, \text{ на котором } f=1 \\ 1, & \text{иначе.} \end{cases}$

Тогда если $g(y)=0$ – то $f(y)=0$. – и больше ничего узнать не надо для f на этом наборе уже знали.

если $g(y)=1$ – то непонятно, чему равно f, т.к. это могла быть единица, а могла и ноль с единицей склеиться.

Сколько пар склеилось?

Рассм. мн-во лин. операторов как вероятн. пр-во.

Тогда вер-ть того, что у двух лин. ф-ций совпадет вектор значения – равна $\frac{1}{2^m}$.

$\Rightarrow$ максимальное кол-во склеившихся пар $= \sum_{\text{пар}_2} \frac{1}{2^m} = \frac{\text{кол-во пар}}{2^m} = \frac{\|f_0\| \cdot \|f_1\|}{2^m}$.

$\Rightarrow \exists$ лин. ф-ция, которая склеивает $\le \frac{\|f_0\| \cdot \|f_1\|}{2^m}$ пар – т.к. иначе М.О $> \frac{\|f_0\| \cdot \|f_1\|}{2^m}$.

$\Rightarrow$ после 1-го применения g, размер "хорошего" мн-ва,

т.е. где $g=0$ – стал $2^n - N - \frac{\|f_0\| \cdot \|f_1\|}{2^m} = N$ (где N – кол-во единичных значений)

Возьмем $m := \log 2N$

$\Rightarrow$ |хорошие мн-ва| $= 2^n - N - \frac{\|f_0\| \cdot \|f_1\|}{2N} = 2^n - N - \frac{2^n - N}{2} = \frac{2^n - N}{2}$

т.е. после 1-го шага "плохое" мн-во уменьшилось в 2 раза.

Теперь к "плохому" мн-ву (которое образовалось после 1-го шага) применим новую g и новый лин. оператор L, который еще в 2 раза уменьшит плохое мн-во.

И так будем продолжать, пока плохое мн-во не закончится.

сложность g: $\frac{|D|}{\log|D|} = \frac{2^m}{\log(2^m)} = \frac{2^m}{m} = \frac{2N}{\log(2N)}$

сложность L: n^2 – это мало по сравнению с этим

$\Rightarrow$ сложность $= \frac{2N}{\log 2N} \cdot \underbrace{2^n}_{\frac{1}{2}2N=N} + \frac{2N}{\log 2N} \cdot \left(\frac{2^n - N}{2}\right) + \frac{2N}{\log 2N} \cdot \left(\frac{2^n - N}{4}\right) + \dots =$

$= \frac{2N^2}{\log 2N} + \frac{2N}{\log 2N} \cdot (2^n - N) \underbrace{\left(1 + \frac{1}{2} + \frac{1}{4} + \dots\right)}_{\le 2} = \frac{2N}{\log 2N}\left(N + \frac{2^n - N}{2}\right) = \frac{2N}{\log 2N}\left(\frac{N}{2} + 2^{n-1}\right)$

У нас $N = 2^{n/2}$

$\Rightarrow \frac{2N}{\log 2N}\left(\frac{N}{2} + 2^{n-1}\right) = \frac{2 \cdot 2^{n/2}}{\frac{n}{2}+1} \cdot \left(2^{n/2} + 2^{n-1}\right)$ – что-то слишком много получилось, больше, чем $2^{n/2}$.

(Я не знаю, почему так)

(2.) Алгоритм декодирования кода Рида–Малера.

<u>Решение:</u> $RM(n,k)$ – это столбцы-значения ф-ций от n перем. степени $\leq k$.

Мы выяснили, что $d = 2^{n-k}$

Мы знаем, что чтобы исправить x ошибок, нужно, чтобы мин расст. было $\geq 2x+1$. – ну потому что шары.

