Теория кодов

Лектор: Завьялов Олег Геннадьевич
кандидат физико-математических наук, доцент

Теория кодов

1. Проблема передачи информации, которая должна быть понятна тому, для кого предназначена.

2. Взламывание кодов. Перехват и декодирование сообщений.

Введение

Раздел теории кодов, посвященный декодированию сообщений, называется ***криптографией***.

Код – представление множества символов строками, состоящими из 0 и 1. Это множество символов обычно включает буквы алфавита, цифры и определенные контрольные символы.

Коды представляются бинарными строками с целью использования из компьютерами для хранения и передачи данных.

Коды должны обладать (по возможности) некоторыми ***свойствами***. ***Наиболее важное свойство кода***: когда сообщение кодируется как двоичная строка, состоящая из конкатенации элементов кода, эта конкатенация однозначна.

Конкатенация – склеивание объектов линейной структуры.

Конкатенация - бинарная операция, определенная на словах данного алфавита.

Введение

При декодировании сообщения не должно возникать проблем с тем, какую букву представляет элемент кода.

Такой код называют ***декодируемым кодом***.

Способы достижения этой цели:

1) кодирование всех символов двоичными строками одной длины. Такой код называют ***блоковым***. Полезен при ограничении длины кода для каждого посланного символа или буквы.

2) ***префиксный*** код иначе ***мгновенный код***). Код *C* является префиксным, если элемент кода не может быть начальной строкой другого элемента кода. При чтении строки из единиц и нулей, изображающий символ, всегда известен момент завершения сроки.

Кома-код

Разновидность префиксного кода.

При его использовании каждый символ кодируется строкой из единиц, в конце которой находится 0.

Множество строк имеет вид {0, 10, 110, 1110, 11110, …}.

Недостаток: элементы кода могут быть очень длинными и занимать большой объем памяти.

Код минимизирующий время передачи данных и объем памяти

- код ***Хаффмана***.

- код ***Морзе***. Буквы разделяются пробелами, а слова тремя пробелами. Пробел – единица времени.

Код Морзе

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

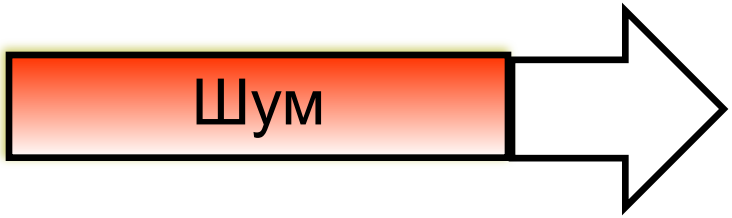
Код Морзе

Знаки кода Морзе	Буквы рус.	Буквы лат.	Знаки кода Морзе	Цифры	Знаки препинания и служебные сигналы
·—	А	Aa	····—	1	·—·—·— (.) запятая
—···	Б	Bb	···——	2	······ (.) точка
·——	В	Ww	····—	3	—·—·—· (:) точка с запятой
——·	Г	Gg	·····	4	———··· (:) двоеточие
—··	Д	Dd	·····	5	··——·· (?) вопросительный знак
·	Е	Ee	—····	6	·————· (№) номер
···—	Ж	Vv	—····	7	·—··—· (“”) кавычки
——··	З	Zz	———··	8	·————· (’) апостроф
··	И	Ii	————·	9	—·——·— () скобки
·———	Й	Jj	—————	0	——·—·— (!) восклицательный знак
—·—	К	Kk			—····— (–) тире
·—··	Л	Ll			—···· Ждать
——	М	Mm			····· Понял
—·	Н	Nn			—··—· (/) дробная черта
———	О	Oo			—·—·— Знак раздела
·——·	П	Pp			····· Перебой (исправление ошибки)
·—·	Р	Rr			·—·—· Сигнал о начале передачи (НП)
···	С	Ss			·—·—· Сигнал о готовности к приёму (ПО)
—	Т	Tt			·—·—·—·—·— Начало действия
··—	У	Uu			·—·—· Знак окончания передачи
··—·	Ф	Ff			
····	Х	Hh			
—·—·	Ц	Cc			
———	Ч	—			
————	Ш	—			
——·—	Щ	Qq			
—·——	Ы	Yy			
··——	Ю	—			
··—·	Я	—			
·—·—	Й	Jj			
—·——	Ь,Ъ	Xx			
··—··	Э	Ee			

Шум

В процессе передачи данных могут возникать ошибки.
Все, что может стать причиной ошибок, называется термином

“шум”.



