Тема 9. Коды, корректирующие ошибки

Теоретическая часть

Введение

Когда по каналу связи передаётся сообщение, то возможна ситуация, когда помехи повлияют на сообщение и изменят его. В связи с этим возникают задачи: (а) уметь определять, было ли изменение сообщения или нет; (б) уметь определять, что именно было изменено в сообщении, и исправить изменения, сделанные помехами.

Простейший пример, с помощью которого можно определить, было ли изменено сообщение – *коды с проверкой на чётность*. Представим, что сообщение состоит из последовательностей нулей и единиц одинаковой длины. Данные последовательности назовём *информационными блоками*. В конец каждого информационного блока допишем один бит так, чтобы в блоке было чётное количество единиц. Так, сообщение

0001 1100 1010 1110 1111

надо модифицировать следующим образом:

00011 11000 10100 11101 11110

Если в каком-то информационном блоке один бит будет изменён (единица станет нулём или нуль превратится в единицу), то количество единиц изменится, и мы будем знать, что в данном информационном блоке есть ошибка. Однако определить, какой бит был изменён, не представляется возможным.

Принципы проверки

Для того, чтобы обеспечить контроль и коррекцию ошибок, в двоичный код необходимо добавить проверочные разряды.

Пусть m – количество разрядов информационного блока. Количество k проверочных разрядов должно удовлетворять условию

$$2^k \geq k + m + 1$$

m	Минимальное k
1	2
2 – 4	3
5 – 11	4
12 – 26	5
25 – 57	6

Код, в который добавлены проверочные разряды, и в которых информационные и корректирующие символы расположены по строго определенной системе и всегда занимают строго определенные места в кодовых комбинациях, называется *систематическим*.

Сравнение несистематического и систематического кода:

- пример несистематического кода:

m_1	m_2	m_3	m_4	k_1	k_2	k_3

- пример систематического кода:

m_1	k_1	m_2	k_2	m_3	k_3	m_4

Пусть $n = m + k$, тогда k/n – *избыточность кода.*

Маркировка кода с n символами, из которых m информационных символов: (n, m).

Код Хэмминга

Код Хэмминга исправляет **одиночную** ошибку. Он состоит из комбинации из m информационных и k проверочных символов.

Избыточная часть кода строится таким образом, чтобы при декодировании можно было установить не только наличие ошибки, но и ее расположение внутри кодовой комбинации. Достигается это путем многократной проверки принятой кодовой комбинации на четность. При этом число проверок всегда равно числу контрольных разрядов k.

При каждой проверке охватывается часть информационных символов и один из контрольных разрядов, в ходе проверке получают один проверочный символ. Если результат проверки дает четное число, то проверочному символу присваивается значение «0», в противном случае «1». В результате проверок получается k-разрядное двоичное число, указывающее номер позиции искаженного символа. Если в результате проверки получена комбинация из k нулей, значит ошибок в кодовой комбинации не обнаружено.

Маркировка кода Хэмминга $(n, m) = (2^k - 1, 2^k - 1 - k)$

k	(n, m)
3	(7, 4)
4	(15, 11)
5	(31, 26)
6	(63, 57)
7	(127, 120)

Кодирование

1. Выбираем k, вычисляем n и m.
2. Определяем позиции информационных разрядов и проверочных разрядов. Проверочные разряды должны занимать позиции, номера которых равны степени двойки. В таблице ниже зелёные разряды – проверочные, жёлтые – информационные.

k	(n, m)	Структура разрядов
3	(7, 4)	1 2 3 4 5 6 7
4	(15, 11)	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
5	(31, 26)	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

3. Вычисление контрольных бит (**первый способ**). Каждый контролирующий бит с номером N контролирует все последующие N бит через каждые N бит, начиная с позиции N

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
×		×		×		×		×		×		×		×		×		×		×		×		×		×		×		×
	×	×			×	×			×	×			×	×			×	×			×	×			×	×			×	×
			×	×	×	×					×	×	×	×					×	×	×	×					×	×	×	×
							×	×	×	×	×	×	×	×									×	×	×	×	×	×	×	×
															×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×

В таблице: первая строка – номера разрядов; остальные строки показывают области контроля проверочных битов: крестиками показаны разряды, контролируемые проверочным битом, стоящим на первом, втором, четвёртом, восьмом, шестнадцатом местах.