$\Rightarrow$ код $RM(n,k)$ исправляет $\left\lfloor \frac{2^{n-k}-1}{2} \right\rfloor = \left\lfloor 2^{n-k-1} - \frac{1}{2} \right\rfloor = 2^{n-k-1} - 1.$

Поэтому считаем, что произошло $2^{n-k-1}-1$ ошибок произошло. И мы хотим восстановить коэф. многочлена Жегалкина нашей ф-ции, при условии, что мы значения f знаем не точно, а только в сумме с произошедшими ошибками.

Т.е известен вектор значений ф-ции $f \oplus c$, где $вес(c) \leq 2^{n-k-1}-1$

Сначала рассм. способ определения коэф. при одночлене $x_1 \cdot \ldots \cdot x_k$ (пусть сначала ошибок нет). Для этого одночлена определим разбиение булева куба $\mathbb{B}^n$ на 2^{n-k} подмн-в B_i; $i = 0, 1 \ldots 2^{n-k}-1$, так, чтобы B_i состояло из таких наборов, у которых на последних $n-k$ местах стоят такие константы $\beta_{k+1} \ldots \beta_n$, что $(\beta_{k+1} \ldots \beta_n)$ = двоичное представление числа i: $i = \sum_{j=1}^{n-k} \beta_{k+j} 2^{n-k-j}$. Т.е $(n-k)$ наборов фикс, а k – бегают.

Ограничение исх. одночлена $x_1 \ldots x_k$ на любое B_i равно единице на единств. наборе $x_1 = 1 \ldots x_k = 1$.

$\Rightarrow \forall i \in \{0, 1 \ldots 2^{n-k}-1\}: \bigoplus_{x \in B_i} x_1 \cdot \ldots \cdot x_k = 1.$

А ограничения любого другого монома на мн-ва B_i, построенные для $x_1 \ldots x_k$, будут степени строго $< k$ (т.к придётся вместо ≥ 1 переменной подставить константу).

~~[illegible]~~

Поэтому у такого монома будет хотя бы одна фиктивная переменная, поэтому все наборы, отличающиеся только в этой переменной – дают одинаковое значение $\Rightarrow$ сумма значений $\vdots 2 \Rightarrow$ по mod 2 она $= 0$: $\bigoplus_{x \in B_i} x_{i_1} \cdot \ldots \cdot x_{i_m} = 0$.

$\Rightarrow$ сумма значений f, взятая на B_i – равна 1 только если одночлен $x_1 \ldots x_k$ входит в многочлен Жегалкина. $\Rightarrow$ есть 2^{n-k} нерав соотношений, любое из которых даёт ответ, есть ли моном $x_1 \ldots x_k$ в многочлене Жегалкина.

Теперь пусть ошибки есть.

$\forall i = \{0, 1 \ldots 2^{n-k}-1\}$: сравним $\bigoplus_{x \in B_i} (f(x) \oplus c(x))$ и $\bigoplus_{x \in B_i} f(x)$, они отличаются не более, чем на $\|c\|$ множествах B_i. Но $\|c\| < 2^{n-k-1}$ – то есть он < половины из всех значений.

$\Rightarrow$ каких значений $\bigoplus_{x \in B_i} (f \oplus c)x$ больше – тому и равен коэф. при $x_1 \ldots x_k$ в многочлене Жегалкина.

Т.е считаем все суммы $\bigoplus_{x \in B_i} (f \oplus c)(x)$ – и если нулей больше, чем единиц, то слагаемого $x_1 \ldots x_k$ в многочлене для f нет, а если единиц больше – то есть. Дальше повторяем построение мн-в B_i и голосование для всех мономов степени k.

После того, как для функции f определены коэф. многочлена Жегалкина при всех одночленах степени k, к функции $f \oplus c$ прибавим все одночлены степени k, коэф. при которых равны единице. В результате получим новую функцию $f' \oplus c$, где $\deg f' \le k-1$, а вес c по-прежнему не превосходит $2^{n-k-1}-1$. Теперь, используя вектор значений ф-ции $f' \oplus c$, описанным выше способом определим в многочлене Жегалкина ф-ции f все коэф. при одночленах степени $k-1$. И т.д., пока степень нуля не дойдет.

