Оглавление

Глава 1

Циклические коды

Циклические коды – подкласс линейных кодов. Эти коды обладают тем замечательным свойством, что все циклические сдвиги любого кодового слова принадлежат коду. В частности, порождающая матрица может быть составлена из сдвигов одного слова. Таким образом, для хранения полной информации о коде длины n достаточно n бит. Трудно было бы представить, что коды такого узкого подкласса также хороши, как и линейные коды общего вида. Действительно, асимптотически эти коды плохи, но на удивление большая часть известных хороших кодов конечных длин – циклические коды. Больше того, циклическими являются и коды Рида-Соломона, которые сегодня являются безусловными рекордсменами по числу реализованных в реальных устройствах кодеров и декодеров, поскольку они применяются во всех накопителях на жестких дисках, в устройствах записи информации на CD и DVD и т.п. Основное достоинство циклических кодов – не простота описания, а возможность построения простых схем кодирования и декодирования. Об этом пойдет речь позже после изучения специальных классов циклических кодов – БЧХ-кодов и кодов Рида-Соломона.

1.1 Порождающий и проверочный полиномы циклического кода

Начнем прямо с определения.

Определение 1.1. *Линейный (n, k)-код над полем F_q называется циклическим, если циклический сдвиг любого слова также является кодовым словом.*

Применительно к циклическим кодам удобно использовать представление двоичных последовательностей длины n в виде полиномов степени не выше $n-1$.

Пример 1.1. *Последовательностям 10110, 10001, 00100 соответствуют полиномы* $1+x^2+x^3$, $1+x^4$, x^2.

В этом примере рассмотрены полиномы с коэффициентами из поля F_2. Точно также записываются полиномы с коэффициентами из произвольного конечного поля F_q. До определенного момента нас будут интересовать, в основном, двоичные коды. Тем не менее, по мере изложения мы все чаще будем опираться на свойства групп, колец и полей. Самые необходимые определения даны в приложении к данной главе.

Пусть имеется полином

$$f(x) = f_0 + f_1 x + ... + f_{n-1} x^{n-1}$$

с коэффициентами из поля F_q, причем значения коэффициентов могут быть, в том числе, и нулевыми. Умножим этот полином на x. Получим новый полином

$$xf(x) = 0 + f_0 x + ... + f_{n-2} x^{n-1} + f_{n-1} x^n.$$

Видно, что последовательность коэффициентов полинома $xf(x)$ получается сдвигом вправо последовательности коэффициентов полинома $f(x)$. Если мы теперь приведем правую часть последнего соотношения по модулю $x^n - 1$, т.е. примем, что $x^n = 1$, то получим

$$xf(x) = f_{n-1} + f_0 x + ... + f_{n-2} x^{n-1} + f_{n-1} x^n \mod (x^n - 1).$$

Таким образом, умножение $f(x)$ на x по модулю $x^n - 1$ дает циклический сдвиг $f(x)$.

Упражнение 1. *Докажите, что множество вычетов всех полиномов по модулю $x^n - 1$ образует кольцо.*

Мы называем полином *приведенным*, если коэффициент при его старшей степени равен единице.

С этого момента мы не различаем последовательностей и полиномов. Для нас теперь (n, k)-код $C = \{c(x)\}$ – это линейное подпространство размерности k в пространстве полиномов $c(x)$ степени не выше $n - 1$. Мы уже убедились в том, что представление кодовых слов в виде полиномов удобно хотя бы потому, что циклические сдвиги описываются умножением на мономы.

Теорема 1.1. *Пусть имеется циклический (n, k)-код. Если $g(x)$– кодовое слово, то и $m(x)g(x) \mod x^n - 1$ тоже будет кодовым словом для любого $m(x)$.*

Упражнение 2. *Докажите теорему 2.1.*

Теорема 1.2. *Пусть имеется циклический (n, k)-код. Среди слов кода существует только один приведенный полином $g(x)$ наименьшей степени h.*

Доказательство. Если бы нашлось два различных полинома степени h, то их сумма оказалась бы полиномом степени меньше h, что невозможно по условию теоремы. □

Теорема 1.3. *Пусть имеется циклический (n, k)-код и $g(x)$ – кодовое наименьшей степени h. Тогда любое слово кода делится на $g(x)$.*

Доказательство. Поскольку степень любого слова не меньше степени $g(x)$, запишем произвольное слово $c(x)$ в виде

$$c(x) = g(x)q(x) + r(x), \tag{1.1}$$

где $q(x)$ и $r(x)$ – частное и остаток от деления $c(x)$ на $g(x)$. Из этого представления по теореме 2.1 следует, что $r(x)$ – кодовое слово. Но степень $r(x)$ строго меньше степени $g(x)$, что невозможно по теореме 2.2. Следовательно, $r(x)$ должен быть равен нулю, а кодовое слово $c(x)$, согласно (2.1), кратно $g(x)$. □

Следствие 1. *Пусть имеется циклический (n,k)-код и $g(x)$ – кодовое наименьшей степени h. Тогда $g(x)$ является делителем x^n-1.*

Упражнение 3. *Докажите следствие 1 из теоремы 1.3. Подсказка: запишите x^n-1 через частное и остаток от деления на $g(x)$ аналогично (2.1). Поскольку все выражение равно нулю по модулю x^n-1, остаток должен делиться на $g(x)$, либо он тоже должен быть равен нулю.*

Теорема 1.4. *Пусть имеется циклический (n,k)-код и $g(x)$ – кодовое слово наименьшей степени h. Тогда $h=n-k$.*

Доказательство. Из теоремы 1.3 вытекает, что все слова кода кратны $g(x)$. Для доказательства надо подсчитать размерность множества полиномов кратных $g(x)$. Полиномы $g(x)$, $xg(x),\ldots$, $x^{n-h-1}g(x)$ линейно независимы и являются базисом линейного пространства размерности $n-h$. В виде их линейной комбинации можно получить любое слово вида $m(x)g(x) \bmod x^n-1$, т.е. это множество линейных комбинаций порождает весь код и его размерность равна размерности кода, т.е. $k=n-h$, что и требовалось доказать. □

Определение 1.2. *Полином $g(x)$, фигурирующий в теоремах 2.1–1.4, называется порождающим полиномом циклического кода.*

Пример 1.2. *Пусть $n=7$ и $g(x)=1+x^2+x^3$. Этот порождающий полином согласно теореме 1.4 порождает циклический код размерности 4. Выписывая базис $g(x)$, $xg(x),\ldots$, $x^{n-h-1}g(x)$ в виде двоичных последовательностей, получаем порождающую матрицу*

$$G=\begin{pmatrix}1&0&1&1&0&0&0\\0&1&0&1&1&0&0\\0&0&1&0&1&1&0\\0&0&0&1&0&1&1\end{pmatrix}.$$

Нетрудно подсчитать, что минимальное расстояние кода равно 3, что означает, что мы имеем циклическое представление кода Хэмминга.

Предыдущий пример подсказывает, что в общем случае порождающая матрица легко выписывается через коэффициенты порождающего полинома:

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_r & & & \\ & g_0 & g_1 & \cdots & g_{r-1} & g_r & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & & g_0 & \cdots & g_{r-1} & g_r \end{pmatrix}, \tag{1.2}$$

где пустые позиции заполняются нулями.

Следуя логике рассмотрения линейных кодов, нам предстоит найти проверочную матрицу, точнее, ее представление с помощью полинома или полиномов.

Рассмотрим полином

$$h(x) = \frac{x^n - 1}{g(x)}. \tag{1.3}$$

Согласно следствию 1, деление в правой части выполняется без остатка. Из этого равенства имеем $h(x)g(x) = 0 \mod x^n - 1$, более того, для любого кодового слова $c(x) = m(x)g(x)$ имеем

$$c(x)h(x) = 0 \bmod x^n - 1. \tag{1.4}$$

Определение 1.3. *Полином $h(x)$, определенный соотношением (2.3), называется* проверочным полиномом *циклического кода длины n, заданного порождающим полиномом $g(x)$.*

Степень проверочного полинома равна размерности кода $k = n - r$.

Чтобы выписать проверочную матрицу кода, перепишем (2.4) через коэффициенты полиномов:

$$\begin{aligned} &g_0 h_0 = 0 \\ &g_0 h_1 + g_1 h_0 = 0 \\ &- - - - - - - - - - \\ &g_r h_k = 0 \end{aligned}$$

Из этой записи нетрудно составить проверочную матрицу в виде

$$H = \begin{pmatrix} h_k & h_{k-1} & h_{k-2} & \cdots & h_0 & & & & \\ & h_k & h_{k-1} & \cdots & h_1 & h_0 & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & & & h_k & \cdots & h_1 & h_0 \end{pmatrix}. \qquad (1.5)$$

Заметим, что в строках матрицы H коэффициенты проверочного полинома следуют в инверсном порядке. Полином, инверсный полиному $h(x)$ можно записать в форме $x^k h(x^{-1})$. Итак, доказано следующее утверждение.

Теорема 1.5. *Дуальный код циклического (n,k)-кода с порождающим многочленом $g(x)$ и проверочным многочленом $h(x)$ является циклическим $(n, n-k)$ кодом с порождающим многочленом*

$$g^{\perp}(x) = x^k h(x^{-1}).$$

Из теоремы следует, в частности, что проверочный полином может рассматриваться как порождающий многочлен кода эквивалентного дуальному коду. Его кодовые слова могут быть получены инверсией (переписыванием в обратном порядке) слов дуального кода.

Упражнение 4. *Найдите порождающий многочлен кода дуального коду Хэмминга (7,4) (см. пример 1.2).*

Упражнение 5. *Что можно сказать о сложности декодирования циклического кода с помощью решетки? (См. задачу 8.)*

1.2 Примеры циклических кодов

Построить циклический код, как мы знаем из предыдущего параграфа, несложно. Для этого нужно разложить на множители многочлен $x^n - 1$ и выбрать любой его делитель в качестве порождающего многочлена циклического кода длины n. Чем больше степень делителя, тем меньше скорость, и, возможно, больше минимальное расстояние кода.

Пример 1.3. *Положим* $n = 7$. *Имеем*

$$x^7 + 1 = (1 + x)(1 + x + x^3)(1 + x^2 + x^3).$$

Отсюда следует, что множество циклических кодов длины 7 включает в себя

1. *(7,6)-код с* $g(x) = 1 + x$,
2. *(7,4)-коды с* $g(x) = 1 + x + x^3$ *и* $g(x) = 1 + x^2 + x^3$,
3. *(7,1)-код с* $g(x) = (1 + x + x^3)(1 + x^2 + x^3) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6$.
4. *(7,3)-код с* $g(x) = (1 + x)(1 + x^2 + x^3) = 1 + x + x^2 + x^4$,

$$g(x) = (1 + x)(1 + x + x^3) = 1 + x^2 + x^3 + x^4.$$

Коды 1 и 3 – пара дуальных кодов – код с проверкой на четность и код с повторениями. Два кода 2 – коды Хэмминга, отличающиеся порядком следования символов: один из них получается переписыванием кодовых символов другого кода в обратном порядке. Коды 4 – два кода Хэмминга с дополнительной проверкой на четность, т.е. симплексные коды, дуальные по отношению к кодам Хэмминга.

В этом примере рассматриваются коды над полем характеристики 2 поэтому знаки + и – не различаются.

Поскольку $x^n - 1$ *имеет корнем 1, то многочлен* $x + 1$ *всегда его делит. Поэтому при любом* n *можно выбрать* $g(x) = x + 1$ *в качестве порождающего полинома. При любом* $m(x)$ *произведение* $c(x) = m(x)g(x)$ *имеет четное число слагаемых, т.е. получаемый код – это* $(n, n-1)$*-код с проверкой на четность.*

Точно также двойственный к нему код – код с повторениями – тоже циклический код при всех n.

Пример 1.4 (Двоичные коды Хэмминга)**.** *Пусть* $n = 2^m - 1$. *В качестве порождающего многочлена* $g(x)$ *выберем любой примитивный многочлен* $p(x)$ *поля* $GF(2^m)$ *(см. Приложение). Тогда число избыточных символов будет равно* m, *размерность кода*

$k = 2^m - m - 1$. *Осталось убедиться в том, что минимальное расстояние кода равно 3. Предположим, что среди кодовых слов нашлось слово веса 2 кратное* $g(x) = p(x)$. *Слово веса 2 имеет вид* $c(x) = x^i + x^j = x^i(x^{j-1}+1)$, $j > i$. *В этом случае* $p(x)$ *оказывается делителем многочлена вида* $x^h - 1$ *при* $h < m$, *т.е. порядок корней многочлена меньше* m, *что невозможно в силу предположения о минимальность* $p(x)$. *Следовательно, в коде нет слов веса меньше 3 и мы имеем* ($n = 2^m - 1$, $k = 2^m - m - 1$)*-код Хэмминга.*

Описание кода проверочной матрицей имеет вид

$$H = \begin{pmatrix} 1 & \alpha & \ldots & \alpha^{2^m-3} & \alpha^{2^m-2} \end{pmatrix},$$

где α *– примитивный элемент поля. Подразумевается, что вместо элементов поля выписываются в виде столбцов их двоичные представления. Двоичная последовательность при умножении на* H *даст ноль только в том случае, если* α *является корнем соответствующего многочлена, т.е. только если этот многочлен делится на примитивный многочлен.*

Пример 1.5 (Двоичные коды максимальной длины)**.** *Более точно назвать эти коды кодами из последовательностей максимальной длины. Их применение много шире рамок теории кодирования. Они используются для установления синхронизации, в криптографии, для генерации псевдослучайных последовательностей и т.д.*

Циклический код длины $n = 2^m - 1$, *проверочным многочленом которого* $h(x)$ *является примитивный многочлен* $p(x)$ *поля* $GF(2^m)$, *называется кодом максимальной длины.*

По определению, код максимальной длины дуален коду Хэмминга. Порождающий многочлен кода равен $g(x) = (x^{2^m-1} - 1)/h(x)$. *Кодирование, как и для любого циклического кода, можно выполнить умножением информационного полинома* $m(x)$ *на* $g(x)$.

На рис. 1.1 показана одна из наиболее распространенных блок-схем генератора последовательности максимальной длины. Эта же схема может служить кодером для кода максимальной длины. В момент начала работы кодера в ячейки его регистра сдвига записывается начальная последовательность (информационные символы). На каждом из n тактов работы схемы на выход поступает

некоторый кодовый символ, содержимое регистра сдвигается вправо (умножается на x). Как только на выходе появляется ненулевое значение, оно умножается на $p(x)$ и вычитается из содержимого регистра, тем самым производится приведение по модулю $p(x)$. Выходом схемы является последовательность максимальной длины. Поясним работу схемы примером.

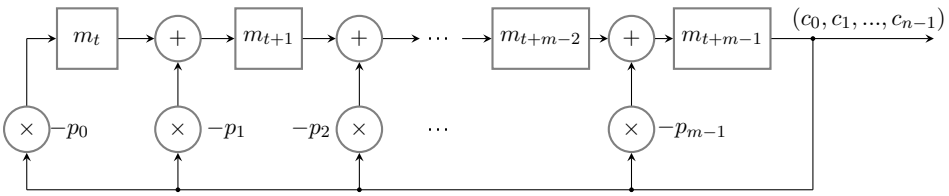

Рис. 1.1: Генератор последовательностей максимальной длины или кодер кода максимальной длины

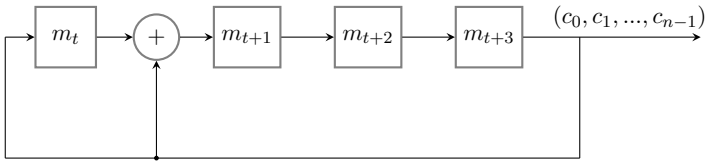

Рис. 1.2: Генератор последовательностей максимальной длины при $p(x) = 1 + x + x^4$ или кодер кода (15,4)

На рис. 1.2 показан пример кодера для случая когда $h(x) = p(x) = 1 + x + x^4$ – примитивный многочлен поля $GF(2^4)$. Положим состояние регистра равным $m(x) = m_0 + m_1 x + m_2 x^2 + m_3 x^3 = x^2 + x^3$, что соответствует кодированию информационной последовательности $(0, 0, 1, 1)$. В таблице 1.1 показаны вычисления, выполняемые кодером, так за тактом.

Нетрудно видеть, что, если мы обозначим через $m_t(x)$ полином, описывающий состояние регистра на такте t, то последовательность состояний окажется циклическим сдвигом последовательности полиномов $1, x, x^2, \ldots, x^{2^m-2}$ по модулю $p(x)$. В таблице 1.1 этой последовательности соответствуют строки с номерами 9, 10, ..., 8.