Код, обладающий свойством определения наличия ошибок, называются ***кодами, обнаруживающими ошибки***.
Коды, позволяющие исправлять ошибки, называются ***кодами, исправляющими ошибки***.

Недостаток: должны включать информацию, менее эффективны в отношении минимизации памяти.

А если ошибка не одна, а возможны многие ошибки?

Код Грея

Имеется вращающийся диск, разделенный на секторы, и серия щеток или лазерных лучей, выделяющих информацию о скорости вращения диска.

Если двоичные строки, записывающие нумерацию соседних секторов, существенно различаются в в отдельных цифрах при переходе от одного сектора к следующему, тогда считывающее устройство воспримет это так, чтобы измененный сектор мог выбрать число, совершенно отличное от числа, выбранного любым из секторов.

Желательно пронумеровать секторы таким образом, чтобы двоичные строки, определяющие соседние секторы, различались только цифрой.

Код ASCII

- блоковый код, использует 7 битов.

Любой закодированный символ изображается строкой из семи символов 1 и 0. Восьмой бит добавляется таким образом, чтобы количество единиц было четным.

Если код переданной строки получен с единственной ошибкой, то количество единиц будет нечетным, получатель информации поймет, что произошла ошибка.

Если произойдет две ошибки, их нельзя обнаружить, т.к. количество единиц будет четным.

Код ASCII

Код ASCII с битом контроля четности

код	символ	код	символ	код	символ	код	символ
00000000	NUL	10100000	SP	11000000	@	01100000	‘
10000001	SOH	00100001	!	01000001	A	11100001	a
10000010	STX	00100010	"	01000010	B	11100010	b
00000011	ETX	10100011	#	11000011	C	01100011	c
10000100	EOT	00100100	$	01000100	D	11100100	d
00000101	ENQ	10100101	%	11000101	E	01100101	e
00000110	ACK	10100110	&	11000110	F	01100110	f
10000111	BEL	00100111	’	01000111	G	11100111	g
10001000	BS	00101000	(	01001000	H	11101000	h
00001001	HT	10101001	)	11001001	I	01101001	i
00001010	LF	10101010	*	11001010	J	01101010	j
10001011	VT	00101011	+	01001011	K	11101011	k
00001100	FF	10101100	’	11001100	L	01101100	l
10001101	CR	00101101	–	01001101	M	11101101	m
10001110	SO	00101110	.	01001110	N	11101110	n
00001111	SI	10101111	/	11001111	O	01101111	o
10010000	DLE	00110000	0	01010000	P	11110000	p
00010001	DC1	10110001	1	11010001	Q	01110001	q
00010010	DC2	10110010	2	11010010	R	01110010	r
10010011	DC3	00110011	3	01010011	S	11110011	s
00010100	DC4	10110100	4	11010100	T	01110100	t
10010101	NAK	00110101	5	01010101	U	11110101	u
10010110	SYN	00110110	6	01010110	V	11110110	v
00010111	ETB	10110111	7	11010111	W	01110111	w
00011000	CAN	10111000	8	11011000	X	01111000	x
10011001	EM	00111001	9	01011001	Y	11111001	y
10011010	SUB	00111010	:	01011010	Z	11111010	z
10011011	ESC	10111011	;	11011011	[	01111011	{
10011100	FS	00111100	<	01011100	\	11111100	|
00011101	GS	10111101	=	11011101	]	01111101	}
00011110	RS	10111110	>	11011110	^	01111110	~
10011111	US	00111111	?	01011111	_	11111111	DEL

Код ASCII

Пусть вероятность ошибки при передаче равна 0, 01 как для изменения 1 и 0, так и для изменения 0 и 1.

Пусть вероятность ошибки не зависит от расположения ошибки и от того, что является ошибкой: изменение 1 и 0 или наоборот.

Пусть появление одной ошибки не влияет на вероятность появления другой.

Поскольку 3 ошибки будут обнаружены, вероятность не обнаружить более двух ошибок меньше, чем вероятность наличия 4-х или более ошибок, т.к. любое нечетное количество ошибок будет обнаружено.

И эта вероятность << вероятности обнаружения одной или менее ошибок.

Порождающие матрицы

Предполагается, что все строки кода имеют фиксированную длину n, эти строки трактуют как векторы или матрицы $(1 \times n)$ –матрицы.

Сложение:

$$1 + 1 = 0$$

$$11110001 + 10100111 = 01010110$$

- ***линейные коды.***

Строки кода C – двоичные строки длины n, которые являются векторами или $(1 \times n)$ –матрицами.