③ Загадали число $0 \le x < 15$. Можно задавать вопросы (сразу все), ответ да/нет. Один из ответов — может быть ошибочным.
Сколько вопросов и какие?

Решение: Если бы все ответы были правильные точно, то нам хватило бы 4 вопросов:
1) Правда ли, что 1-й бит (т.е. a_1) в двоичном разложении $=1$?
2) Правда ли, что 2-й бит (т.е. a_2) в двоичном разложении $=1$?
3) —"— 3-й бит (т.е. a_3) $=1$?
4) —"— 4-й бит (т.е. a_4) $=1$?

~~$a_4 a_3 a_2 a_1$~~ $X = a_1 a_2 a_3 a_4$

Но у нас один ответ может быть неправильным.
Нам нужна такая длина кода n, чтобы в куб 2^n влезло 16 шаров радиуса 1, причем шары не должны пересекаться — иначе мы не исправим ошибку. ~~причем нам надо покомпактнее шары упаковать,~~
$\Rightarrow 2^n \ge (n+1)\cdot 16$ — т.к. в шаре 1 вершина-центр + еще n граничат с ней по ребру.
равенство бывает, только если $n+1=2^m$; m — какое-то.
$n=7$ — подходит. — даст равенство — а нам и надо покомпактнее шары упаковать, чтобы n было поменьше — т.к. n — это число вопросов.
И это и есть код Хемминга! — это такой код, что булев куб $\mathbb{B}^n$ полностью покрывается объединением шаров, т.е. $n+1=2^m$
покажем, что $\forall m$ такой код $\exists$, а в частности, что код Хемминга такой.
Действительно, нам нужно расстояние $=3$ — чтобы исправить 1 ошибку.
А расстояние $= d \Leftrightarrow$ любые $d-1=2$ столбца в проверочной матрице — лин. незав.
(где $d=3$)
Значит, в проверочной матрице не должно быть нулевого столбца и одинаковых столбцов — вот и все требования $\Rightarrow H = m(\underbrace{\quad}_{2^m-1})$
$\Rightarrow$ мощность кода Хемминга $= 2^{n-m} \underset{n=2^m-1}{=} 2^{2^m-1-m}$

Тогда если образовать в каждом кодовом слове шар радиуса 1
$\Rightarrow$ всего покроется $2^{2^m-1-m}\underbrace{(n+1)}_{=2^m} = 2^{2^m-1} = 2^n$.

У нас $2^n = \underbrace{(n+1)}_{=2^m} \cdot \underbrace{16}_{=2^4}$

$n=7$ — решение — это длина кодового слова = кол-во вопросов.
$n+1=8=2^3=2^m \Rightarrow m=3$.
$\Rightarrow$ проверочная матрица — это H_3:

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

проверочная матрица – она "проверяет", что:

$$\begin{cases} y_4+y_5+y_6+y_7=0 \\ y_2+y_3+y_6+y_7=0 \\ y_1+y_3+y_5+y_7=0 \end{cases}$$

– т.к. $Hg=0$, по опр. проверочной матрицы.

Это — и есть наши 3 доп. вопроса.

То есть 1, 2 и 4 биты – это контрольные биты.

3, 5, 6, 7 – это изначальные наши 4 вопроса.

	a_1 1	a_2 2	a_3 3	a_4 4	5	6	7
1							
2							
4							

– это контрольные биты.

– контрольный бит кодирует в себя сумму этих элементов.

То проверочная матрица – она проверяет, что

$$\begin{cases} y_4 \overset{?}{=} y_5+y_6+y_7 \\ y_2 \overset{?}{=} y_3+y_6+y_7 \\ y_1 \overset{?}{=} y_3+y_5+y_7 \end{cases}$$

(Ну "+" и "–" в $\mathbb{Z}_2$ – это одно и то же)

⇒ 1-й вопрос – правда ли, что сумма 1-го, 2-го и 4-го битов = 1? (где $a_1+a_2+a_4 \overset{?}{=} 1$)

2-й вопрос – — // — $a_1+a_3+a_4$ равна единице?