Такт	Состояние регистра	Входной символ
0	0011	1
1	1101	1
2	1010	0
3	0101	1
4	1110	0
5	0111	1
6	1111	1
7	1011	1
8	1001	1
9	1000	0
10	0100	0
11	0010	0
12	0001	1
13	1100	0
14	0110	0

Таблица 1.1: Работа генератора последовательности максимальной длины

Иными словами, множество состояний регистра пробегает все возможные ненулевые состояния.

Для произвольного постоянного во времени конечного автомата с памятью m бит выход автомата определяется детерминированной функцией содержимого его ячеек памяти. Максимальное число различных состояний равно 2^m, и эта же величина определяет максимальную длину псевдослучайной последовательности, генерируемой автоматом. В случае линейного автомата нулевому начальному состоянию соответствует нулевая выходная последовательность, поэтому максимальная длина оказывается равной $2^m - 1$, отсюда и название последовательности – последовательность максимальной длины.

Поскольку кодовые слова отличаются только начальными состояниями регистра, они являются сдвигами друг друга. Нулевому начальному состоянию соответствует нулевое кодовое слово. Минимальное расстояние (минимальный вес кодового слова) равно числу

единиц в последовательности максимальной длины. На рис. 1.1 и 1.2 видно, что единиц в кодовом слове будет столько, сколько ненулевых элементов поля заканчивается на 1, т.е. минимальное расстояние равно 2^{m-1}.

Докажем, что все сдвиги порождающего многочлена $g(x) = (x^{2^m} - 1)/h(x)$ различны. Предположим, что два сдвига совпадают, т.е. для некоторых i и j имеет место равенство

$$x^i g(x) = x^j g(x) \mod x^n - 1$$

Это означает, что для некоторого $r(x)$

$$x^i g(x) = x^j g(x) + r(x)(x^n - 1).$$

Поделив обе части на $g(x)$, получаем

$$x^i - x^j = r(x)p(x),$$

откуда следует, что $p(x)$ является делителем двучлена меньшей степени, чем $n = 2^m - 1$, что невозможно, поскольку $p(x)$ – примитивный многочлен. Тем самым мы убедились в том, все $2^m - 1$ сдвигов последовательностей максимальной длины различны и они вместе с нулевым словом образуют $(2^m - 1, m)$-код с расстоянием 2^{m-1}.

Пример 1.6 (Код Голея)**.** *Положим $n = 23$. Можно, перемножив многочлены, проверить, что*

$$x^{23} - 1 = (x - 1)g(x)\tilde{g}(x),$$

где

$$g(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1,$$

$$\tilde{g}(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1.$$

При этом многочлены $g(x)$ и $\tilde{g}(x)$ – взаимные друг к другу, т.е. коэффициенты одного многочлена получаются переписыванием в обратном порядке коэффициентов другого многочлена, $\tilde{g}(x) = x^{11}g(x^{-1})$. Получается, что существует два нетривиальных кода длины 23: код (23,12), порождаемый $g(x)$ и дуальный к нему код (23,11), порождаемый многочленом $(x-1)\tilde{g}(x)$. Докажем, что расстояния этих кодов равны соответственно 7 и 8.

Код, порождаемый $g(x)$, – частный случай БЧХ кодов, которые будут рассматриваться в следующем параграфе. Из теоремы БЧХ мы получим оценку $d \geq 5$. Покажем, что в коде Голея нет слов веса 5 и 6. Для этого сначала рассмотрим так называемый расширенный код Голея (24, 12), получаемый дописыванием проверки на четность к словам кода (23,12). При этом слова четного веса, конечно, не изменяются. Порождающая матрица расширенного кода Голея состоит из 12 сдвигов $g(x)$ (порождающая матрица исходного кода) и столбца из всех единиц. Все строки в результате имеют вес 8.

Лемма 1.6. *Расширенный код Голея самодуален.*

Доказательство. Этот факт можно проверить непосредственно, помножив порождающую матрицу на транспонированную порождающую матрицу. Другое доказательство основано на том, что проверочный многочлен кода Голея равен $h(x) = (x+1)\tilde{g}(x)$. Следовательно, порождающий многочлен дуального (23,11)-кода равен $g^{\perp}(x) = x^{12}h(x^{-1}) = (1+x)g(x)$. Отсюда следует, что код (23,11) – подкод (23,12), составленный из слов четного веса. Если дописать к порождающей матрице кода (23,11) строку из всех единиц, то получим альтернативную порождающую матрицу для (23,12), которая ортогональна себе, если не принимать во внимание последнюю строку. После дописывания столбца из всех единиц все строки, включая последнюю, станут ортогональными коду. □

Лемма 1.7. *В коде Голея вес слов четного веса кратен 4.*

Доказательство. Пусть $\boldsymbol{a}$ и $\boldsymbol{b}$ – ненулевые слова. Тогда скалярное произведение $(\boldsymbol{a}, \boldsymbol{b})$ равно нулю, если эти слова имеют четное число совпадающих позиций. Отсюда следует, что $w(\boldsymbol{a}+\boldsymbol{b}) = w(\boldsymbol{a}) + w(\boldsymbol{b}) - 4s$, где s – целое. Поскольку в порождающей матрице расширенного кода Голея все строки имеют вес 8, и код самодуален, вес всех линейных комбинаций кратен 4. □

Теорема 1.8. *В коде Голея нет слов веса 5 и 6.*

Доказательство. Наличие таких слов в коде Голея привело бы к существованию слов веса 6 в расширенном коде Голея, что противоречит Лемме 1.7. □

Итак, установлено, что код Голея (23,12) имеет расстояние 7.

Упражнение 6. *Убедитесь, что код Голея (23,12) – совершенный код, т.е. удовлетворяет границе Хэмминга.*

Нетривиальными совершенными двоичными кодами являются только коды Хэмминга и код Голея.

1.3 Кодирование и вычисление синдрома

Казалось бы, при современном уровне техники нет смысла останавливаться на деталях реализации устройств – подобные задачи сводятся к программированию в той или иной среде. Однако, возможность чрезвычайно эффективной реализации элементов кодеров и декодеров на регистрах сдвига – принципиальная особенность циклических кодов, немало способствовавшая их широкому распространению.

Начнем с задачи кодирования. Формула, связывающая кодовое слово $c(x)$ с соответствующим информационным многочленом $m(x)$ и порождающим многочленом $g(x)$ имеет вид

$$c(x) = m(x)g(x). \tag{1.6}$$

Степени при x можно рассматривать как индексы времени, умножение на x соответствует сдвигу на один такт. Поэтому операция (2.6) аналогична операции свертки двух последовательностей или операции фильтрации входной последовательности $m(x)$ фильтром с конечным откликом (или с конечной импульсной характеристикой или КИХ-фильтром) $g(x)$.

Пример 1.7. *Рассмотрим код Хэмминга (15,11). Выберем в качестве порождающего многочлена $g(x) = 1 + x + x^4$. На рис. 1.3. представлен фильтр-умножитель, который можно использовать в качестве кодера.*

До начала работы кодера в регистр записаны нули. Затем один за другим поступают информационные символы и складываются, умножаясь на порождающий многочлен. Выходами кодера будут

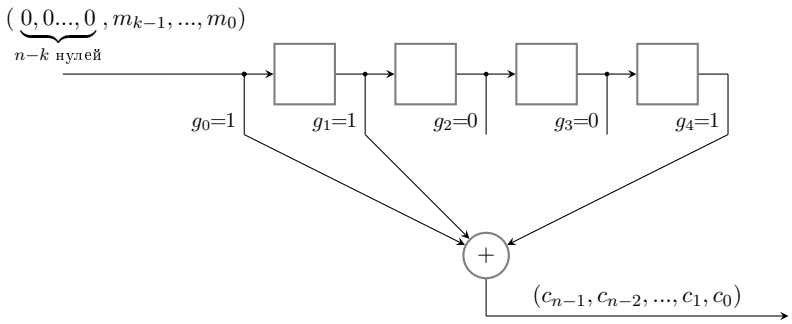

Рис. 1.3: Несистематический кодер циклического кода

последовательно $c_0 = m_0 g_0$, $c_1 = m_0 g_1 + m_1 g_0$, *и т.д., что в точности соответствует (2.6). После подачи* k *информационных символов поступают один за другим* $n-k$ *нулей, и последним кодовым символом будет, как и должно быть,* $c_{n-1} = m_k g_k$.

Аналогично, вычисление синдрома можно выполнить в виде умножения на проверочный полином кода $h(x)$ *принятой последовательности* $b(x) = c(x) + e(x)$, *где* $e(x)$ *– полином, соответствующий вектору ошибок.*

Итак, на базе КИХ-фильтров похожих на фильтр, приведенный на рис. 1.3, можно строить как кодеры, так и вычислители синдромов. Однако, это решение не самое удачное. Во-первых, желательно использовать систематические коды, т.к. при этом уменьшается сложность восстановления информации в декодере. Во-вторых, по крайней мере, если $k > n-k$, желательно уменьшить сложность вычисления синдрома, сделать ее пропорциональной $n-k$, а не k. Обе задачи, как мы увидим, успешно решаются применением фильтров с обратной связью, или, в терминах цифровой обработки сигналов, БИХ-фильтров (фильтров с бесконечной импульсной характеристикой).

Посмотрим, как можно сделать кодирование систематическим. Для этого запишем кодовое слово в виде

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

Эта запись означает, что в старших разрядах слова записана ин-

формационная последовательность, а полином $t(x)$ должен иметь степень не выше $n-k-1$ и быть таким, чтобы $c(x)$ делился на порождающий многочлен $g(x)$. Этого можно добиться, выбрав в качестве $t(x)$ остаток от деления $-x^{n-k}m(x)$ на $g(x)$, т.е. найти такие $t(x)$ и $q(x)$, что

$$-x^{n-k}m(x) = q(x)g(x) + t(x).$$

В этом случае $c(x) = -q(x)g(x)$, т.е. является кодовым словом. Для построения кодового слова само частное $q(x)$ не требуется, достаточно к информационной последовательности дописать остаток $t(x)$ от деления $-x^{n-k}m(x)$ на $g(x)$.

Вычисление остатка от деления с помощью регистра очень похоже на школьное деление чисел «столбиком». Начиная со старших разрядов, мы вычитаем делитель, умноженный на такой элемент поля, который обеспечивает равенство нулю старшего разряда делимого.

Пример 1.8. *Схема кодера для кода Хэмминга (15,11) с порождающим многочленом $g(x) = 1 + x + x^4$ показана на рис. 1.4. Кроме сумматоров и элементов задержки схема содержит два переключателя, на положениях которых надписаны номера соответствующие тактов.*

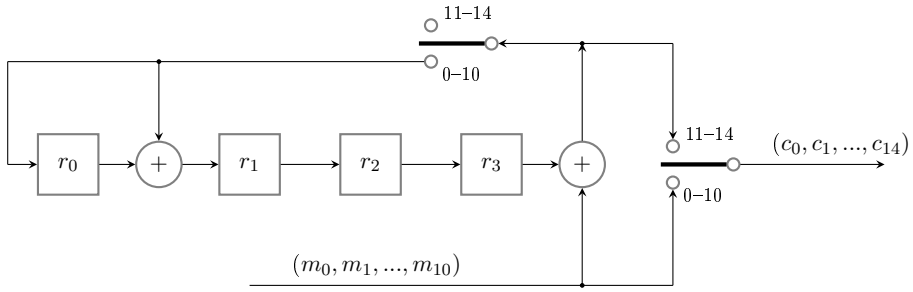

Рис. 1.4: Систематический кодер кода $(15, 11)$

Начальное положение регистра – нулевое. На первых одиннадцати тактах информационные символы поступают одновременно в обратную связь регистра и на выход схемы. За 11 шагов

в регистре формируется остаток от деления на информационной последовательности порождающий многочлен. Этот остаток на последних 4 тактах подается на выход как проверочные символы к данным информационным символам. Кодирование информационной последовательности $m = (m_0, \ldots, m_{10}) = (0,1,0,0,0,0,1,1,0,1,1)$ *систематическим кодером, показанным на рис. 1.4, шаг за шагом поясняется в таблице 1.2. Обратите внимание, что сначала поступают старшие разряды информационной последовательности. Казалось бы, отсутствует умножение на* $x^{n-k} = x^4$. *Оно присутствует неявно: за информационными символами как бы следуют нули и деление осуществляется, как и должно быть, в течение 11 тактов.*

Такт	Вход	Обратная связь	Состояние регистра	Выходной символ
–	–	–	0000	–
0	1	1	1100	1
1	1	1	1010	1
2	0	0	0101	0
3	1	0	0010	1
4	1	1	1101	1
5	0	1	1010	0
6	0	0	0101	0
7	0	1	1110	0
8	0	0	0111	0
9	1	0	0011	1
10	0	1	1101	0
11	–	–	0110	1
12	–	–	0011	0
13	–	–	0001	1
14	–	–	0000	1

Таблица 1.2: Кодирование систематическим кодером кода Хэмминга (15,11)

Результат вычислений в точности совпадает с результатом деления в столбик (единицы соответствуют коэффициентам при соответствующих степенях x*):*

$$
\begin{array}{l|l}
\mathbf{1101100001}0000 & \mathbf{10011} \\
10011 & 1100\ldots \\
\hline
\ 10000 & \\
\ 10011 & \\
\hline
\quad 11000 & \\
\quad 10011 & \\
\hline
\quad\ 1011 & \\
\quad\ 10011 & \\
\hline
\quad\quad 10000 & \\
\quad\quad 10011 & \\
\hline
\quad\quad\quad 11000 & \\
\quad\quad\quad 10011 & \\
\hline
\quad\quad\quad\ \mathbf{1011} &
\end{array}
$$

Рассмотрим теперь задачу вычисления синдрома. Если на выходе канала наблюдается последовательность $b(x) = c(x) + e(x)$, то синдромный полином $s(x)$ вычисляется либо умножением на $h(x)$ (по аналогии с несистематическим кодированием, см. рис. **??**) либо делением на порождающий многочлен $g(x)$:

$$s(x) = b(x) \bmod g(x).$$

Пример 1.9. *Схема вычисления синдрома для кода (15,11) показана на рис. 1.5. Предположим, что последовательность на выходе канала* $(b_0, b_1, \ldots, b_{14}) = (010101101101100)$*, что соответствует полиному* $b(x) = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^3 + x^2$*. Предполагается, что начальное состояние регистра – нулевое и последовательность поступает на вход, начиная со старших коэффициентов. Вычисления, выполняемые схемой на каждом такте работы, показаны в таблице 1.3. Честное деление в столбик показывает, что синдром вычислен правильно.*

Для исправления ошибок нужно не только подсчитать синдром, но и найти соответствующий этому синдрому вектор ошибок. Для этого можно использовать запоминающее устройство, в котором для каждого из $2^{n-k} - 1$ ненулевых синдромов хранится соответ-

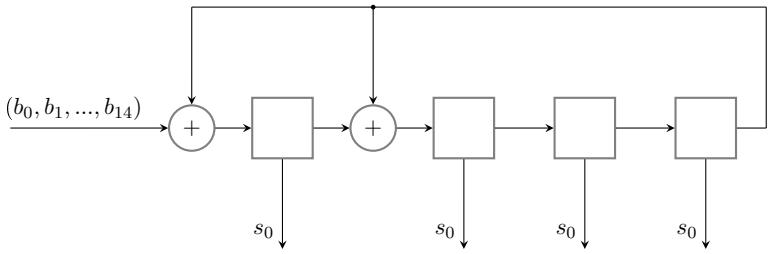

Рис. 1.5: Вычислитель синдрома для кода (15,11)

ствующий ему вектор ошибок длины n. Поскольку код циклический, можно немного сократить объем необходимой памяти.

Теорема 1.9. *Если*

$$s(x) = b(x) \mod g(x)$$

то

$$(xb(x)(\mathrm{mod}x^n - 1))\,(\mathrm{mod}g(x)) = xs(x)(\mathrm{mod}g(x)).$$

Доказательство. Обозначим через $q_b(x)$ частное от деления $b(x)$ на $g(x)$. Тогда

$$b(x) = g(x)q_b(x) + s(x)$$

и

$$xb(x) = xg(x)q_b(x) + xs(x).$$

Поскольку $xb(x)(\mathrm{mod}(x^n - 1) = xb(x) - b_{n-1}(x^n - 1)$, и поскольку $g(x)$ делит $x^n - 1$, из последнего тождества после приведения по модулю $g(x)$ получаем утверждение теоремы. □

Иными словами, теорема утверждает, что циклическому сдвигу последовательности длины n соответствует циклический сдвиг синдрома, выполняемый с приведением по модулю порождающего многочлена.