Если B_n - множество всех двоичных строк длины n, то C – подмножество множества B_n.

Определение

Код *C* называется ***линейным***, если *C* – подгруппа группы B_n .

Закон дистрибутивности:

$$A\,(B + C) = AB + AC$$

для любых матриц *A*, *B*, *C*, произведение которых определено.

Если

$$u = (u_1, u_2, \ldots, u_n)$$

$$v = (v_1, v_2, \ldots v_n)$$

то скалярное произведение

$$uv = u_1v_1 + u_2v_2 + \ldots u_nv_n$$

Определение

Весом строки кода, обозначается wt(c), называется количество единиц в строке.

Например, если c = 1011010, то wt(c) = 4.

Пусть имеется ($k \times n$) – матрица $G = [I_k \mid A_{n-k}]$

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

G - называется ***порождающей матрицей***.

Строки порождающей матрицы – векторы или строки кода.

Множество строк обозначается S.

$$S = \{100101, 010110, 001011\}$$

Определение

Пусть C – код, образованный всеми векторами, которые являются конечными суммами строк из S.

C – подгруппа B_n.

Складывая две первые строки из S, 110011 будет принадлежать C.

Группа C порождена множеством S.

S является минимальным множеством, порождающим код C.

Обозначение – группа C порождена множеством S:

$$C = S^*$$

Код C вида $[I_k \mid A_{n-k}]$ (порожденный строками $[I_k \mid A_{n-k}]$) называется $[n, k]$ – кодом.

Теорема

[n, k] – код C содержит 2^k строк.

Пример

Если необходимо передать строки сообщения длины k, то кодируем их, умножая справа на матрицу C.

Если $w = (w_1, w_2, \ldots w_k)$

то кодируем это строкой wC.

Например, закодируем 110

$$(1,1,0)\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = (1,1,0,0,1,1)$$

или 110011.

В общем случае

Если $S = (s1, s2, \ldots, sk)$ – множество строк порождающей матрицы

$$w = (w_1, w_2, \ldots w_k)$$

то закодированная строка имеет вид

$$w_1 s_1 + w_2 s_2 + \ldots + w_k s_k$$

- сумма строк из S, т.к. каждое w_i равно либо 1, либо 0 принадлежит C, т.к. C – группа, порожденная множеством S.

Закодированная строка имеет вид

$$(w_1, w_2, w_3)\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = (w_1, w_2, w_3, w_1 + w_2, w_2 + w_3, w_1 + w_3)\,.$$

Четвертый бит строки должен быть равен $w_1 + w_2$.

Пятый бит должен быть равен $w_2 + w_3$.

Шестой бит должен быть равен $w_1 + w_3$.

Если закодированная строка имеет вид $(w_1, w_2, w_3, w_4, w_5, w_6)$,

То $w_4 = w_1 + w_2$, $w_5 = w_2 + w_3$, $w_6 = w_1 + w_3$.

Если любая закодированная строка после передачи не удовлетворяет этим соотношениям, то при передаче произошла ошибка.

Матрица A_{n-k} служит для контроля точности передачи данных.

В общем случае, если имеется закодированная строка

$w_1\ w_2\ w_3\ \ldots\ w_k\ \ldots\ w_n$ и

$$R = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & A_{1,k+1} & A_{1,k+2} & \cdots & A_{1,n} \\ 0 & 1 & 0 & \cdots & 0 & A_{2,k+1} & A_{2,k+2} & \cdots & A_{2,n} \\ 0 & 0 & 1 & \ddots & \vdots & A_{3,k+1} & A_{3,k+2} & \cdots & A_{3,n} \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & 0 & 1 & A_{k,k+1} & A_{k,k+2} & & A_{k,n} \end{bmatrix},$$

то для $i > k$ имеем $w_i = w_1 A_{1,i} + w_2 A_{2,i} + w_3 A_{3,i} + \cdots + w_k A_{k,i}$, и закодированная строка должна удовлетворять всем этим $n - k$ соотношениям.

Метод – использование лидеров смежных классов

Проблема – исправление ошибки, если известно, что произошла единственная ошибка.

Пример.

$$R = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Известно, что

$$S = \{100101, 010110, 001011\},$$

так что

$$C = \{000000, 100101, 010110, 001011, 110011, 011101, 101110, 111000\}.$$

Сформируем для C смежные классы в B_n .

Первым смежным классом является сам C.

Для формирования следующего смежного класса выберем элемент из B_n , который имеет минимальный вес и не принадлежит C.

Например, можно выбрать $b_1 = 100000$.