3-й вопрос – правда ли, что $a_1=1$?

4-й вопрос – правда ли, что $a_2+a_3+a_4=1$?

5-й вопрос – правда ли, что $a_2=1$?

6-й вопрос – правда ли, что $a_3=1$?

7-й вопрос – правда ли, что $a_4=1$?

$X=a_1a_2a_3a_4$

А как теперь декодировать?

Ну получили вектор длины 7.

Посчитаем для него 1, 2 и 4 – контрольные биты – по самому этому вектору.

Если неправильный только один из них – то ~~ошибка произошла~~ нам неправильно сказали именно эту контрольную сумму – т.к. если бы нам неправильно сказали какой-то бит исх. числа $a_4a_3a_2a_1$ – то были бы неправильные ≥ 2 контрольные суммы – ну см. таблицу с данными ответственности.

⇒ ~~просто~~ просто отбрасываем 1, 2 и 4 биты – и вот наше число.

Если не совпало ≥ 2 контрольных сумм – это значит нам неправильно ответили на какой-то из 4-х вопросов, когда мы его/её спрашивали.

Сумма ~~1+2+4~~ – это и есть номер бита, где ошибка произошла.

номеров несовпавших контрольных бит

⇒ инвертируем бит на этом ошибочном месте, голом отбрасываем 1, 2 и 4 бит – и вот наше исх. число $X=a_4a_3a_2a_1$ в двоичной записи.

Пример 1001100 (где 1 = "да", 0 = "нет")

контр. сумма №1 = 1+0+1+0 = 0

контр. сумма №2 = (0+0)+(0+0) = 0

контр. сумма №4 = 1+1+0+0 = 0

⇒ все биты правильные

⇒ Вычеркиваем 1, 2 и 4 биты ⇒ $X=0100=4$.

(4) Для кода с длиной n: $d = \frac{n}{2}$ — оценка плоткина

Док-ть, что число слов в коде $\le 2n$.

Док-во: посчитаем сумму попарных расстояний $\sum_{x,y \in G} d(x,y)$ двумя способами.

Пусть M – мощность кода = кол-во векторов в коде

d – min расстояние между словами

n – длина слов.

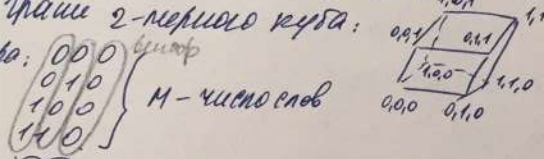
$\Rightarrow \sum_{x,y \in G} d(x,y) \ge \underbrace{\frac{1}{2} M(M-1)}_{\text{кол-во пар (неупорядоченных)}} \cdot d$ — расстояние между каждыми двумя векторами $\ge \frac{n}{2}$.

Теперь посчитаем вторым способом:

при фикс. координате $i \in [0; n]$: кол-во пар из 0 и 1 среди векторов, составленного из i-х координат всех векторов – даст свой вклад в эту сумму.

длины M

Ну на примере: пусть мы хотим сумму расстояний в худшей (плохой) трехмерного куба:



Это 4 вектора: 0 0 0 / 0 1 0 / 1 0 0 / 1 1 0 — M – число слов

вектор

n – длина слова.

Считаем по столбцам в 1-м столбце – четыре пары (0,1)

во 2-м столбце – четыре пары из нуля и единицы (неупорядоченных)

в 3-м столбце – кол-во пар (0,1)

в сумме: $4+4=8$

проверка: 1,0,0 — 1,1,0 (1); 0,0,0 — 0,1,0 (1); 1,0,0 — 0,0,0 (2); 1,1,0 — 0,1,0 (2)

сумма: $1+1+1+1+2+2=8$. – совпало

И здесь видно отличие линейных кодов (кода G – это подпр-во) от нелинейных.