При декодировании циклического кода принятую последовательность можно записать в циклический регистр сдвига из n ячеек. Для каждого циклического сдвига заново вычисляется синдром и проверяется наличие вычисленного вектора в памяти декодера. Если он там есть, соответствующий вектор ошибок прибавляется к

Такт	Вход	Обратная связь	Состояние регистра
–	–	–	0000
0	0	0	0000
1	0	0	0000
2	1	0	1000
3	1	0	1100
4	0	0	0110
5	1	0	1011
6	1	1	0001
7	0	1	1100
8	1	0	1110
9	1	0	1111
10	0	1	1011
11	1	1	0001
12	0	1	1100
13	1	0	1110
14	0	0	0111

Таблица 1.3: Вычисление синдрома кода Хэмминга (15,11)

принятому слову и на этом декодирование заканчивается. Из теоремы 1.9 следует, что достаточно хранить только такие синдромы, которые не являются циклическими сдвигами других синдромов, уже помещенных в память декодера. Кажется, что таким способом можно сократить объем памяти примерно в $r = n - k$ раз, но на самом деле многие последовательности длины r имеют много меньше чем r различных сдвигов (например, последовательность из всех единиц). По этой причине память уменьшается не в r раз, а примерно в 4 раза. Декодер, использующий эту возможность, называют *декодером Меггитта*. Платой за экономию памяти является значительное увеличение времени работы декодера.

Задачи

1. Постройте циклические коды длин $n = 3, ..., 9$. Определите скорость и минимальное расстояние кодов и дуальных к ним.

2. Постройте все поля характеристики 2 из 8 элементов. Проверьте, что они изоморфны друг другу.

3. Определите порядки элементов мультипликативной группы поля $GF(q)$ при $q = 2, 3, 4, 5, 7, 8, 9$.

4. Постройте поле $GF(2^4)$ как кольцо вычетов по модулю $p(x) = 1 + x + x^4$. Найдите обратные элементы к элементам поля.

5. Постройте поле $GF(2^4)$ как кольцо вычетов по модулю $p(x) = 1 + x + x^2 + x^3 + x^4$. Найдите обратные элементы к элементам поля.

6. Представьте порождающую матрицу циклического кода Хэмминга (7,4) в систематической форме и выпишите соответствующую проверочную матрицу. Как будет выглядеть декодер Меггитта для этого кода?

7. Предположим, что кодом Хэмминга (7,4) передается сообщение (1011) и в канале ошибка произошла в позиции с номером 7. Повторите вычисления, выполняемые кодером при кодировании и декодером при вычислении синдрома. Используя результаты выполнения задачи 4, покажите, как произойдет исправление этой ошибки декодером Меггитта.

8. Докажите, что минимальная решетка циклического кода имеет максимальную по всем уровням сложность $\min\{2^k, 2^{n-k}\}$. Эта величина представляет собой максимально возможную сложность решетки линейного кода. Означает ли это, что сложность решетки кода Голея не может быть меньше 2^12?

Приложение

1.4 Кольца и поля. Основные определения и свойства

1.4.1 Кольцо вычетов по модулю некоторого числа.

Определение 1.4. *Кольцом называется множество R, на котором определены две операции, называемые сложением и умножением, и для которого справедливы аксиомы:*

1. *R образует абелеву группу по сложению;*
2. *R замкнуто относительно операции умножения;*
3. *ассоциативность: $a(bc) = (ab)c$;*
4. *дистрибутивность: $a(b + c) = ab + ac$.*

Обратим внимание на то, что в кольце может не быть единичного элемента по умножению. Если же он есть, кольцо называется *кольцом с единицей*.

В общем случае не требуется, чтобы операция умножения была коммутативной. Если же это условие выполняется, то имеем *коммутативное кольцо*.

Отметим также, что в кольце с единицей не обязательно определены обратные элементы для каждого элемента кольца. Если бы они были, то можно было бы говорить о том, что ненулевые элементы образуют группу по умножению, и тогда это уже было бы поле.

Определение 1.5. *Поле т называется множество F, на котором определены две операции, называемые сложением и умножением, и для которого справедливы аксиомы:*

1. *F образует абелеву группу по сложению;*
2. *множество ненулевых элементов F образует абелеву группу по умножению;*
3. *дистрибутивность: $a(b + c) = ab + ac$.*

Пример 1.10. *При любом q множество целых чисел по модулю числа q является кольцом, при условии, что операции сложения и умножения выполняются с последующим приведением по модулю q. Это кольцо является коммутативным кольцом с единицей. Оно является полем, если число q – простое.*

Упражнение 7. *Пусть R_q обозначает кольцо вычетов целых чисел по модулю q. Для $q = 2, \ldots, 7$ определите, какие из элементов полей и колец имеют обратные элементы. Сформулируйте общее утверждение о существовании обратных элементов в таких кольцах.*

Поле из q элементов в дальнейшем будем обозначать $GF(q)$. GF – аббревиатура словосочетания Galois Field (поле Галуа), связанного с именем французского ученого с очень необычной биографией. Настоятельно рекомендуем почитать о нем, хотя бы в Интернете.

1.4.2 Кольцо многочленов

Рассмотренные примеры подсказывают, что при простых q существуют поля с q элементами – это поля целых чисел по модулю q. Можно построить поля мощности q при составном q. Чтобы построить такие поля, рассмотрим специальное кольцо – кольцо многочленов.

Элементами кольца являются многочлены вида $a(x) = a_0 + a_1x + ... + a_mx^m$, где коэффициенты выбираются из некоторого поля $GF(q)$. Операции сложения и умножения многочленов выполняются обычным образом. Такое кольцо обозначим $F[x]$.

Напомним, что приведенным называется многочлен, у которого коэффициент при старшей степени аргумента равен 1.

Определение 1.6. *Для приведенного многочлена $p(x)$ степени больше единицы кольцом многочленов по модулю $p(x)$ называется множество всех многочленов степени меньше степени $p(x)$ с операциями умножения и сложения, определенными как сложение и умножение по модулю $p(x)$. Это кольцо обозначим $F_{p(x)}[x]$.*

Теорема 1.10. *Кольцо $F_{p(x)}[x]$ является полем тогда и только тогда, когда $p(x)$ – простой (т.е. приведенный и неразложимый)*

многочлен.

Доказательство. Начнем с доказательства необходимости. Покажем, что при простом $p(x)$ каждый элемент кольца $F_{p(x)}[x]$ имеет обратный элемент по умножению. Для этого выберем произвольный элемент $s(x)$ и покажем, что все произведения вида $s(x)a(x) \bmod p(x)$ различны. Это будет означать, что одно из произведений равно 1. Соответствующий элемент $a(x)$ будет обратным к $s(x)$. Предположим противное, т.е. что для некоторых $a(x)$ и $b(x)$ имеет место $s(x)a(x) = s(x)b(x) \bmod p(x)$, или, что то же самое, $s(x)d(x) = 0 \bmod p(x)$ имеет место для некоторого $d(x)$. Это означает, что $s(x)d(x) = t(x)p(x)$, при некотором $t(x)$. Один из многочленов в левой части обязан делиться на $p(x)$, поскольку $p(x)$ – простой. Это невозможно, поскольку степень каждого из этих двух многочленов меньше степени $p(x)$. Тем самым доказано существование обратного элемента и данное кольцо – поле.

Чтобы доказать необходимость, заметим, что при непростом $p(x) = u(x)v(x)$ делители $p(x)$ не имеют обратных, поскольку иначе можно было бы записать.

$$u(x) = [u(x)v(x)]\, v^{-1}(x) = p(x)v^{-1}(x) = 0 \mod p(x).$$

Отсюда следует, что $F_{p(x)}[x]$ – не удовлетворяет аксиомам поля при непростом $p(x)$. □

Поле, построенное как кольцо многочленов по модулю неприводимого $p(x)$ степени m с коэффициентами из $GF(q)$, содержит q^m элементов, называется *расширением* $GF(q)$ и обозначается как $GF(q^m)$.

Итак, любой простой многочлен может быть использован для построения поля Галуа $GF(q^m)$.

Пример 1.11. *Два примера колец многочленов по модулю многочленов второй степени приведены в таблице 6П.1. Кольцо многочленов по модулю* $1 + x + x^2$ *является полем.*

Хотя поле Галуа, как мы видели, может быть задано любым простым многочленом, оно единственно с точностью до изоморфизма. Для построения кодов и вычислений в поле Галуа оказывается

$p(x) = 1 + x^2$		$p(x) = 1 + x + x^2$	
Элемент	Обратный элемент	Элемент	Обратный элемент
0	–	0	–
1	1	1	1
x	x	x	$1 + x$
$1 + x$	не определен	$1 + x$	x

Таблица 1.4: Два кольца многочленов по модулю полинома второй степени

удобным описание мультипликативной группы поля в виде набора степеней одного из элементов поля.

Остановимся подробнее на свойствах мультипликативной группы поля.

1.4.3 Мультипликативная группа поля Галуа

Пусть имеется произвольная группа по умножению, состоящая из $q-1$ элементов и выберем некоторый элемент группы a. Очевидно, вместе с этим элементом группе принадлежать все элементы вида a^i, $i = 1, 2, \ldots$ Поскольку в группе конечное число элементов, для некоторого числа s получим $a^s = 1$.

Определение 1.7. *Наименьшее s такое, что $a^s = 1$ называется порядком элемента a.*

Согласно теореме 4П.2, порядок подгруппы является делителем порядка группы, поэтому $q - 1$ делится на s и, в частности, может совпадать с s. Таким образом, каждый ненулевой элемент поля удовлетворяет уравнению

$$x^{q-1} - 1 = 0 \tag{1.7}$$

и любой элемент поля уравнению

$$x^q - x = 0. \tag{1.8}$$

Это утверждение называется *малой теоремой Ферма*. Оно приводит к важному свойству, формулируемому ниже в виде теоремы

6П.2, доказательство которой станет простым, благодаря двум элементарным леммам.

Лемма 1.11. *Если порядки s_a и s_b элементов a и b взаимно просты, то порядок их произведения ab равен произведению их порядков $s_{ab} = s_a s_b$.*

Доказательство. Поскольку $(ab)^{s_a s_b} = 1$, достаточно доказать, что порядок ab не может быть меньше чем $s_a s_b$. Если для некоторого l имеет место $(ab)^l = 1$, то $(ab)^{ls_a} = b^{ls_a} = 1$. Это возможно только если показатель степени ls_a кратен порядку s_b элемента b. Поскольку s_a и s_b взаимно просты, делаем вывод, что l кратно s_b. Аналогично можно убедиться, что l кратно s_a. В итоге заключаем, что $l = s_a s_b$. □

Лемма 1.12. *Если число s^t является делителем $q-1$, то в поле Галуа найдется элемент порядка s^t.*

Доказательство. Положим $q - 1 = s^t u$. Уравнению $x^u = 1$ удовлетворяет $u - 1 < q - 1$ элементов, т.е. не все элементы поля. Пусть a не удовлетворяет этому уравнению, т.е. $a^u \neq 1$. При этом для элемента $b = a^u$ имеет место $b^{s^t} = a^{q-1} = 1$, т.е. его порядок либо равен s^t, либо делит его, т.е. равен s^j, $j < t$. Поскольку множество всех элементов поля, которые удовлетворяют уравнению $x^{s^t} = 1$ содержит все элементы, удовлетворяющие $x^{s^j} = 1$, и при этом число решений $s^t - 1 > s^j - 1$, непременно найдется элемент, удовлетворяющий только первому из этих двух уравнений, т.е. элемент порядка s^t (см. пример 6П.3). □

Теорема 1.13. *В поле из q элементов найдется элемент порядка $q-1$.*

Доказательство. При простом $q-1$ все элементы имеют такой порядок, т.к. $q-1$ – единственный делитель порядка группы. Пусть число $q-1$ составное, его разложение на простые множители запишем в виде $q - 1 = \prod_{i=1}^{s} p_i^{\nu_i} = \prod_{i=1}^{s} \theta_i$.

Из леммы 1.11 и 1.12 теперь следует, что найдется элемент порядка $q-1$. □

Теорема 1.13, по сути, утверждает, что все ненулевые элементы поля Галуа можно получить возведением в степень одного элемента поля. Таким образом, в терминологии теории групп мультипликативная группа поля Галуа является *циклической группой*. Элемент, из которого последовательным применением групповой операции можно получить всю группу, называется *образующим элементом циклической группы*. Этот элемент играет настолько большую роль в теории групп, что у него есть еще одно название.

Определение 1.8. *В поле $GF(p^m)$ элемент порядка $p^m - 1$ называется примитивным.*

Пример 1.12. *Мультипликативная группа поля $GF(5^2)$ содержит 24 элемента, т.е. порядки элементов являются числами из множества $\{2, 3, 4, 6, 8, 12, 24\}$. Если x – примитивный элемент, то x^s имеет порядок 2 при $s = 12$, порядок 3 при $s = 8, 16$, порядок 4 при $s = 6, 18$, порядок 6 при $s = 4, 20$, порядок 8 при $s = 3, 9, 15, 21$, порядок 12 при $s = 2, 10, 14, 22$. Элементы степеней 5,7,11,13,17,19,23 (взаимно простые с 24) имеют порядок 24, т.е. являются примитивными.*

Мы показали выше, что любой неприводимый многочлен можно использовать для построения поля Галуа. Теперь стало ясно, что поле может быть описано заданием одного примитивного элемента. Наша следующая цель – связать эти два представления, полиномиальное и степенное.

Пример 1.13. *Рассмотрим кольцо многочленов по модулю $p(x) = 1 + x + x^3$. Поскольку многочлен неразложим над $GF(2)$, это кольцо является полем $GF(2^3)$. В нем должен существовать элемент порядка 7. Возможно, элемент x является таковым. Чтобы проверить этот факт, нужно подсчитать его степени по модулю $p(x)$. Вычисления сведены в таблицу 1.5. Каждый новый элемент полиномиального представление получен умножением предыдущего на x по модулю $p(x)$.*

Вычисления подтвердили, что x – примитивный элемент. Как мы увидим позже, совсем не любой многочлен примитивен и мо-

Степень примитивного элемента	Полиномиальное представление	Двоичная запись полинома
x	x	010
x^2	x^2	100
x^3	$x^3 = 1 + x$	011
x^4	$x + x^2$	110
x^5	$x^2 + x^3 = 1 + x + x^2$	111
x^6	$x + x^2 + x^3 = 1 + x^2$	101
$x^7 = x^0$	$x + x^3 = 1$	001

Таблица 1.5: Мультипликативная группа поля $GF(2^3)$

жет быть использован для описания ненулевых элементов в виде циклической группы.

Упражнение 8. *Постройте поле* $GF(3^2)$.

Определение 1.9. *Простой многочлен* $p(x)$ *степени* m *с коэффициентами из* $GF(p)$ *называется примитивным многочленом, если в поле полиномов по модулю* $p(x)$ *элемент* x *является примитивным.*

Рассмотрим произвольное поле $GF(q)$ и число 1 в этом поле. Складывая единицу с собой, будем получать различные элементы поля, и, в силу конечности поля, для некоторого p выполняется равенство $1+1+\ldots+1(p$ слагаемых$)=0$. Минимальное число p, для которого выполняется это равенство, называется *характеристикой поля*. Число p – простое, т.к. иначе мы получили бы равенство $p = st = 0$, из которого бы следовало, что либо r, либо s равны нулю.

Само поле называют полем характеристики p. В таком поле имеют место тождества

$$px = 0, \tag{1.9}$$

$$(x + y)^p = x^p + y^p. \tag{1.10}$$

Первое следует из определения, а для доказательства второго нужно воспользоваться формулой бинома Ньютона. При этом окажется, что все члены кроме x^p и y^p содержат в качестве коэффици-

ентов биномиальные коэффициенты кратные числу p и равны нулю в силу (2.9).

1.4.4 Минимальные многочлены

Корнем многочлена $p(x)$ называется элемент x, для которого имеет место равенство $p(x) = 0$. Если a является корнем $p(x)$, то $p(x)$ делится на двучлен $(x - a)$. Таким образом, многочлен степени m может иметь не больше m корней.