Смежный класс

$$b_1 + C = \{100000, 000101, 110110, 101011, 010011, 111101, 001110, 011000\}.$$

Опять выбираем элемент из B_n , который имеет минимальный вес и не ни одному из предыдущих смежных классов.

Например, можно выбрать $b_2 = 010000$.

Смежный класс

$$b_2 + C = \{010000, 110101, 000110, 011011, 100011, 001101, 111110, 101000\}.$$

Продолжая этот процесс, получится таблица для Bn , разделенного на смежные классы, где элемент с минимальным весом выписан первым.

Элементы первого столбца называются ***лидерами смежных классов***.

000000	100101	010110	001011	110011	011101	101110	111000
100000	000101	110110	101011	010011	111101	001110	011000
010000	110101	000110	011011	100011	001101	111110	101000
001000	101101	011110	000011	111011	010101	100110	110000
000100	100001	010010	001111	110111	011001	101010	111100
000010	100111	010100	001001	110001	011111	101100	111010
000001	100100	010111	001010	110010	011100	101111	111001
100010	000111	110100	101001	010001	111111	001100	011010

Определение

Вектор v называется ортогональным вектору w, если скалярное произведение $v \cdot w = 0$.

Пусть задан код C.

Двойственным к коду C, обозначается $C^{\perp}$, называется множество всех строк из B_n , ортогональных каждой строке из кода C .

Теорема.

$C^{\perp}$ - подгруппа группы B_n.

Теорема

Пусть C - групповой код, а $C^{\perp}$ - двойственный ему код. Строка t принадлежит коду $C^{\perp}$ тогда и только тогда, когда она ортогональна каждой строке из множества S, множества порождающих элементов кода C.

Прошлый пример . Матрица контроля честности

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = [I_3|A_3]\,.$$

Определим теперь

$$G^{\perp} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} = [A_3^t|I_3]\,,$$

где A^t_3 – транспозиция матрицы A_3 , полученная заменой строк матрицы A_3 на столбцы.

Матрица $G^{\perp}$ называется ***матрицей контроля честности***.

Перебирая все возможные варианты, можно показать, что скалярное произведение любой строки из G с любой строкой из $G^{\perp}$ равно 0. Отсюда получаем, что

$$G^{\perp} r_i^t = 0,$$

где r_i^t — транспозиция i-ой строки матрицы G. По определению транспозиции r_i^t является i-ой строкой матрицы G, преобразованной в столбец. Мы превращаем строку в столбец, чтобы на него можно было умножить матрицу $G^{\perp}$:

$$\begin{aligned} G^{\perp}(w_1 r_1^t + w_2 r_2^t + w_3 r_3^t) &= w_1 G^{\perp} r_1^t + w_2 G^{\perp} r_2^t + w_3 G^{\perp} r_3^t = \\ &= 0 + 0 + 0 = \\ &= 0 . \end{aligned}$$

Таким образом, если умножить матрицу $G^{\perp}$ на транспозицию любого элемента из C, то получим 0.

В общем случае, если

$$G = [I_k | A_{n-k}] = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & A_{1,k+1} & A_{1,k+2} & \cdots & A_{1,n} \\ 0 & 1 & 0 & \cdots & 0 & A_{2,k+1} & A_{2,k+2} & \cdots & A_{2,n} \\ 0 & 0 & 1 & \ddots & \vdots & A_{3,k+1} & A_{3,k+2} & \cdots & A_{3,n} \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & 0 & 1 & A_{k,k+1} & A_{k,k+2} & & A_{k,n} \end{bmatrix},$$

то

$$G^{\perp} = [A^t_{n-k} | I_{n-k}] = \begin{bmatrix} A_{1,k+1} & A_{2,k+1} & \cdots & A_{k,k+1} & 1 & 0 & 0 & \cdots & 0 \\ A_{1,k+2} & A_{2,k+2} & \cdots & A_{k,k+2} & 0 & 1 & 0 & \cdots & 0 \\ A_{1,k+3} & A_{2,k+3} & \cdots & A_{k,k+3} & 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & & \vdots & \vdots & \vdots & \ddots & 0 \\ A_{1,n} & A_{2,n} & & A_{k,n} & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Скалярное произведение i-ой строки матрицы G на j-ю строку матрицы $G^{\perp}$ равно

$$0+0+\cdots 0+A_{i,j}+0+\cdots 0+A_{i,j}+\cdots+0+0=0\,,$$

так что в общем случае

$$G^{\perp}r_i^t=0,$$

где r_i^t — транспозиция i-ой строки матрицы R. Если умножить матрицу $G^{\perp}$ на транспозицию любого элемента из C, то, используя те же самые рассуждения, имеем в результате 0. Мы также получаем еще один замечательный результат. Если два элемента b_1 и b_2 из B_n принадлежат одному и тому же смежному классу, образованному в B_n с использованием группы C, как это было сделано выше, то

$$G^{\perp}b_1^t=G^{\perp}b_2^t\,.$$

Доказательство:

Для доказательства воспользуемся тем фактом, что если b_1 и b_2 принадлежат одному смежному классу, то $b_1 = b_2 + c$ для некоторого $c \in C$. Следовательно, $b_1^t = b_2^t + c^t$ и

$$\begin{aligned} G^{\perp} b_1^t &= G^{\perp}(b_2^t + c^t) = \\ &= G^{\perp} b_2^t + G^{\perp} c^t = \\ &= G^{\perp} b_2^t + 0 = \\ &= G^{\perp} b_2^t \ , \end{aligned}$$

т.к.

$$G^{\perp} c^t = 0$$

для всех $c \in C$.

Определение. Синдром

Поскольку $G^{\perp}b^t$ одно и то же для всех b из класса смежности, то можно выбрать любое b из класса смежности и определить это значение. Таким образом, в каждую строку приведенной выше таблицы можно добавить это общее значение образа $G^{\perp}$, т.к. элементы каждой строки определяют класс смежности. Мы выбираем лидера смежного класса, потому что он простейший, и помещаем значение его образа во второй столбец. Эти значения называются ***синдромами***.

Уже известно, что первый синдром есть $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$.

Находим, что

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

поэтому $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$ — второй синдром, и

поэтому $\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$ — второй синдром, и

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

поэтому $\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$ — третий синдром.

Продолжая процесс, получаем следующую таблицу.

000000	000	100101	010110	001011	110011	011101	101110	111000
100000	101	000101	110110	101011	010011	111101	001110	011000
010000	110	110101	000110	011011	100011	001101	111110	101000
001000	011	101101	011110	000011	111011	010101	100110	110000
000100	100	100001	010010	001111	110111	011001	101010	111100
000010	010	100111	010100	001001	110001	011111	101100	111010
000001	001	100100	010111	001010	110010	011100	101111	111001
100010	111	000111	110100	101001	010001	111111	001100	011010

Заполнив таблицу, предположим, что получена переданная строка 101100. Умножение матрицы $G^{\perp}$ на транспозицию строки дает

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

Отсюда следует, что 101100 находится в строке 6. Лидер смежного класса — 000010, элемент крайнего левого столбца строки, содержащей 101100. Элемент C, расположенный в верхней строке столбца, содержащего 101100, равен 101110. Согласно способу построения таблицы имеем $101100 = 101110 + 000010$, поэтому можем предположить, что переданная строка 101100 должна иметь вид 101110, и в пятом бите была ошибка.

Этот метод намного быстрее, поскольку требует только умножить матрицу $G^{\perp}$ на транспозицию строки-сообщения, найти строку, содержащую синдром,

и найти переданное сообщение. Лидер для этого сообщения — индикатор ошибки, а элемент кода C, расположенный в первой строке столбца, содержащего переданное сообщение, есть исправленное сообщение.

Заметим, однако, что процесс можно сделать еще быстрее. При этом потребуются только два первых столбца приведенной выше таблицы. Предположим, что получено сообщение 110000. Умножение матрицы $G^{\perp}$ на его транспозицию дает

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

поэтому синдром — $\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$, а лидер смежного класса — 001000. Поскольку лидер указывает, что ошибка появилась в третьем бите, то, добавляя 001000 к 110000, получаем 111000, исправленный код. Таким образом, этот метод прост. Умножить транспозицию сообщения на матрицу $G^{\perp}$, чтобы найти синдром. Для этого нужно найти лидера смежного класса и прибавить его к сообщению, чтобы получить исправленный код. Обратите внимание, что используются только первые два столбца таблицы. Однако, теперь возникла еще одна проблема. Если полученным сообщением будет строка 101001, то синдром будет $\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$, и в соответствующей строке будет три элемента с весом 2. Вспомним, что строка 100010 была выбрана произвольно. Любая из таких строк равновероятно может содержать ошибку, поэтому в данном случае совершенно безнадежно использовать синдромы для исправления ошибки. Кроме того, мы пытаемся в данном случае исправить строку с двумя ошибками вместо одной.

Последний слайд лекции

!!.