В линейных: во всех столбцах будет ровно половина нулей и половина единиц – т.к. это подпр-во – и сумма будет $n \cdot \frac{M}{2} \cdot \frac{M}{2}$ (т.к. $\frac{M}{2}$ нулей и $\frac{M}{2}$ единиц в столбце)

А у нелинейных будет разное кол-во единиц в столбце, в зависимости от номера разряда. Пусть в i-м разряде he единиц

$\Rightarrow$ будем $he \cdot (M - he)$ пар в i-м столбце.

$$\Rightarrow \sum_{1 \le i < j \le M} d(x_i; x_j) = \sum_{1 \le i < j \le M} \sum_{i=1}^{n} |x_i^i - x_j^i| = \sum_{i=1}^{n} \underbrace{\sum_{1 \le i < j \le M} |x_i^i - x_j^i|}_{he \cdot (M-he)} = \sum_{i=1}^{n} \underbrace{he(M-he)}_{\text{парабола по } he.} \le \begin{cases} n \cdot \frac{M}{2} \cdot \frac{M}{2}, & \text{если } M \text{ – четно} \\ n \cdot \frac{M-1}{2} \cdot \frac{M+1}{2}, & \text{если } M \text{ – нечетно.} \end{cases}$$

Max достигается при $he = \frac{M}{2}$, если M четное, и $he = \frac{M \pm 1}{2}$, если M – нечетное

$$\Rightarrow \begin{matrix} n \cdot \frac{M}{2} \cdot \frac{M}{2}, \text{ если } M \text{ – четно} \\ n \cdot \frac{M-1}{2} \cdot \frac{M+1}{2}, \text{ если } M \text{ – нечет} \end{matrix} \ge \sum_{1 \le i < j \le M} d(x_i; x_j) \ge \frac{1}{2} \cdot M(M-1) d. \quad (*)$$

$\Rightarrow$ $\begin{cases} M \le \frac{2d}{2d-n}, & \text{если } M \text{ четно и } 2d > n. \\ M \le \frac{n}{2d-n} = \frac{2d}{2d-n} - 1, & \text{если } M \text{ – нечетно} \end{cases}$

Заметим, что нам бы хотелось подставить $d = \frac{n}{2}$, но это нельзя, т.к. в пункте сократится.

поэтому заметим, что $\max \#\text{слов}(n,d) \le 2 \cdot \max \#\text{слов}(n-1,d)$

↑ длина n с max расстоянием d

Действительно, разделим все кодовые слова из (n,d) на 2 класса: в один отнесем слова, которые начинаются с нуля, а в другой – слова, начинающиеся с единицы. тогда один из этих классов содержит по крайней мере половину кодовых слов, т.е. верно $\#(n-1,d) \ge \frac{\#(n,d)}{2}$

$\Rightarrow \#(n,d) \le 2 \cdot \#(n-1,d)$

$\Rightarrow \#(n, \frac{n}{2}) \le 2 \cdot \#(\underbrace{n-1}_{N}; \frac{n}{2}) \le 2 \cdot \left[\frac{2 \cdot \frac{n}{2}}{2 \cdot \frac{n}{2} - (n-1)}\right] = 2 \cdot \frac{n}{d \le N+1} = 2n$. что.

↑ здесь учли $2d = N > n-1$

Зам. такой код есть: $\Rightarrow$ применяем (*):

это код Адамара $M \le 2 \cdot \left\lfloor \frac{d}{2d-N} \right\rfloor$

2 способ. Заметим, что на языке единицы и расст. такое скалярное произв:

$\langle (x_1 \dots x_n); (y_1 \dots y_n) \rangle := \sum \frac{x_i y_i}{n}$.

тогда если кодовое расстояние было $> 50\%$ (разрешая ему быть очень близким к 50%), то все углы между кодовыми словами будут тупые, и число слов $\le n+1$.

А если еще разрешить и прямые углы (т.е $d = 50\%$), то число слов $\le 2n$. И оценка достижима: можно время $\pm$ базисные вектора – их как раз $2n$ штук.