Поскольку все элементы поля $GF(q)$ удовлетворяют (2.8), имеет место разложение

$$x^q - x = \prod_{\alpha \in GF(q)} (x - \alpha). \tag{1.11}$$

Отсюда следует, что в поле $GF(p^m)$ всякий элемент является корнем некоторого многочлена степени p^m или меньше.

Определение 1.10. *Минимальным многочленом $M(x)$ элемента α над полем $GF(p)$ в поле $GF(p^m)$ называется нормированный многочлен наименьшей степени, корнем которого является α.*

Свойства минимальных многочленов:

Свойство 1. Минимальный многочлен элемента α неприводим.

Доказательство. В противном случае один из сомножителей имел бы корнем α и его степень была бы меньше степени $M(x)$. □

Свойство 2. Любой многочлен, имеющий корень α, делится на $M(x)$.

Доказательство. Пусть для некоторого $f(x)$ имеет место $f(\alpha) = 0$. Тогда остаток $f(x)$ от деления $f(x)$ на $M(x)$ тоже должен иметь корень α. Но степень остатка меньше степени $M(x)$, что противоречит предположению о минимальности $M(x)$. □

Свойство 3. $x^{p^m} - x$ делится на $M(x)$.

Доказательство. Это свойство непосредственно следует из свойства 2. □

Свойство 4. Степень $M(x)$ не превышает m.

Доказательство. Поле $GF(p^m)$ представляет собой m-мерное линейное пространство полиномов степени не более $m-1$. Элементы $1, \alpha, \alpha^2, ...\alpha^m$ линейно зависимы и, следовательно, для некоторого набора коэффициентов $f_0, f_1, ..., f_m$ имеет место равенство $f_0 + f_1\alpha + ... + f_m\alpha^m = f(\alpha) = 0$. □

Свойство 5. Степень минимального многочлена $M(x)$ примитивного элемента равна m.

Доказательство. Если бы степень была меньше, например d, то степени корней $M(x)$, приведенные по модулю $M(x)$ образовали бы поле из $p^d < p^m$ элементов, и порядок примитивного элемента был бы меньше числа элементов поля, что противоречит определению примитивного элемента. □

Глава 2

БЧХ-коды и РС-коды

Не будет преувеличением сказать, что БЧХ и РС-коды – рекордсмены по практическому использованию в разнообразных системах и устройствах хранения и передачи данных. Кроме того, эти коды важны с теоретической точки зрения, поскольку, по сути, все остальные кодовые конструкции алгебраических блоковых кодов являются в той или иной степени их обобщением и развитием. Большая часть лучших известных кодов получается той или иной модификацией кодов этого класса.

Коды БЧХ получили свое название от имен первооткрывателей Боуза, Рой-Чоудхури (1960) и Хоквингема (1959), название РС-кодов связано с их авторами Ридом и Соломоном (1960).

Важно отметить, что коды БЧХ и РС мы рассматриваем как «длинные» коды с «простым» декодированием. Эти слова надо понимать следующим образом. Коды БЧХ образуют бесконечную последовательность кодов, для которых известен способ построения и способ их декодирования с полиномиальной сложностью в канале с жесткими решениями с исправлением числа ошибок до половины «конструктивного расстояния». Конструктивное расстояние БЧХ-кодов при фиксированном размере алфавита асимптотически плохое, доля исправляемых ошибок убывает с ростом длины кодов, тем не менее, при конечных длинах эти коды достаточно хороши. Коды Рида-Соломона имеют наибольшее возможное расстояние, удовлетворяющее границе Синглтона, но алфавит кода растет с его длиной.

Итак, достоинство рассматриваемых в данном разделе кодов – простота конструкции и алгоритма декодирования в канале с жесткими решениями. Недостаток – отсутствие разумного по сложности алгоритма декодирования в канале с мягкими решениями. Этот недостаток частично преодолевается использованием предложенного Форни декодирования по МОР (минимуму обобщенного расстояния), которое сводится к многократному декодированию в жестком канале. Этот метод будет рассмотрен позже при изучении каскадных кодов.

2.1 Определение БЧХ-кода

Прежде, чем будут даны формальные определения, попробуем самостоятельно построить код с минимальным расстоянием 5, т.е. код, исправляющий две ошибки. В примерах на построение линейных кодов мы легко научились строить коды с расстоянием 1, 2, 3, 4. Не может быть, чтобы следующий шаг оказался для нас непосильным.

Запишем проверочную матрицу кода Хэмминга (15,11) в виде

$$H = \begin{pmatrix} 1 & \alpha^1 & \alpha^2 & \cdots & \alpha^{14} \end{pmatrix}.$$

где α обозначает примитивный элемент поля $GF(2^4)$, построенного, например, с помощью примитивного многочлена $p(x) = 1 + x + x^4$. Чтобы получить двоичную форму матрицы (она нам сейчас не нужна!), достаточно записать степени примитивного элемента в виде двоичных коэффициентов их разложения по степеням α.

Упражнение 9. *Постройте поле* $GF(2^4)$.

Результат выполнения этого упражнения приведен в таблице 2.1.

Одну ошибку, как и должно быть, код исправляет. Пусть $b(x) = c(x) + e(x)$ – полином, соответствующий принятой из канала последовательности при передаче кодового слова $c(x)$, полином $e(x)$ соответствует вектору ошибок. Тогда синдром мы запишем как $s = b(\alpha) = e(\alpha)$. Если, скажем, ошибка произошла в позиции с номером i, то $b(x) = x^i$ и $s = \alpha^i$ и по s мы однозначно определим

номер ошибочной позиции i. Посмотрим, что произойдет в случае двух ошибок.

Пусть $e(x) = x^i + x^j$, $i \neq j$. Тогда

$$s = \alpha^i + \alpha^j. \tag{2.1}$$

В правой части получили некоторый элемент поля, который ошибочно указывает нам на некоторую позицию, где ошибки нет. Очевидно, имея только одну строку в матрице H, мы не сможем исправить две ошибки. Уравнение (1) – одно уравнение с двумя неизвестными. Вот если бы у нас было второе уравнение ... Таким образом, нужна дополнительная избыточность в виде еще одной строки и тогда у нас была бы система уравнений вида

$$\begin{array}{l} s_1 = \alpha^i + \alpha^j \\ s_2 = f(\alpha^i) + f(\alpha^j) \end{array}. \tag{2.2}$$

Если выбрать подходящую функцию f, то из этих уравнений мы смогли бы найти единственную пару (i, j) и тем самым гарантировать исправление двух ошибок, т.е. минимальное расстояние 5. Заметим, что линейные функции не годятся, поскольку уравнения окажутся линейно зависимыми. Из нелинейных функций первое, что приходит на ум – возведение в квадрат.

Это правильное решение, но не для любого поля. Для полиномов с коэффициентами из поля $GF(2)$ имеет место тождество $b(x^2) = (b(x))^2$, из которого следует $s_2 = (s_1)^2$. Это означает, что любое решение первого уравнения одновременно является решением второго.

Выберем функцию $f(x) = x^3$. Тогда новая проверочная матрица примет вид

$$H = \begin{pmatrix} 1 & \alpha^1 & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} & \alpha^{12} & \alpha^{13} & \alpha^{14} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} \end{pmatrix}. \tag{2.3}$$

Во второй строке записаны элементы первой строки, возведенные в третью степень.

Пусть, как и прежде, i и j – номера ошибочных позиций, и введем обозначения $x = \alpha^i$, $y = \alpha^j$. Тогда, обозначив скалярное произ-

Степень	Многочлен	Последовательность
$-\infty$	0	0000
0	1	0001
1	x	0010
2	x^2	0100
3	x^3	1000
4	$1+x$	0011
5	$x+x^2$	0110
6	x^2+x^3	1100
7	$1+x+x^3$	1011
8	$1+x^2$	0101
9	$x+x^3$	1010
10	$1+x+x^2$	0111
11	$x+x^2+x^3$	1110
12	$1+x+x^2+x^3$	1111
13	$1+x^2+x^3$	1101
14	$1+x^3$	1001

Таблица 2.1: Поле $GF(2^4)$, порожденное многочленом $p(x) = 1 + x + x^4$

ведение первой строки с принятой последовательностью через s_1, второй – через s_3, получаем уравнения

$$\begin{cases} x+y=s_1 \\ x^3+y^3=s_3 \end{cases}. \tag{2.4}$$

При одной или двух ошибках $s_1 \neq 0$, поэтому, деля второе уравнение на первое, находим

$$x^2+y^2+xy = s_3/s_1.$$

Здесь мы воспользовались тем, что в поле характеристики 2 вычитание и сложение эквивалентны. Поскольку в этом поле $(a+b)^2 = a^2+b^2$, имеем

$$xy = s_3/s_1 - s_1^2.$$

По теореме Виета, произведение и сумму двух чисел, можно интерпретировать их как коэффициенты квадратного уравнения с корнями x и y:

$$z^2 + \sigma_1 z + \sigma_2 = 0, \tag{2.5}$$

где

$$\sigma_1 = s_1, \quad \sigma_2 = s_3/s_1 - s_1^2. \tag{2.6}$$

Итак, задачу декодирования можно считать решенной, при условии, что мы умеем решать квадратные уравнения над конечными полями. Хотя решения уравнений небольшой степени (до 4-й) можно искать неперебо́рными методами, на практике чаще всего их решают просто подстановкой в уравнение вида (2.5) всех ненулевых элементов поля. Порядок сложности этой процедуры примерно такой же, как порядок сложности вычисления синдрома. Поэтому логическая простота такого решения делает его предпочтительным. Подытожим процедуру декодирования данного кода в виде алгоритма.

Алгоритм

Исходные данные: двоичная последовательность b.

Шаг 1. Вычисляем компоненты синдрома s_1 и s_3.

Шаг 2. Если $s_1 = s_3 = 0$, имеем кодовое слово, информационные символы выдаются получателю. Работа алгоритма закончена. В противном случае переходим к шагу 3.

Шаг 3. Если $s_1 = 0$ и $s_3 \neq 0$, число ошибок не меньше 3. Отказ от декодирования. Работа алгоритма закончена. В противном случае переходим к шагу 4.

Шаг 4. Если $s_1 \neq 0$ и $s_3 = s_1^3$, имеет место одиночная ошибка. Позиция ошибки определяется значением s_1. После исправления ошибки информационные символы выдаются получателю. Работа алгоритма закончена. В противном случае переходим к шагу 5.

Шаг 5. Находим коэффициенты уравнения (2.5) по формулам (2.6).

Шаг 6. Поочередной подстановкой элементов $1, \alpha^1, \ldots, \alpha^{n-1}$ в уравнение (2.5) находим корни уравнения. Если корни найдены,

исправляем ошибки и выдаем информационные символы получателю. В противном случае (если корни не найдены), число ошибок не меньше 3 . Отказ от декодирования. Работа алгоритма закончена.

Пример 2.1. *Рассмотрим код с проверочной матрицей (2.3). Предположим, что на выходе канала наблюдается последовательность (1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1). Вычисляем синдром:* $s_1 = \alpha^{11}$ *и* $s_3 = \alpha^4$*, находим коэффициенты уравнения,* $\sigma_1 = \alpha^{11}$*,* $\sigma_2 = \alpha^{11}$*. Поочередной подстановкой элементов поля убеждаемся в том, что корнями являются* α^2 *и* α^9*. Это означает, что ошибочными являются позиции с номерами 3 и 10. Кодовое слово с исправленными ошибками имеет вид (1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1). Информационные символы (1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1) выдаются получателю (в предположении, что использовано систематическое кодирование).*

Итак, мы убедились в том, что код, заданный проверочной матрицей вида (2.3), исправляет любые 2-кратные ошибки. Можно утверждать, что его минимальное расстояние равно 5, а число проверочных символов не больше 8 (оно равно 8, если строки H линейно независимы, что имеет место в данном случае).

Построенный код – БЧХ-код (15,7) с минимальным расстоянием 5.

Успешный опыт построения кода с расстоянием 5 подсказывает прямой путь к построению кодов с еще большими расстояниями. Прежде, чем мы продвинемся дальше по этому пути, имеет смысл рассмотреть подробнее свойства построенного кода.

Первая строка в (2.3) показывает, что $c(x)$ – кодовое слово, если $c(\alpha) = 0$, откуда немедленно следует, что $\alpha, \alpha^2, \alpha^4, \ldots$ являются корнями $c(x)$, т.е. все корни минимального многочлена являются корнями $c(x)$. Это означает, что $c(x)$ делится на минимальный многочлен $M_1(x) = p(x)$ примитивного элемента α. Точно также мы устанавливаем, что $c(x)$ делится на минимальный многочлен $M_3(x)$ элемента α^3. Отсюда следует, что построенный код – циклический с порождающим многочленом

$$g(x) = M_1(x)M_3(x).$$

Теорема 2.1 (Граница БЧХ). *Пусть $g(x)$ – порождающий многочлен циклического кода и α – примитивный элемент конечного поля. Если для некоторых чисел $b \geq 0$ и $d \geq 1$ имеют место равенства*

$$g(\alpha^i) = 0, \quad i = b, \ldots, b + d - 2 \tag{2.7}$$

то минимальное расстояние кода не меньше d.

Иными словами, если $d-1$ последовательных степеней примитивного элемента поля обращают в ноль порождающий многочлен кода, то минимальное расстояние не меньше d. В рассмотренном выше примере кода, исправляющего 2 ошибки, такими степенями были 1, 2, 3, 4. При этом, как мы убедились, расстояние кода было равно 5.

Доказательство. Из условия теоремы следует, что каждое кодовое слово $c(x)$ обращается в ноль при подстановке вместо x любого из $d-1$ элементов вида α^i, $i = b, \ldots, b + d - 2$. Отсюда следует, что проверочная матрица кода может быть записана в виде

$$H = \begin{pmatrix} 1 & \alpha^b & \alpha^{2b} & \cdots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \cdots & \alpha^{(n-1)(b+1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{b+d-2} & \alpha^{2(b+d-2)} & \cdots & \alpha^{(n-1)(b+d-2)} \end{pmatrix}. \tag{2.8}$$

Докажем, что любые $d-1$ столбцов линейно независимы. Для этого рассмотрим произвольный набор индексов $i_1, \ldots, i_{d-1}$ и соответствующую подматрицу

$$\begin{pmatrix} \alpha^{i_1 b} & \alpha^{i_2 b} & \cdots & \alpha^{i_{d-1} b} \\ \alpha^{i_1 (b+1)} & \alpha^{i_2 (b+1)} & \cdots & \alpha^{i_{d-1} (b+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{i_1 (b+d-2)} & \alpha^{i_2 (b+d-2)} & \cdots & \alpha^{i_{d-1} (b+d-2)} \end{pmatrix}$$

Ее определитель равен

$$\alpha^{(i_1+\ldots+i_{d-1})b}\det\begin{pmatrix}1 & 1 & \cdots & 1\\ \alpha^{i_1} & \alpha^{i_2} & \cdots & \alpha^{i_{d-1}}\\ \vdots & \vdots & \ddots & \vdots\\ \alpha^{i_1(d-2)} & \alpha^{i_2(d-2)} & \cdots & \alpha^{i_{d-1}(d-2)}\end{pmatrix}. \qquad (2.9)$$

Известно тождество

$$\det\begin{pmatrix}1 & 1 & \cdots & 1\\ a_1 & a_2 & \cdots & a_m\\ \vdots & \vdots & \ddots & \vdots\\ a_1^{m-1} & a_2^{m-1} & \cdots & a_m^{m-1}\end{pmatrix} = \prod_{j=1}^{m-1}\prod_{i=j+1}^{m}(a_i - a_j), \qquad (2.10)$$

в котором матрица в левой части называется *матрицей Вандермонда*. Ее определитель называют *определителем Вандермонда*. Он отличен от нуля, если элементы $a_1, \ldots, a_m$ различны.

Нетрудно видеть, что определитель в (2.9) является частным случаем определителя Вандермонда. Поскольку α – примитивный элемент, его различные степени различны, определитель не равен нулю, ранг матрицы H не меньше $d-1$, а, следовательно, минимальное расстояние кода не меньше d. □

Определение 2.1. *Циклический код длины n над $GF(q)$ называется кодом БЧХ с конструктивным расстоянием d, если для некоторого $b \geq 0$ порождающий многочлен кода равен*

$$g(x) = HOK\{M_i(x), i = b, b+1, \ldots, b+d-2\}. \qquad (2.11)$$

Заметим, что минимальные многочлены различных элементов поля либо взаимно просты либо совпадают. В определение мы пишем наименьшее общее кратное, а не произведение минимальных многочленов, поскольку минимальные многочлены часто совпадают.