Чтобы вычислить значение проверочного бита, необходимо просуммировать разряды, подконтрольные этому биту. Если сумма получилась чётной, то записать в проверочный бит 0, если сумма получилась нечётной, то записать единицу.

4. Вычисление контрольных бит (**второй способ**). Строится матрица $M(2^k - 1, k)$ из $2^k - 1$ строк и k столбцов. В i-ой строке стоят цифры двоичного представления числа i. Приведём пример для $k = 3$ и $k = 4$.

$$M(7,3) = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}, \qquad M(15,4) = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Пусть сообщение, составленное из информационных и проверочных битов, имеет вид $s = (s_1, s_2, \dots, s_n)$. Составим систему уравнений $s \cdot M = 0$. В этой системе k уравнений.

ВНИМАНИЕ! Сложение в системе – это сложение по модулю 2.

Так, для случая $(7, 3)$ система имеет вид:

$$\begin{cases} s_4 \oplus s_5 \oplus s_6 \oplus s_7 = 0 \\ s_2 \oplus s_3 \oplus s_6 \oplus s_7 = 0 \\ s_1 \oplus s_3 \oplus s_5 \oplus s_7 = 0 \end{cases}$$

В эту систему вместо переменных, соответствующих информационным битам, необходимо подставить значения информационных битов, и затем подобрать значение переменной, соответствующую проверочному биту:

$$\begin{cases} s_4 = s_5 \oplus s_6 \oplus s_7 \\ s_2 = s_3 \oplus s_6 \oplus s_7 \\ s_1 = s_3 \oplus s_5 \oplus s_7 \end{cases}$$

Декодирование

1. **ВНИМАНИЕ! Код Хэмминга может исправить только одну ошибку!**
2. Проверка на корректность сообщения (**первый способ**). Вычисляем проверочные биты, как показано в пункте (3) кодирования. Сравниваем их с проверочными битами сообщения. Если все значения проверочных бит совпали, то сообщение корректно. В противном случае суммируем порядковые номера тех бит, которые не совпали, и получаем **номер** того бита, который был искажён.

3. Проверка на корректность сообщения (**второй способ**). Составляем произведение $s \cdot M$ и подставляем в него биты принятого сообщения. Если при вычислении произведения получается ноль-вектор, то сообщение переслано корректно. В противном случае мы получим вектор, который представляет собой двоичное представление **номера** позиции искажённого бита.
4. Если какой-то бит был искажён, то его необходимо инвертировать.
5. Удалив все проверочные биты, получаем исходное сообщение.

Практическая часть

Составить компьютерную программу (на любом языке программирования), которая реализует:
А) кодирование по методу Хэмминга;
Б) проверку на обнаружение ошибок.

Входные данные: алфавит и сообщение, заданное буквами этого алфавита.

Что должна делать программа.
1) По заданному алфавиту программа подбирает соответствующую длину блока кода, которым будет кодироваться один символ алфавита. Так, если алфавит состоит из 33 символов, то минимальное количество двоичных разрядов для его кодирования равно 6. Следовательно, нужно подобрать такой код, маркировка которого содержит количество разрядов информационных символов, не меньшее, чем требуемое нами количество разрядов. Отсюда следует, что нужно выбрать код с маркировкой (15, 11). Если алфавит состоит из 3000 символов, то количество информационных разрядов должно быть не менее 12, отсюда следует, что нужно выбрать код с маркировкой (31, 26).

2) Сопоставьте каждой позиции буквы в алфавите его двоичный код. При необходимости дополните его ведущими нулями, чтобы использовать его в кодировании Хэмминга.

3) Закодируйте сообщение кодом Хэмминга. На выходе нужно получить список из последовательностей, состоящих из нулей и единиц (каждая последовательность кодирует один символ).

4) Предусмотрите процедуру или функцию, с помощью которой можно в произвольном элементе полученного списка инвертировать один разряд (позицию разряда можно задавать вручную, либо генерировать случайно). С помощью этой процедуры или функции «испортите» сообщение.

5) Найдите с помощью кода Хэмминга «испорченные» последовательности, исправьте их, раскодируйте сообщение и выведите на экран раскодированное сообщение и информацию о том, в каких последовательностях и на каком месте были нарушенные разряды.