Доказ: возьмем вектора x_k при $k > n$ через $x_1 \dots x_n$; $x_k = \lambda_1 x_1 + \dots + \lambda_n x_n$

1) тогда во всех таких выражениях коэф – отрицательно. т.к если есть отрицательные – все непустые – их перенесем в левую, и получим $x_k + (\underbrace{-\lambda_s}_{>0}) x_s + \dots = \underbrace{\lambda_t}_{>0} x_t + \dots$

тогда скал. произв. левой части на правую одновременно > 0 – как скалярный квадрат ненулевого вектора, и < 0 – т.к это сумма неположительных скалярных произведений с положит. коэф. против.

2) если общее число векторов $> 2n$, то вектора x_k при $k > n$ – лин. зависимы. т.к коэф. их нулевой или. комб не могут быть одного знака, т.к в выражениях через базисные вектора $x_1 \dots x_n$ коэф. отрицательны и сократься не могут

$\Rightarrow$ есть разные знаки $\Rightarrow$ разнесем в разные части

$\Rightarrow \underbrace{\mu_s}_{>0} x_s + \dots = \underbrace{\mu_t}_{>0} x_t + \dots$

В ее части содержат хотя бы один ненулевой член $\Rightarrow$ ненулевое. Опять, скал. произв. левой части на правую одновременно > 0 – как скал. квадрат – и < 0 – как сумма неполож. скал. произв. с полож. коэф.

(5.) **проверочная матрица** $H = \{E_k \; \tilde{H}\}_k$.

$\tilde{H} = k \times n-k$

G - линейный $(n, n-k)$-код.

найти порождающую матрицу G.

Решение: $a \to aG =: g$ - код.

$Hg = 0$ - условие проверочной матрицы.

$$Hg=0: \; k\left\{\begin{pmatrix} 1 & & & h_{11} & \dots & h_{1,n-k} \\ & \ddots & & & & \\ & & 1 & h_{k,1} & \dots & h_{k,n-k} \end{pmatrix}\right. \begin{pmatrix} g_1 \\ \vdots \\ g_k \\ g_{k+1} \\ g_n \end{pmatrix} = \begin{pmatrix} g_1 + h_{11} g_{k+1} + \dots + h_{1,n-k} g_n \\ \vdots \\ g_k + h_{k,1} g_{k+1} + \dots + h_{k,n-k} g_n \end{pmatrix} = \overline{0}$$

(первый блок — k столбцов, второй — $n-k$)

$\Rightarrow$ условие "проверки" состоит в том, что

$$\begin{cases} -g_1 = h_{11} g_{k+1} + \dots + h_{1,n-k} g_n \\ \vdots \\ -g_k = h_{k,1} g_{k+1} + \dots + h_{k,n-k} g_n \end{cases}$$

$\Rightarrow$ породить код с такими условиями легко: надо просто положить $g_1 \dots g_k$ равным этим лин. комбинациям, а $g_{k+1} \dots g_n$ — просто $a_1 \dots a_{n-k}$

$$(a_1 \dots a_{n-k}) \begin{pmatrix} h_{11} & & h_{k,1} & -1 & & 0 \\ \vdots & & \vdots & & \ddots & \\ h_{1,n-k} & & h_{k,n-k} & 0 & & -1 \end{pmatrix} = (a_1 h_{11} + \dots + a_{n-k} h_{1,n-k}; \dots ; a_1 h_{k,1} + \dots + a_{n-k} h_{k,n-k}; -a_1 \dots -a_{n-k})$$

(столбцы $k+1 \dots n$; матрица размера n — это G; $N^\circ k$, $N^\circ k+1$, $N^\circ n$)

$$\Rightarrow \boxed{G = \begin{pmatrix} \tilde{H}^T & \begin{matrix} -1 & & \\ & \ddots & \\ & & -1 \end{matrix} \end{pmatrix} \Big\} n-k}$$

($\tilde{H}^T$ — $n-k \times k$; блоки: k и $n-k$) ← ответ

Зам. на самом деле, в $\mathbb{F}_2$ "−" и "+" - это одно и то же, поэтому минусы можно убрать.