Например, рассмотренный выше код с расстоянием 5 имел корнями 1,2,3 и 4 степени примитивного элемента. Для 1, 2 и 4 степени

минимальным многочленом служит общий многочлен, по которому было построено само поле. Поэтому порождающий многочлен – произведение не 4, а только двух минимальных многочленов, соответствующих первой и третьей степеням примитивного элемента.

Теорема 2.2. *Пусть код БЧХ длины n с конструктивным расстоянием d над полем $GF(p)$ задан порождающим многочленом, корнями которого являются $d-1$ последовательных степеней элемента α порядка n в некотором расширении $GF(p^m)$ поля $GF(p)$. Тогда минимальное расстояние кода не меньше d, а размерность кода не меньше $n - m(d-1)$.*

Доказательство. Обоснования требует только оценка размерности кода. Поскольку степень каждого минимального многочлена не превышает m, а число сомножителей, образующих порождающий многочлен, не превышает $d-1$, степень произведения не больше $m(d-1)$. Эта величина является верхней оценкой избыточности кода. □

Коды БЧХ длины $n = q^m - 1$ называют *примитивными*. Длины непримитивных БЧХ-кодов являются делителями числа $q^m - 1$.

Для двоичных БЧХ кодов оценка теоремы 2.2 может быть существенно улучшена. Поскольку минимальный многочлен $M_{2i}(x)$ совпадает с $M_i(x)$, выбрав в качестве порождающего многочлена

$$g(x) = \text{НОК}\left\{M_1(x), M_3(x), \ldots, M_{2t-1}(x)\right\} \tag{2.12}$$

можно гарантировать конструктивное расстояние $d = 2t+1$. Отсюда получаем оценку размерности двоичного БЧХ кода с минимальным расстоянием $d = 2t + 1$:

$$k \geq n - mt. \tag{2.13}$$

2.2 Построение БЧХ-кодов. Примеры

Из предыдущего параграфа, в принципе, понятно, как можно построить БЧХ код с заданным числом исправляемых ошибок. Для этого нужно подобрать последовательность корней порождающего

многочлена в соответствии с определением **??** и построить порождающий многочлен в соответствии с (2.11) или (2.12).

Заметим, что истинная избыточность кода зависит от того, какими конкретно окажутся наборы примитивных многочленов.

Те элементы поля, которые имеют одинаковые минимальные многочлены, называют *сопряженными элементами*. Поскольку в поле характеристики p значения любого полинома в точках β и β^p одинаковы, то подмножества сопряженных элементов можно получать возведением элементов в степень p. Пусть для некоторого s имеет место равенство $\beta = \alpha^s$, где α– примитивный элемент поля, и пусть m_s – наименьшее число такое, что $p^{m_s} s = s \pmod{p^m - 1}$. Наборы показателей степеней

$$\{s, ps, p^2 s, \ldots, p^{m_s} s\}$$

при различных s образуют непересекающиеся множества, называемые *циклотомическими классами*. В каждом классе наименьшее число s называется *представителем класса*.

Циклотомические классы не пересекаются и покрывают все множество показателей степеней мультипликативной группы поля. Оценки параметров БЧХ-кодов определяются структурой циклотомических классов, и выбранным расширением конечного поля.

Пример 2.2. *Циклотомическими классами по модулю 15 являются множества*

$$C_0 = \{0\};$$
$$C_1 = \{1, 2, 4, 8\};$$
$$C_3 = \{3, 6, 12, 9\};$$
$$C_5 = \{5, 10\};$$
$$C_7 = \{7, 14, 13, 11\}.$$

Исходя из структуры циклотомических классов, легко построить таблицу БЧХ-кодов длины 15. Их параметры приведены в таблице 2.2. Жирным шрифтом выделены показатели степеней корней порождающих многочленов.

Упражнение 10. *Постройте циклотомические классы и таблицу кодов длины 31.*

Показатели степеней корней	Порождающий многочлен	Размерность $n - \deg g(x)$	Расстояние
0	$M_0(x) = x + 1$	14	2
1,2,4,8	$M_1(x)$	11	3
0,1,2,4,8	$M_0(x)M_1(x)$	10	4
1,2,3,4,6,...	$M_1(x)M_3(x)$	7	5
1,...,5, 6,8,...	$M_1(x)M_3(x)M_5(x)$	5	7
1,2,...,14	$M_1(x)M_3(x)M_5(x)M_7(x)$	1	15

Таблица 2.2: Примеры БЧХ-кодов длины 15

Пример 2.3. *Приведем пример непримитивного БЧХ кода. Положим* $n = 23$*. Циклотомические классы имеют вид:*

$$C_0 = \{0\};$$
$$C_1 = \{1, 2, 4, 8, 16, 9, 18, 13, 3, 6, 12\};$$
$$C_5 = \{5, 10, 20, 17, 11, 22, 21, 19, 15, 7, 14\}.$$

Примеры кодов приведены в таблице 2.3. Код во второй строке – код Голея. На этом примере мы видим, что оценка БЧХ на минимальное расстояние бывает неточной.

Показатели степеней корней	Порождающий многочлен	Размерность $n - \deg g(x)$	Расстояние	
			конструктивное	истинное
0	$M_0(x) = x + 1$	22	2	2
1,2,3,4,6,8,...	$M_1(x)$	12	5	7
...19,20,21,22,...	$M_5(x)$	12	5	7
0,1,2,3,4,6,...	$M_0(x)M_1(x)$	11	6	8
1,...,23	$M_1(x)M_5(x)$	1	23	23

Таблица 2.3: Примеры БЧХ-кодов длины 23

Итак, для получения параметров БЧХ кода достаточно знания структуры циклотомических классов поля из которого выбираются корни порождающего многочлена.

Для нахождения порождающего многочлена нужно знать минимальные многочлены корней. Один из способов их получения:

$$M_i(x) = \prod_{j \in C_i} (x - \alpha^j),$$

где произведение вычисляется по всем элементам циклотомического класса.

Пример 2.4. *Минимальный многочлен элемента α^5 в поле $GF(4)$ заданном примитивным полиномом $p(x) = 1 + x + x^4$, равен*

$$\begin{aligned} &M_5(x) = (x + \alpha^5)(x + \alpha^{10}) = \\ &= x^2 + (\alpha^5 + \alpha^{10})x + \alpha^0 = \\ &= x^2 + x + 1 \end{aligned}$$

При выполнении этих вычислений использована таблица 2.1. Согласно этой таблице двоичными представлениями α^5 и α^{10} служат (0110) и (0111). Сумма этих элементов равна единице.

Для построения двоичных БЧХ кодов нужны примитивные многочлены, задающие поля $GF(2^m)$. Примеры таких многочленов представлены в таблице 2.4. В таблице указаны степени, входящие в многочлен с ненулевыми коэффициентами.

Для построения непримитивных колов БЧХ полезны разложения чисел вида $2^m - 1$ на простые сомножители. Разложения для относительно малых значений m приведены в таблице 2.5.

2.3 Коды Рида-Соломона

Хотя это не подчеркивалось особо, при изучении кодов БЧХ мы, в основном, интересовались двоичными кодами. По крайней мере, все коды в примерах были двоичными. Вся теория, на которой базируются БЧХ коды, в равной степени верна и для недвоичных кодов. Недвоичные коды могут быть полезны в системах связи с недвоичной модуляцией. В этом смысле среди недвоичных кодов особенно интересны коды, алфавиты которых – расширения двоичного поля.

Очевидно, корректирующая способность недвоичных кодов при одинаковом числе исправляемых ошибок несколько лучше, чем для двоичных кодов: если несколько двоичных ошибок приходятся на один недвоичный символ недвоичного кода, они рассматриваются

Степень	Полином	Степень	Полином
2	2,1,0	16	16,12,3,1,0
3	3,1,0	17	17,3,0
4	4,1,0	18	18,7,0
5	5,2,0	19	19,5,2,1,0
6	6,1, 0	20	20,3,0
7	7,3,0	21	21,2,0
8	8,4, 3, 2,0	22	22,1,0
9	9,4,0	23	23,5,0
10	10,3,0	24	24,7,2,1,0
11	11,2,0	25	25,3,0
12	12,6,4,1,0	26	26,6,2,1,0
13	13,4,3,1,0	27	27,5,2,1,0
14	14,10,6,1,0	28	28,3,0
15	15,1,0	29	29,2,0

Таблица 2.4: Двоичные примитивные многочлены

как одна ошибка. Это означает, что код над полем $GF(2^m)$, исправляющий t ошибок, в действительности может исправить t «синхронизированных» пакетов ошибок длины m каждый, т.е. вплоть до tm ошибок. Во многих системах передачи и хранения информации ошибки группируются, и поэтому коды над большими алфавитами могут оказаться весьма практичными.

По сути, РС-коды – частный случай БЧХ-кодов. В соответствии с (2.11), избыточность БЧХ-кода тем меньше, чем меньше степень минимальных многочленов, образующих порождающий многочлен циклического кода. Степень не может быть меньше 1, и она равна единице, если формальная переменная x принимает значения в том же поле, что и поле $GF(2^m)$, использованное для построения проверочной матрицы кода. Действительно, в таком поле минимальным многочленом элемента β служит $x - \beta$. Поскольку степеням переменной x соответствуют позиции кодового слова, длина кода должна быть не меньше числа элементов мультипликативной группы поля.

Определение 2.2. *Кодом Рида-Соломона (РС-кодом) над* $GF(q)$

Таблица 2.5: Разложения чисел $2^m - 1$ на простые сомножители

m	Разложение	m	Разложение
3	7	14	3×43×127
4	3×5	15	7×31×151
5	31	16	3×5×17×257
6	3×3×7	17	131071
7	127	18	$3^3 \times 7 \times 19 \times 73$
8	3×5×17	19	524287
9	7×73	20	3×5×5×11×31×41
10	3×11×31	21	7×7×127×337
11	23×89	22	3×23×89×683
12	3×3×5×7×13	23	47×178841
13	8191	24	3×3×5×7×13×17×241

называется код БЧХ длины $n = q - 1$.

Из определения следует, что порождающий многочлен кода Рида-Соломона с конструктивным расстоянием d имеет вид

$$g(x) = (x - \alpha^b)(x - \alpha^{b+1}) \times \cdots \times (x - \alpha^{b+d-2}). \tag{2.14}$$

Чаще всего полагают $b = 1$, но другие значения b также используются и, более того, иногда приводят к более простым схемам кодирования.

Из (2.14) и теорем 2.1 и 2.2 следует, что параметры РС-кода связаны соотношением

$$d - 1 = n - k, \tag{2.15}$$

т.е. удовлетворяют границе Синглтона. Поскольку граница Синглтона одновременно является верхней границей на минимальное расстояние кодов, РС-коды являются оптимальными в смысле минимального расстояния, их еще называют кодами с максимально достижимым расстоянием (МДР-кодами).

Пример 2.5. *Рассмотрим код длины* $n = 15$ *над полем* $GF(2^4)$. *Для построения кода с минимальным расстоянием 5 положим* $b = 1$ *и тогда*

$g(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4) = x^4+\alpha^{13}x^3+\alpha^6x^2+\alpha^3x+\alpha^{10}$.

Имеем (15,11)-код с $d = 5$. Напомним, что двоичный код с таким же расстоянием имел параметры (15,7). Если записывать информационные и кодовые символы $GF(2^4)$ в двоичном виде, то получим линейный код (60, 44) с расстоянием 5. Как двоичный код этот код не оптимален, т.к. для кода с такой длиной и размерностью достижимое расстояние точно неизвестно, но находится в интервале от 6 до 7.

Одно из основных применений РС-кодов – использование в качестве компонентных кодов в каскадных конструкциях, которые будут рассматриваться позже (см. задачу 7.

При заданном размера алфавита желательно иметь код как можно большей размерности. В этом смысле полезны расширенные коды Рида-Соломона длины $n = q + 1$ размерности k и расстоянием $d = n - k + 1$. Проверочная матрица такого кода может быть представлена, в частности, в виде

$$H = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 & 0 \\ 0 & \alpha & \alpha^2 & \cdots & \alpha^{(q-1)} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \alpha^{q-k} & \alpha^{2(q-k)} & \cdots & \alpha^{(q-k)(q-1)} & 1 \end{pmatrix}.$$

Легко убедиться в том, что любые $n-k$ столбцов этой матрицы линейно независимы.

Задачи

1. Постройте матрицу вида (2.3), но с возведением элементов первой строки не в третью, а во вторую степень. Определите минимальное расстояние кода.

2. Напишите программы для построения поля, нахождения минимальных многочленов, нахождения корней уравнений перебором по элементам поля. Напишите программу удобного

Матлаб-калькулятора для выполнения арифметических операций в конечном поле. Этот калькулятор пригодится при решении задач на декодирование БЧХ и РС-кодов.

3. Указать параметры БЧХ-кодов длины 31 и 63. Сравните параметры с параметрами лучших известных линейных кодов.

4. Укажите параметры непримитивных БЧХ-кодов длины 17 и 21. Сравните их с параметрами лучших известных линейных кодов.

5. Построить порождающий многочлен кода длины 31, исправляющего 2-кратные ошибки.

6. Код длины 31, исправляющий 2-кратные ошибки использован для передачи сообщений и на выходе двоичного канала связи наблюдается последовательность y =403415447 (это двоичная последовательность, записанная в восьмеричной форме). Определите переданное сообщение.

7. Каскадные коды. Построим два кода:

   - (N, K)-код Рида-Соломона с минимальным расстоянием D над полем $GF(2^k)$ (внутренний код);
   - двоичный (n, k)-код с минимальным расстоянием d (внешний код).

   Кодирование выполняется в два этапа. Последовательность из kK информационных символов разбивается в блоки по k бит, каждый блок интерпретируется как элемент поля $GF(2^k)$ и кодируется внутренним кодом. Затем N кодовых символов этого кода интерпретируются как двоичные последовательности длины k и кодируются внешним кодом. Полученная на выходе кодера последовательность представляет собой кодовое слово каскадного кода.

   - Найдите параметры (длину, скорость и расстояние) каскадного кода.
   - Является ли код линейным?

- По аналогии с декодированием итеративных кодов (см. задачу... главы...) предложите процедуры декодирования каскадного кода.
- Сравните каскадные и итеративные коды по скорости, расстоянию, сложности построения, сложности декодирования.

Глава 3

Декодирование БЧХ- и РС-кодов

В предыдущем параграфе на примере кодов, исправляющих двукратные ошибки, мы рассмотрели одну из возможных процедур декодирования. Непосредственно из структуры проверочной матрицы видно, что задача декодирования сводится к решению системы нелинейных уравнений относительно позиций ошибок (точнее, элементов поля, соответствующих позициям ошибок). Огромное практическое значение кодов БЧХ и РС послужило стимулом к поиску более эффективных алгоритмов. Одно из первых решений задачи – алгоритм Питерсона (1960) и Горенстейна-Цирлера (1961), который сводит задачу исправления ошибок к решению системы из линейных уравнений. Сложность такого декодирования пропорциональна t^3.

Берлекэмп (1968), используя особенности матрицы коэффициентов системы уравнений, уменьшил сложность решения до величины порядка t^2. Месси (1969) сумел интерпретировать алгоритм Берлекэмпа как алгоритм построения рекуррентного фильтра, тем самым упростив и понимание алгоритма, и его реализацию. В окончательном виде этот алгоритм получил название алгоритма Берлекэмпа-Месси. Дальнейшее упрощение декодирования связано с применением быстрых преобразованием Фурье над конечными полями. Основные исследования в этой области были выполнены

Блэйхутом (1981). В результате, достигнута асимптотическая сложность декодирования кода длины n пропорциональная $n \log n$. Тем не менее, на практике, при реализации декодирования на БИС, чаще всего применяется алгоритм Берлекэмпа-Месси. В данном параграфе рассматриваются алгоритм Питерсона-Горенстейна-Цирлера (ПГЦ) и алгоритм Берлекэмпа-Месси (БМ).

3.1 Алгоритм Питерсона-Горенстейна-Цирлера

При описании декодирования кодов БЧХ и РС нет необходимости различать эти два класса кодов. Достаточно рассматривать БЧХ коды над произвольным полем $GF(q)$. Объяснения будут немного проще, если мы предположим, что последовательными корнями порождающего многочлена являются $\alpha, \alpha^2, ..., \alpha^{d-1}$, где α – примитивный элемент поля. Через d мы обозначили конструктивное расстояние кода, ему соответствует кратность исправляемых ошибок $\lfloor (d-1)/2 \rfloor$.

Переданное кодовое слово $c(x)$, вектор ошибок $e(x)$ и принятую из канала последовательность $v(x)$ запишем в виде

$$\begin{aligned} c(x) &= c_0 + c_1 x + ... + c_{n-1} x^{n-1} \\ e(x) &= e_0 + e_1 x + ... + e_{n-1} x^{n-1} \\ v(x) &= c(x) + e(x) = v_0 + v_1 x + ... + v_{n-1} x^{n-1} \end{aligned}$$

Поскольку $c(\alpha^j) = 0$ при $j = 1, ..., d-1$, для соответствующих компонент синдрома имеем

$$S_j = v(\alpha^j) = e(\alpha^j).$$

Предположим, что число ошибок (вес вектора ошибок) $\nu < t$ и что ошибки произошли на позициях с номерами $i_1, i_2, ..., i_\nu$. Тогда

$$S_j = e_{i_1} \alpha^{i_1 j} + e_{i_2} \alpha^{i_2 j} + \cdots + e_{i_\nu} \alpha^{i_\nu j},$$

где e_{i_h} обозначает значение ошибки на позиции h. Запись упростится, если использовать обозначения $Y_h = e_{i_h}$, $X_h = \alpha^{i_h}$. Получаем

систему уравнений

$$S_j = Y_1 X_1^j + Y_2 X_2^j + \cdots + Y_\nu X_\nu^j, \quad j = 1, ..., d-1. \tag{3.1}$$

Величины Y_h и X_h называют соответственно *значениями* и *локаторами* ошибок. При $\nu < t$ комбинация ошибок может быть исправлена, система имеет единственное решение, но найти его непросто, поскольку система нелинейна и число неизвестных велико. Ключ к решению задачи – в замене переменных: вместо X_h мы будем искать коэффициенты полинома, корнями которого служат X_h^{-1} . Этот полином называется *полиномом локаторов ошибок*. Итак, введем полином

$$\Lambda(x) = \prod_{j=1}^{\nu}(1 - xX_j) = 1 + \Lambda_1 x + ... + \Lambda_\nu x^n u. \tag{3.2}$$

Поясним дальнейшие выкладки на примере случая $\nu = 3$. Из (3.1) имеем

$$\begin{cases} Y_1 X_1 + Y_2 X_2 + Y_3 X_3 = S_1 \\ Y_1 X_1^2 + Y_2 X_2^2 + Y_3 X_3^2 = S_2 \\ Y_1 X_1^3 + Y_2 X_2^3 + Y_3 X_3^3 = S_3 \end{cases} \tag{3.3}$$

а из (3.2) последовательной подстановкой $x = X_1^{-1}$, $x = X_2^{-1}$ и $x = X_3^{-1}$, получаем

$$\begin{cases} X_1^3 + \Lambda_1 X_1^2 + \Lambda_2 X_1 + \Lambda_3 = 0 \\ X_2^3 + \Lambda_1 X_2^2 + \Lambda_2 X_2 + \Lambda_3 = 0 \\ X_3^3 + \Lambda_1 X_3^2 + \Lambda_2 X_3 + \Lambda_3 = 0 \end{cases} \tag{3.4}$$

Умножая эти уравнения, соответственно, на $X_1 Y_1$, $X_2 Y_2$ и $X_3 Y_3$, и суммируя, с учетом (3.3) получаем уравнение

$$S_4 + \Lambda_1 S_3 + \Lambda_2 S_2 + \Lambda_3 S_1 = 0.$$

Еще два уравнения получим, умножая (3.4) на $X_i^2 Y_1$, $i = 1, 2, 3$, а потом на $X_i^3 Y_1$, $i = 1, 2, 3$. Получим систему линейных уравнений, которая в матричной форме может быть записана как

$$\begin{pmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{pmatrix} \begin{pmatrix} \Lambda_1 \\ \Lambda_2 \\ \Lambda_3 \end{pmatrix} = \begin{pmatrix} -S_1 \\ -S_2 \\ -S_3 \end{pmatrix} \tag{3.5}$$

Контуры будущего алгоритма становятся понятными: из (3.5) находим коэффициенты полинома локаторов ошибок, потом из (3.2) – его корни, т.е. локаторы ошибок. После этого из (3.3) найдем значения ошибок. Чтобы окончательно сформулировать алгоритм, введем обозначения

$$\mathbf{M}_\mu = \begin{pmatrix} S_1 & S_2 & \cdots & S_\mu \\ S_2 & S_3 & \cdots & S_{\mu+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_\mu & S_{\mu+1} & \vdots & S_{2\mu-1} \end{pmatrix} \tag{3.6}$$

$$\mathbf{L}_\mu = \begin{pmatrix} X_1 & X_2 & \cdots & X_\mu \\ X_1^2 & X_2^2 & \cdots & X_\mu^2 \\ \vdots & \vdots & \ddots & \vdots \\ X_1^\mu & X_2^\mu & \vdots & X_\mu^\mu \end{pmatrix} \tag{3.7}$$

Формулы (3.3) и (3.5) в общем случае имеют вид

$$\mathbf{L}_\mu \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_\nu \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_\nu \end{pmatrix} \tag{3.8}$$

$$\mathbf{M}_\mu \begin{pmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \vdots \\ \Lambda_1 \end{pmatrix} = \begin{pmatrix} -S_{\nu+1} \\ -S_{\nu+2} \\ \vdots \\ -S_{2\nu} \end{pmatrix} \tag{3.9}$$

Теорема 3.1. *Матрицы $\mathbf{M}_\mu$, $\mathbf{L}_\mu$ невырождены, если μ равно числу ошибок ν и матрица $\mathbf{M}_\mu$ вырождена, если $\nu < \mu$.*

Доказательство. Докажем теорему на примере случая $\mu = 3$. Положим сначала $\nu = \mu = 3$ и запишем $\mathbf{M}_3$ в виде

$$\mathbf{M}_3 = \begin{pmatrix} 1 & 1 & 1 \\ X_1 & X_2 & X_3 \\ X_1^2 & X_2^2 & X_3^2 \end{pmatrix} \begin{pmatrix} Y_1X_1 & 0 & 0 \\ 0 & Y_2X_2 & 0 \\ 0 & 0 & Y_3X_3 \end{pmatrix} \begin{pmatrix} 1 & X_1 & X_1^2 \\ 1 & X_2 & X_2^2 \\ 1 & X_3 & X_3^2 \end{pmatrix} \tag{3.10}$$

Матрица $\mathbf{M}_3$ невырождена, поскольку является произведением трех невырожденных матриц (две из них – матрицы Вандермонда). Определитель матрицы $\mathbf{L}_3$ можно также выразить через определитель Вандермонда и эта матрица также невырождена.

Допустим теперь, что $\mu = 3$, но $\nu = 2$. При этом разложение (3.10) остается в силе, если положить $Y_3 = 0$. Очевидно, определитель матрицы $\mathbf{M}_3$ станет равным нулю, т.к. центральная матрица в (3.10) окажется вырожденной. □

Формальное описание алгоритма приведено на рис. 3.1.

Пример 3.1. *Рассмотрим РС-код (15,9) над полем* $GF(2^4)$. *Поле задано примитивным многочленом* $p(x) = 1+x+x^4$, *список элементов поля приведен в таблице 2.1. Пример декодирования со всеми промежуточными вычислениями представлен в таблице 3.1.*

3.2 Алгоритм Берлекэмпа-Месси

Алгоритм ПГЦ сводит задачу нахождения позиций и значений ν ошибок к решению двух систем линейных уравнений порядка ν. Для решения можно воспользоваться, например, методом Гаусса, и тогда сложность вычислений будет иметь порядок ν^3. Можно заметить, что в (3.5) и (3.9) мы имеем дело с весьма специфической матрицей коэффициентов: строки являются сдвигами предыдущих строк. Такие матрицы (матрицы с постоянными диагоналями) называются теплицевыми. Для них существуют более простые способы обращения, чем для матриц общего вида. Рассматриваемый ниже алгоритм БМ использует сходство структуры матрицы коэффициентов со структурой, которая часто встречается во многих приложениях, в частности, в задачах цифровой обработки сигналов.

Вернемся к системе уравнений (3.5). Каждое из уравнений этой системы получается увеличением на единицу индексов при коэффициентах предыдущего уравнения. Уравнение с номером r имеет

Исправление $t = \lfloor (d-1)/2 \rfloor$ ошибок
Input: Выход канала $v(x)$
Output: Кодовое слово $c(x)$

1. Вычислить компоненты синдрома
for $j = 1$ **to** $d-1$ **do**
$S_j = v(\alpha^j)$
end

2. Положить $\nu = t$

3. Вычислить определитель системы уравнений

$$D = \det(\mathbf{M}_\nu)$$

if $D == 0$ **then** $\nu \leftarrow \nu - 1$ **Goto** Step 3.
else

4. Вычислить коэффициенты полинома локаторов ошибок как решение системы (3.9)

5. Найти локаторы ошибок как величины обратные корням $\Lambda(x)$

6. Найти значения ошибок как решение системы (3.8)

7. Исправить ошибки
$c(x) = v(x) - e(x)$

Рис. 3.1: Алгоритм ПГЦ.

Порождающий многочлен $g(x) = \alpha^6 + \alpha^9 x + \alpha^6 x^2 + \alpha^4 x^3 + \alpha^{14} x^4 + \alpha^{10} x^5 + x^6$	
Кодовое слово $c(x) = \alpha + \alpha^3 x + \alpha^{10} x^2 + \alpha^{12} x^3 + \alpha^{13} x^4 + \alpha^3 x^5 + \alpha^9 x^6 + \alpha^5 x^7 +$ $+ x^8 + \alpha^{10} x^9 + \alpha^8 x^{10} + \alpha x^{11} + \alpha x^{12} + \alpha^{13} x^{13} + \alpha^2 x^{14}$	
Выход канала $v(x) = \alpha + \alpha^3 x + \alpha^7 x^2 + \alpha^{12} x^3 + \alpha^{13} x^4 + \alpha^{14} x^5 + \alpha^9 x^6 + \alpha^5 x^7 +$ $+ x^8 + \alpha^{10} x^9 + \alpha^8 x^{10} + \alpha x^{11} + \alpha^{10} x^{12} + \alpha^{13} x^{13} + \alpha^2 x^{14}$	
Синдромный многочлен	$S(x) = \alpha^8 x + \alpha^2 x^2 + \alpha^{10} x^3 + x^4 + \alpha^{12} x^6$
Система уравнений для коэффициентов многочлена локаторов ошибок	$\begin{pmatrix} \alpha^8 & \alpha^2 & \alpha^{10} \\ \alpha^2 & \alpha^{10} & \alpha \\ \alpha^{10} & \alpha & 0 \end{pmatrix} \begin{pmatrix} \Lambda_3 \\ \Lambda_2 \\ \Lambda_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \alpha^{12} \end{pmatrix}$
Многочлен локаторов ошибок	$\Lambda(x) = 1 + \alpha^{13} x + \alpha^5 x^2 + \alpha^4 x^3$
Локаторы ошибок	$\alpha^2, \alpha^5, \alpha^{12}$
Система уравнений для значений ошибок	$\begin{pmatrix} \alpha^2 & \alpha^5 & \alpha^{12} \\ \alpha^4 & \alpha^{10} & \alpha^9 \\ \alpha^6 & 1 & \alpha^6 \end{pmatrix} \begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \end{pmatrix} = \begin{pmatrix} \alpha^8 \\ \alpha^2 \\ \alpha^{10} \end{pmatrix}$
Значения ошибок	$\alpha^6, 1, \alpha^8$
Вектор ошибок	$e(x) = \alpha^6 x^2 + x^5 + \alpha^8 x^{12}$

Таблица 3.1: Декодирование РС-кода с исправлением 3 ошибок с помощью алгоритма Питерсона-Горенстейна-Цирлера

вид:

$$S_r = -\sum_{j=1}^{\nu} \Lambda_j S_{r-j} = -\Lambda_1 S_{r-1} - \Lambda_2 S_{r-2} - ... - \Lambda_\nu S_{r-\nu}. \qquad (3.11)$$

Можно интерпретировать это соотношение как предсказание величины следующей компоненты синдрома по предыдущим. Если коэффициенты предсказания $\Lambda_1, \Lambda_2, ..., \Lambda_\nu$ известны, то по ним последовательным применением рекуррентного соотношения (3.11) из компонент синдрома $S_1, ..., S_\nu$ получаются компоненты $S_{\nu+1}, ..., S_{2\nu}$. Устройство, реализующее (3.11), в цифровой обработке сигналов называют рекурсивным фильтром. Таким образом, задача вычисления коэффициентов многочлена локаторов ошибок сводится к задаче построения кратчайшего рекурсивного фильтра по его выходной последовательности. Фильтр, соответствующий уравнению (3.11), показан на рис. 3.2. Отличие рассматриваемой задачи от обычной задачи построения рекурсивного фильтра состоит в том, что коэффициенты фильтра и наблюдаемая последовательность принимают значения в конечном поле.

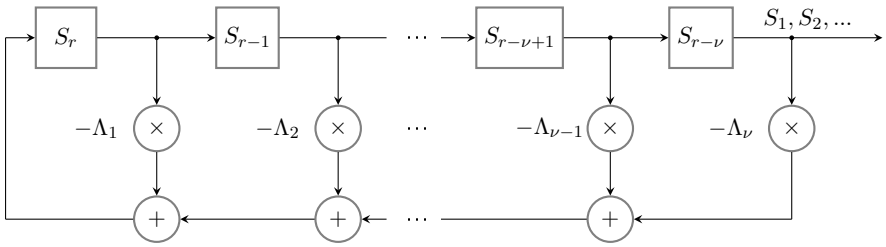

Рис. 3.2: Интерпретация вычисления многочлена локаторов ошибок как построения БИХ-фильтра

Для построения фильтра надо найти его длину L и коэффициенты полинома $\Lambda(x)$. Фильтр строится шаг за шагом. Начиная с $r = 1$, строим фильтр, который порождает последовательность $S_1, ..., S_\nu$. На r-й итерации, имея уже построенный на предыдущий итерации фильтр $\Lambda^{(r-1)}(x)$, вычисляем очередное значение

$$\hat{S}_r = -\sum_{j=1}^{r-1} \Lambda_j^{(r-1)} S_{r-j}.$$

(для упрощения записи мы игнорируем тот факт, что порядок фильтра, порождающего последовательность длины r, может быть меньше r). Это предсказанное значение $\hat{S}_r$ может не совпадать с требуемым значением S_r. В этом случае вычисляется так называемая *невязка* (ошибка предсказания)

$$\Delta_r = S_r - \hat{S}_r = S_r + \sum_{j=1}^{r-1} \Lambda_j^{(r-1)} S_{r-j} = \sum_{j=0}^{r-1} \Lambda_j^{(r-1)} S_{r-j}.$$

Если невязка равна нулю, итерация выполнена успешно, $\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x)$. Если же невязка не равна нулю, то ранее вычисленные коэффициенты должны быть подправлены, чтобы сделать ее нулевой, сохранив нулевыми все предыдущие невязки. Положим

$$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x) + Ax^{r-m}\Lambda^{(m-1)}(x) \tag{3.12}$$

где A – элемент поля, а $\Lambda^{(m-1)}(x)$ – один из многочленов, полученных на предыдущих итерациях. Параметры A и m нужно подобрать так, чтобы невязка стала равной нулю.

Выберем $m < r$ таким, что $\Delta_m = \sum_{j=0}^{r-1} \Lambda_j^{(m-1)} S_{r-j-l} \neq 0$.

Заметим, что умножение на x^{r-m} в (3.12) соответствует сдвигу содержимого регистра сдвига "вперед " на $r - m$ тактов. Поэтому в фильтр, $x^{r-m}\Lambda^{(m-1)}(x)$, соответствующий второму слагаемому в (3.12), до такта с номером $r - 1$ получает на вход те же значения, что поступали на фильтр $\Lambda^{(m-1)}(x)$ до такта $m - 1$. Невязка на этих шагах была нулевой, следовательно, вклад второго слагаемого до такта с номером $r-1$ равен нулю, а на такте r вклад равен $A\Delta_m$. Положим $l = r - m$, $A = -\Delta_r/\Delta_m$.

Тогда новая невязка равна

$$\Delta'_r = \Delta_r - \frac{\Delta_r}{\Delta_m}\Delta_m = 0.$$

Тем самым мы убедились в том, что модифицированный в соответствии с (3.12) фильтр порождает правильную последовательность синдромов.

Подсчитаем длину L_r фильтра Λ_r, получаемого на шаге r. Эта длина, конечно, зависит от выбора параметра m. Для построения

фильтра, порождающего нужную последовательность, можно выбрать любое m, при котором $\Delta_m \neq 0$, но длина фильтра зависит от m, а решаемая задача – построение фильтра минимально возможной длины (длина фильтра равна весу предполагаемой комбинации ошибок) .

Порядок L_r фильтра Λ_r равен степени многочлена в правой части (3.12), т.е.

$$L_r = \max\{L_{r-1}, r - m + L_{m-1}\} \tag{3.13}$$

Отсюда видим, что порядок фильтра либо не меняется, либо растет от шага к шагу. Выберем m равным номеру последнего шага, на котором увеличился порядок фильтра. Следующая лемма показывает, насколько может увеличиться порядок фильтра за одну итерацию.

Лемма 3.2. *Пусть на шаге r невязка $\Delta_r \neq 0$, и предыдущее увеличение порядка фильтра произошло на шаге m при невязке $\Delta_m \neq 0$. Тогда фильтр, порождающий $S_1, S_2, ..., S_r$ задается полиномом*

$$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x) - \frac{\Delta_r}{\Delta_m} x^{r-m} \Lambda^{(m-1)}(x) \tag{3.14}$$

и имеет порядок

$$L_r = \max\{L_{r-1}, r - L_{r-1}\} \tag{3.15}$$

Доказательство. Тождество (3.14) уже доказано, осталось доказать (3.15). Для этого воспользуемся индукцией по r, причем интересны только те шаги, на которых увеличивается порядок фильтра.

Положив в качестве начального значения $L_0 = 0$, при появлении первого ненулевого элемента S_j в последовательности $S_1, S_2, ..., S_j, ...$ порядок фильтра, порождающего $0, 0, ..., S_j$ равен j (см. лемму 3.11 в Приложении), т.е. удовлетворяет (3.15). Тем самым, мы проверили справедливость леммы для начального шага индукции.

Предположим теперь что, на всяком на шаге j, когда увеличивалась длина фильтра, имело место равенство $L_j = j - L_{j-1}$. В

частности, на шаге с номером $j = m$ длина фильтра увеличилась и, по предположению, леммы стала равной $L_m = L_{r-1}$, следовательно,

$$L_m = m - L_{m-1} = L_{r-1}$$

Подстановка этого соотношения в (3.13) приводит к (3.15). □

Формальное описание алгоритма, который вытекает из доказанных утверждений, сформулируем в виде теоремы.

Теорема 3.3. Алгоритм Берлекэмпа-Месси. *Пусть заданы* $S_1, S_2, ..., S_{2t}$ *и начальные условия* $\Lambda^{(0)}(x) = 1$,$B^{(0)}(x) = 1$, $L_0 = 0$. *При* $r = 1, 2, ..., 2t$ *рекуррентно вычислим*

$$\Delta_r = \sum_{j=0}^{r-1} \Lambda_j^{(r-1)} S_{r-j}, \tag{3.16}$$

$$L_r = \delta_r(r - L_{r-1}) + (1 - \delta_r)L_{r-1}, \tag{3.17}$$

$$\begin{pmatrix} \Lambda^{(r)}(x) \\ B^{(r)}(x) \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_r x \\ \Delta_r^{-`1}\delta_r & (1-\delta_r)x \end{pmatrix} \begin{pmatrix} \Lambda^{(r-1)}(x) \\ B^{(r-1)}(x) \end{pmatrix}, \tag{3.18}$$

где $\delta_r = 1$, *если одновременно* $\Delta_r \neq 0$ *и* $2L_{r-1} \leq r - 1$, *и* $\delta_r = 0$ *в противном случае. Тогда* $\Delta^{(2t)}(x)$ *является многочленом наименьшей степени, коэффициенты которого удовлетворяют равенствам* $\Lambda_0^{(2t)} = 1$ *и*

$$S_r + \sum_{j=1}^{r-1} \Lambda_j^{(2t)} S_{r-j} = 0, \quad r = L_{2t} + 1, ..., 2t.$$

Обсуждение. В вычислениях по формуле (3.18) присутствует обратный элемент к Δ_r, который не определен при $\Delta_r = 0$. Однако, этот обратный элемент востребован только при $\delta_r = 0$. В этом случае принимаем $\Delta_r^{-1}\delta_r = 0$. Заметим также, что переменная δ_r играет роль переключателя, в зависимости от ее значения изменяется способ вычисления "модифицирующего многочлена" $B^{(r)}(x)$, который, по сути, представляет собой второе слагаемое в (3.14). Заметим, что выражение $\delta^{-}1_m\Lambda^(m-1)(x)$ достаточно пересчитывать только при изменении длины регистра, а сам регистр модифицируется всякий раз, когда невязка отлична от нуля.

Псевдокод алгоритма Берлекэмпа-Месси приведен на рис. 3.3. Длина регистра L неявно участвует во всех операциях с полиномами, т.к. определяет число нетривиальных элементов в соответствующих массивах.

Доказательство. Все соотношения в формулировке теоремы уже доказаны, за исключением того, что построенный ЛРОС является кратчайшим среди всех ЛРОС, порождающих заданную последовательность синдромных компонент. В Приложении мы обсудим подробнее задачу построения регистра, порождающего заданную последовательность и докажем оптимальность алгоритма БМ. □

3.3 Алгоритм Форни

В предыдущем параграфе мы рассмотрели быстрый алгоритм нахождения локаторов ошибок. Быстрым он является в том смысле, что решает задачу нахождения решения системы из t линейных уравнений за число операций порядка t^2 , вместо числа t^3, которое потребовалось бы для системы уравнений общего вида. Чтобы декодирование в целом имело сложность порядка t^2, нужно упростить решение системы уравнений (3.8) для нахождения значений ошибок по известным локаторам. Эта задача была решена Форни (1965). Для описания алгоритма введем многочлен значений ошибок

$$\Omega(x) = S(x)\Lambda(x) \qquad \bmod x^{2t} \tag{3.19}$$

Теорема 3.4. *Имеет место соотношение*

$$\Omega(x) = \sum_{i=1}^{\nu} Y_i X_i \prod_{j \neq i} (1 - X_j x). \tag{3.20}$$

Вычисление коэффициентов многочлена локаторов ошибок
по алгоритму Берлекэмпа-Месси
Input: Синдромный многочлен $S(x)$
Output: Многочлен локаторов $\Lambda(x)$

1. Инициализация
$L = 0$; % Текущая длина регистра
$\Lambda(x) = 1$; % Многочлен локаторов
$B(x) = 1$; % Многочлен компенсации невязки
2. Основной цикл
for $r = 1$ **to** $d - 1$ **do**
    $\Delta = \sum_{j=0}^{L} \Lambda_j S_{r-j}$; % Невязка
    $B(x) = xB(x)$; % Сдвиг
    if $\Delta \neq 0$ **then** % ЛРОС модифицируется
        $T(x) = \Lambda(x) - \Delta B(x)$; % Вспомогательный многочлен
        if $2L \leq r - 1$ **then** % Длина увеличивается
            $B(x) = \Delta^{-1}\Lambda(x)$;
            $L = r - L$;
        end;
        $\Lambda(x) = T(x)$;
    end;
end
3. Формирование результата
if $\deg \Lambda(x) == L$ **then**
    Многочлен локаторов = $\Lambda(x)$;
else
    Число ошибок больше t;
end;

Рис. 3.3: Алгоритм Берлекэмпа-Месси

Доказательство. Напомним:

$$\begin{aligned} S(x) &= \sum_{j=1}^{2t} S_j x^j = \sum_{j=1}^{2t}\sum_{j=1}^{\nu} Y_i X_i^j x^j, \\ \Lambda(x) &= \prod_{j=1}^{\nu}(1 - X_j x). \end{aligned}$$

Подставим эти выражения в (3.19). Получим

$$\Omega(x) = \left[\sum_{j=1}^{2t}\sum_{i=1}^{\nu} Y_i X_i^j x^j\right]\left[\prod_{j=1}^{\nu}(1 - X_j x)\right] \tag{3.21}$$

Следующий шаг использует тождество

$$\frac{1-a^n}{1-a} = 1 + a + ... + a^{n-1}.$$

При $a = X_i x$ и $n = 2t$ из него следует

$$\sum_{j=1}^{2t} X_i^j x^j = \frac{1 - X_i^{2t} x^{2t}}{1 - X_i x}.$$

Подставив в (3.21), получим

$$\Omega(x) = \sum_{i=1}^{\nu} Y_i X_i \left(1 - X_i^{2t} x^{2t}\right) \prod_{j \neq i} (1 - X_j x).$$

После приведения по модулю x^{2t} получаем требуемый результат. □

Теорема 3.5. *Алгоритм Форни.*

$$Y_i = \frac{X_i^{-1}\Omega(X_i^{-1})}{\prod_{j \neq i}\left(1 - X_j X_i^{-1}\right)} \tag{3.22}$$

Доказательство. В сумме в правой части (3.20) при подстановке $x = X_i^{-1}$ остается только одно ненулевое слагаемое. Поэтому из Теоремы 3.4 имеем

$$\Omega(x) = Y_i X_i \prod_{j \neq i} (1 - X_j x)\,.$$

Отсюда следует (3.22). □

Таким образом, Теорема 3.5 дает нам формулу для прямого вычисления значений ошибок. Сложность непосредственного вычисления каждого значения пропорциональна степени многочлена значений, следовательно, общая сложность алгоритма имеет порядок t^2.

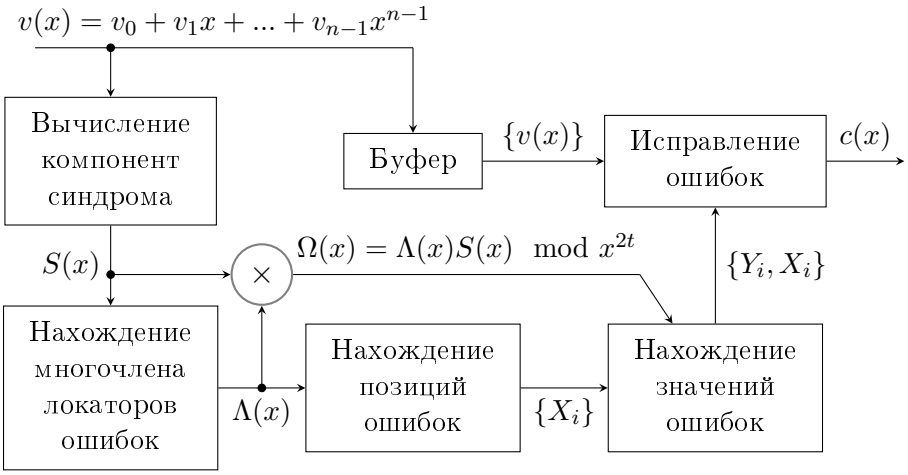

Рис. 3.4: Блок-схема декодера БЧХ-кода

Полная схема декодера БЧХ-кода приведена на рис. 3.4. Ее входом служит последовательность $v(x) = v_0 + v_1 x + ... + v_{n-1} x^{n-1}$ на выходе канала. Она хранится в буфере до тех пор, пока не будут выполнены основные этапы декодирования: вычисление компонент синдрома, многочлена локаторов ошибок, значений ошибок и локаторов ошибок. В случае двоичных кодов блок нахождения значений

ошибок не нужен. Выходом блока исправления ошибок служат информационные символы исправленного кодового слова, либо сигнал о неисправимой ошибке.

Блок-схема алгоритма работы модуля нахождения многочлена локаторов ошибок показана на рис. 3.4. Алгоритм в точности следует утверждению теоремы 3.3. Участвующий в теореме и алгоритме полином коррекции невязки $B(x)$ хранит последнюю ненулевую невязку, на каждом шаге "сдвигая" ее умножением на x. После того как снова появляется ненулевая невязка, содержимое $B(x)$ обновляется. Выходом алгоритма является либо вычисленный многочлен локаторов, либо сообщение о том, что число ошибок превышает корректирующую способность кода.

Упражнение 11. *Тот же код и тот же вектор ошибок, что и в примере 1, декодируем по алгоритму Берлекэмпа-Месси. Все промежуточные вычисления представлены в таблице 3.2.*

3.4 Исправление ошибок и стираний

Как было объяснено в главе 1, эффективность кодирования намного повышается при использовании в декодере информации о надежности символов. Простейшей формой использования такой информации является стирание ненадежных символов. Манипулируя порогом стирания и многократно декодируя при различных значениях порога, можно добиться эффективности декодирования близкой к декодированию по максимуму правдоподобия. Такое декодирование называют декодированием по минимуму обобщенного расстояния (Форни, 1966). Точная формулировка алгоритма будет приведена в разделе х.х.

Упражнение 12. *Докажите, что код с минимальным расстоянием d гарантировано исправляет комбинации из ν ошибок и f стираний при условии*

$$2\nu + f + 1 \leq d. \tag{3.23}$$

<table>
<tr><td colspan="5">Порождающий многочлен
$g(x) = \alpha^6 + \alpha^9 x + \alpha^6 x^2 + \alpha^4 x^3 + \alpha^{14} x^4 + \alpha^{10} x^5 + x^6$</td></tr>
<tr><td colspan="5">Кодовое слово
$c(x) = \alpha + \alpha^3 x + \alpha^{10} x^2 + \alpha^{12} x^3 + \alpha^{13} x^4 + \alpha^3 x^5 + \alpha^9 x^6 + \alpha^5 x^7 +$
$+ x^8 + \alpha^{10} x^9 + \alpha^8 x^{10} + \alpha x^{11} + \alpha x^{12} + \alpha^{13} x^{13} + \alpha^2 x^{14}$</td></tr>
<tr><td colspan="5">Выход канала
$v(x) = \alpha + \alpha^3 x + \alpha^7 x^2 + \alpha^{12} x^3 + \alpha^{13} x^4 + \alpha^{14} x^5 + \alpha^9 x^6 + \alpha^5 x^7 +$
$+ x^8 + \alpha^{10} x^9 + \alpha^8 x^{10} + \alpha x^{11} + \alpha^{10} x^{12} + \alpha^{13} x^{13} + \alpha^2 x^{14}$</td></tr>
<tr><td colspan="5">Синдромный многочлен $S(x) = \alpha^8 x + \alpha^2 x^2 + \alpha^{10} x^3 + x^4 + \alpha^{12} x^6$</td></tr>
<tr><td>r</td><td>Δ</td><td>$B(x)$</td><td>$\Lambda(x)$</td><td>L</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>α^8</td><td>α^7</td><td>$1 + \alpha^8 x$</td><td>1</td></tr>
<tr><td>2</td><td>α^5</td><td>$\alpha^7 x$</td><td>$1 + \alpha^9 x$</td><td>1</td></tr>
<tr><td>3</td><td>α^{14}</td><td>$\alpha + \alpha^{10} x$</td><td>$1 + \alpha^9 x + \alpha^6 x^2$</td><td>2</td></tr>
<tr><td>4</td><td>α^{10}</td><td>$\alpha x + \alpha^{10} x^2$</td><td>$1 + \alpha^2 x + \alpha^9 x^2$</td><td>2</td></tr>
<tr><td>5</td><td>α^{10}</td><td>$\alpha^5 + \alpha^7 x + \alpha^{14} x^2$</td><td>$1 + \alpha^2 x + \alpha^2 x^2 + \alpha^5 x^3$</td><td>3</td></tr>
<tr><td>6</td><td>α^9</td><td>$\alpha^5 x + \alpha^7 x^2 + \alpha^{14} x^3$</td><td>$1 + \alpha^{13} x + \alpha^5 x^2 + \alpha^4 x^3$</td><td>3</td></tr>
<tr><td colspan="5">Многочлен локаторов ошибок $\Lambda(x) = 1 + \alpha^{13} x + \alpha^5 x^2 + \alpha^4 x^3$</td></tr>
<tr><td colspan="5">Локаторы ошибок $\alpha^2, \alpha^5, \alpha^{12}$</td></tr>
<tr><td colspan="5">Многочлен значений ошибок
$S(x)\Lambda(x) = \alpha^8 + \alpha^3 x + \alpha^7 x^2 + \alpha^2 x^6 + \alpha^2 x^7 + \alpha x^8$
$\Omega(x) = S(x)\Lambda(x)(\mod x^6) = \alpha^8 + \alpha^3 x + \alpha^7 x^2$</td></tr>
<tr><td colspan="5">Значения ошибок $\alpha^6, 1, \alpha^8$</td></tr>
<tr><td colspan="5">Вектор ошибок $e(x) = \alpha^6 x^2 + x^5 + \alpha^8 x^{12}$</td></tr>
</table>

Таблица 3.2: Декодирование РС-кода с исправлением 3 ошибок с помощью алгоритма Берлекэмпа-Месси

Подсказка: нужно убедиться в том, что расстояние Хэмминга, подсчитанное по нестертым позициям, является метрикой, а затем воспользоваться неравенством треугольника.

В случае двоичных БЧХ кодов такая корректирующая способность достигается двукратным декодированием с исправлением только ошибок. Для этого при первом декодировании нужно положить все стертые позиции равными нулю, а при втором – единице. В одной из двух попыток декодирования на f стертых позициях будет не больше $f/2$ ошибок. Условием правильного декодирования будет $2(f/2+v)+1 \le d$, что эквивалентно (3.23). В случае недвоичных кодов БЧХ или кодов РС такой метод не годится, его непосредственное применение потребовало бы перебора по множеству значений ошибок на стертых позициях. К счастью, как мы увидим ниже, небольшая модификация описанных выше декодеров ПГЦ и БМ позволяет декодировать ошибки и стирания с вычислительной сложностью примерно того же порядка, что и декодирование только ошибок. Рассмотрим БЧХ код, заданный определением 2.1, при $b = 1$. Запишем кодовое слово, принятую последовательность и вектор ошибок в виде полиномов

$$\begin{aligned} c(x) &= c_0 + c_1 x + ... + c_{n-1} x^{n-1}, \\ v(x) &= v_0 + v_1 x + ... + v_{n-1} x^{n-1}, \\ e(x) &= e_0 + e_1 x + ... + e_{n-1} x^{n-1}. \end{aligned}$$

Поскольку позиции стираний известны, при вычислении позиций ошибок стертые позиции можно игнорировать (положить равными нулю). Соответствующие последовательности обозначаем как $c'(x)$, $v'(x)$ и $e'(x)$, а соответствующие компоненты (модифицированного) синдрома – как S'_i, $i = 1, 2, ..., d-1$. Модифицированным синдромом мы назвали синдром, вычисленный по нестертым позициям принятой последовательности.

Пусть $i_1, i_2, ..., i_f$ – номера позиций стираний, и $U_j = \alpha^{i_j}$, $j = 1, 2, ..., f$. По аналогии с полиномом локаторов ошибок $\Lambda(x)$ вводим в рассмотрение полином локаторов стираний

$$\Psi(x) = \prod_{j=1}^{f} (1 - xU_j). \tag{3.24}$$

Нам потребуется также полином локаторов ошибок и стираний, который вычисляется как

$$\tilde{\Lambda}(x) = \Psi(x)\Lambda(x). \tag{3.25}$$

Очевидно, зная $\tilde{\Lambda}(x)$, можно построить многочлен значений ошибок по формуле (3.19) и затем по алгоритму Форни найти значения ошибок. Многочлен локаторов стираний известен, следовательно, достаточно найти второй сомножитель в (3.25). Можно поступить иначе, предположив, что ошибок было $\nu + f$, но для f из них локаторы уже найдены. Тогда для нахождения оставшихся локаторов можно воспользоваться теоремой БМ, заменив начальные условия $\Lambda^{(0)}(x) = B^{(0)}(x) = 1$ на $\Lambda^{(0)}(x) = B^{(0)}(x) = \Psi(x)$. Заметим, что вычисление полинома локаторов стираний по формуле (3.24) также можно организовать в рекуррентной форме. Итоговый алгоритм приведен на рис. 3.5.

Упражнение 13. *Снова рассмотрим РС-код (15,9) над полем* $GF(2^4)$. *Пример декодирования комбинации ошибок кратности 2 при наличии двух стираний подробно разобран в таблице 3.3.*

3.5 Декодирование по минимуму обобщенного расстояния

В главах 1 и 4 мы не раз подчеркивали, что декодирование с мягкими решениями, т.е. с учетом оценок надежности принимаемых символов намного эффективнее декодирования с жесткими решениями. Очевидный недостаток рассмотренных выше алгоритмов декодирования БЧХ и РС-кодов состоит в том, что они не могут быть непосредственно применены для декодирования с мягкими решениями. Наиболее практичным среди известных методов учета надежности входных символов при сохранении преимуществ алгебраического декодирования кодов является декодирование по критерию минимума обобщенного расстояния [**?**].

```
Вычисление многочлена локаторов ошибок и стираний по
алгоритму Берлекэмпа-Месси
Input: Синдромный многочлен S(x);
Число стираний f;
Локаторы стираний U_1, ..., U_f;
Output: Многочлен локаторов Λ(x)
1. Инициализация
L = f; % Текущая длина регистра
Λ(x) = 1; % Многочлен локаторов
B(x) = 1; % Многочлен компенсации невязки
2. Построение полинома локаторов стираний Ψ(x)
for r = 1 to f do
    Λ(x) = Λ(x) − U_r x Λ(x)
end
B(x) = Λ(x);
for r = f + 1 to d − 1 do
    Δ = Σ_{j=0}^{L} Λ_j S_{r−j}; % Невязка
    B(x) = xB(x); % Сдвиг
    if Δ ≠ 0 then % ЛРОС модифицируется
        T(x) = Λ(x) − ΔB(x); % Вспомогательный многочлен
        if 2L ≤ r + f − 1 then % Длина увеличивается
            B(x) = Δ^{−1} Λ(x);
            L = r − L + f;
        end;
        Λ(x) = T(x);
    end;
end
4. Формирование результата
if deg Λ(x) == L then
    Многочлен локаторов ошибок и стираний = Λ(x);
else
    Отказ от декодирования;
end;
```

Рис. 3.5: Алгоритм Берлекэмпа-Месси для исправления ошибок и стираний

Порождающий многочлен $g(x) = \alpha^6 + \alpha^9 x + \alpha^6 x^2 + \alpha^4 x^3 + \alpha^{14} x^4 + \alpha^{10} x^5 + x^6$				
Кодовое слово $c(x) = \alpha + \alpha^3 x + \alpha^{10} x^2 + \alpha^{12} x^3 + \alpha^{13} x^4 + \alpha^3 x^5 + \alpha^9 x^6 + \alpha^5 x^7 +$ $+x^8 + \alpha^{10} x^9 + \alpha^8 x^{10} + \alpha x^{11} + \alpha x^{12} + \alpha^{13} x^{13} + \alpha^2 x^{14}$				
Выход канала $v(x) = \alpha + \alpha^3 x + \alpha^7 x^2 + \alpha^{12} x^3 + \emptyset x^4 + \alpha^{14} x^5 + \alpha^9 x^6 + \alpha^4 x^7 +$ Символ $\emptyset$ обозначает стирание $+x^8 + \alpha^{10} x^9 + \emptyset x^{10} + \alpha x^{11} + \alpha x^{12} + \alpha^{13} x^{13} + \alpha^2 x^{14}$				
Локаторы стираний α^4, α^{10}				
Синдромный многочлен $S(x) = \alpha^{12} x + x^2 + \alpha^6 x^3 + \alpha^{11} x^4 + \alpha^{10} x^6$				
Полином локаторов стираний $\Psi(x) = (1 + \alpha^4 x)(1 + \alpha^{10} x) = 1 + \alpha^2 x + \alpha^{14} x^2$				
r	Δ	$B(x)$	$\Lambda(x)$	L
0	0	1	1	0
1	0	$1 + \alpha^4 x$	$1 + \alpha^4 x$	1
2	0	$1 + \alpha^2 x + \alpha^{14} x^2$	$\Psi(x) = 1 + \alpha^2 x + \alpha^{14} x^2$	2
3	α^5	$\alpha^{10} + \alpha^{12} x + \alpha^9 x^2$	$1 + \alpha x + \alpha x^2 + \alpha^4 x^3$	3
4	α^8	$\alpha^{10} x + \alpha^{12} x^2 + \alpha^9 x^3$	$1 + \alpha^9 x + \alpha^2 x^2 + \alpha^{10} x^3$	3
5	α^2	$\alpha^{13} + \alpha^7 x + x^2 + \alpha^8 x^3$	$1 + \alpha^9 x + \alpha^7 x^2 + \alpha^{11} x^3 + \alpha^{11} x^4$	4
6	α^{10}	$\alpha^{13} x + \alpha^7 x^2 + x^3 + \alpha^8 x^4$	$1 + \alpha^5 x + \alpha^9 x^2 + \alpha^7 x^3 + \alpha^6 x^4$	4
Многочлен локаторов ошибок и стираний $\Lambda(x) = 1 + \alpha^5 x + \alpha^9 x^2 + \alpha^7 x^3 + \alpha^6 x^4$				
Локаторы ошибок и стираний $\alpha^2, \alpha^4, \alpha^5, \alpha^{10}$				
Многочлен значений ошибок $S(x)\Lambda(x) = \alpha^{12} + \alpha^8 x + \alpha^5 x^2 + \alpha^{14} x^3 + \alpha^5 x^6 + \alpha^{10} x^7 + \alpha^2 x^8 + \alpha x^9$ $\Omega(x) = S(x)\Lambda(x)(\mod x^6) = \alpha^{12} x + \alpha^8 x^2 + \alpha^5 x^3 + \alpha^{14} x^4$				
Значения ошибок $\alpha^6, 1$				
Значения стираний α^{13}, α^8				
Вектор ошибок $e(x) = \alpha^6 x^2 + x^5$				
Вектор стираний $e(x) = \alpha^{13} x^4 + \alpha^8 x^{10}$				

Таблица 3.3: Декодирование РС-кода с исправлением 2 ошибок и 2 стираний с помощью алгоритма Берлекэмпа-Месси

Цель метода – сведение декодирования в канале с мягкими решениями к многократным попыткам алгебраического декодирования с исправлением ошибок и стираний. В двух слова, суть алгоритма в том, что сначала выполняется обычное декодирование без стираний, с исправлением максимально возможного числа ошибок, затем декодирование с двумя стертыми наименее надежными символами, с четырьмя, и т.д. вплоть до $d-1$ стертых символов. Из всех результатов декодирования выбирается наиболее близкий к полученной из канала последовательности.

Напомним некоторые обозначения, использованные в главе 4.

Предположим, что символы двоичного (n,k)-кода $C=\{\boldsymbol{c}_m, m=1,...,2^k\}$ передаются по каналу с гауссовским шумом. Двоичные кодовые слова $\boldsymbol{c}_m$ преобразуются в кодовые слова из евклидова пространства $\boldsymbol{\xi}_m$ заменой нулей и единиц на вещественные числа -1 и $+1$ соответственно.

Предположим теперь, что значения последовательности $\boldsymbol{\alpha}=(\alpha_1,\alpha_2,...,\alpha_n)$ на выходе демодулятора (на входе декодера) нормированы и ограничены так, что $|\alpha_i|\le 1$, $i=1,2,...,n$.

Знаки компонент $\boldsymbol{\alpha}$ представляют собой жесткие решения, а их модули – надежности принятых символов. Если α_i вычислены как логарифмы отношения правдоподобия, то, как было показано в параграфе 4.1, в отсутствие ограничения на значения компонент, декодирование по максимуму скалярного произведения $(\boldsymbol{\alpha},\boldsymbol{\xi}_m)$ равносильно декодированию по максимуму правдоподобия. Введение нормировки не влияет на результат декодирования, зато дает возможность доказать следующую теорему.

Теорема 3.6. *В коде с длиной n и минимальным расстоянием d найдется не больше одного слова $\boldsymbol{\xi}_m$ такого, что*

$$(\boldsymbol{\alpha},\boldsymbol{\xi}_m) > n-d, \tag{3.26}$$

если $|\alpha_i|\le 1$, $i=1,2,...,n$.

Доказательство. Найдем слово $\boldsymbol{\xi}_{m'}$, отличающееся от $\boldsymbol{\xi}_m$ ровно в d позициях и обозначим через S множество позиций, в которых знаки

$\boldsymbol{\xi}_m$ и $\boldsymbol{\xi}_{m'}$ не совпадают, $|S| = d$. Положим

$$\begin{aligned} A &= \sum_{i \notin S} \alpha_i \xi_{mi}; \\ B &= \sum_{i \in S} \alpha_i \xi_{mi}. \end{aligned}$$

Заметим, что $A \leq n-d$, поскольку $|A| = n-d$ и по условию теоремы каждое слагаемое меньше единицы. Для двух скалярных произведений имеем

$$\begin{aligned} (\boldsymbol{\alpha}, \boldsymbol{\xi}_m) &= A + B; \\ (\boldsymbol{\alpha}, \boldsymbol{\xi}_{m'}) &= A - B. \end{aligned}$$

Их сумма

$$(\boldsymbol{\alpha}, \boldsymbol{\xi}_m) + (\boldsymbol{\alpha}, \boldsymbol{\xi}_{m'}) = 2A \leq 2(n-d).$$

Отсюда следует, что только одно из двух слагаемых может быть строго больше $n-d$. □

Определение 3.1. *Декодирование по минимуму евклидова расстояния между вектором $\boldsymbol{\alpha}$ и кодовыми словами $\boldsymbol{\xi}_m$ называется декодированием по минимуму обобщенного расстояния (МОР).*

Упражнение 14. *Докажите, что декодирование по МОР эквивалентно декодированию по максимуму скалярного произведения $(\boldsymbol{\alpha}, \boldsymbol{\xi}_m)$.*

Подсказка: Рассмотрите квадрат нормы $\|\boldsymbol{\alpha} - \boldsymbol{\xi}_m\|^2$.

Упражнение 15. *Докажите, что декодирование в ДСК по минимуму расстояния Хэмминга и декодирование в ДСтК по минимуму расстояния, вычисленного по нестертым позициям являются частными случаями декодирования по МОР. Запишите аналоги неравенства (3.26) для этих частных случаев.*

Следующая теорема подсказывает способ нахождения единственного решения $\boldsymbol{\xi}_m$ неравенства (3.26), если такое решение существует.

Теорема 3.7. *Если для некоторого m имеет место неравенство $(\boldsymbol{\alpha}, \boldsymbol{\xi}_m) > n-d$, то при некотором $f \in \{0,1,...,d-1\}$ при стирании f наименее надежных позиций декодер, исправляющий f стираний и комбинации ошибок кратности до $t = \lfloor (d-f-1)/2 \rfloor$, вынесет решение в пользу кодового слова $\boldsymbol{\xi}_m$.*

Доказательство. Для того, чтобы упростить запись, будем считать, что перед декодированием символы упорядочены по возрастанию надежности, т.е. $|\alpha_1| \le |\alpha_2| \le ... \le |\alpha_n|$. Символы кодовых слов, конечно, должны быть соответствующим образом перенумерованы.

Предположим сначала, что f принимает любые значения из множества $\{0,1,...,n\}$. Обозначим

$$\begin{aligned} p_0 &= |\alpha_1|; \\ p_1 &= |\alpha_2| - |\alpha_1|; \\ \dots &\quad \dots \quad \dots; \\ p_{n-1} &= |\alpha_n| - |\alpha_{n-1}|; \\ p_n &= 1 - |\alpha_n|. \end{aligned}$$

Числа $\{p_i\}$ неотрицательны и их сумма равна единице, что дает возможность рассматривать их как распределение вероятностей.

Введем вспомогательные векторы $\boldsymbol{\beta}^{(f)} = (\beta_1^{(f)}, \beta_2^{(f)}, ..., \beta_n^{(f)})$, где

$$\begin{aligned} \beta_i^{(f)} &= \begin{cases} 0, & i \le f \\ +1, & i > f, \alpha_i \ge 0 \\ -1, & i > f, \alpha_i < 0 \end{cases}, \\ f &= 0,1,...,n. \end{aligned}$$

Нетрудно подсчитать, что

$$\boldsymbol{\alpha} = \sum_{f=0}^{n} p_f \boldsymbol{\beta}^{(f)}. \tag{3.27}$$

В то же время, скалярные произведения $(\boldsymbol{\beta}^{(f)}, \boldsymbol{\xi}_m)$ представляет собой метрику декодирования , с жесткими решениями при наличии f стираний и неравенство $(\boldsymbol{\beta}^{(f)}, \boldsymbol{\xi}_m) > n-d$ может выполняться

только для одного слова, которое и является решением при данном f. Из (3.27) имеем

$$(\boldsymbol{\alpha}, \boldsymbol{\xi}) = \sum_{f=0}^{n} p_f (\boldsymbol{\beta}^{(f)}, \boldsymbol{\xi}). \tag{3.28}$$

Итак, $(\boldsymbol{\alpha}, \boldsymbol{\xi})$ равно среднему по всем f значению $(\boldsymbol{\beta}^{(f)}, \boldsymbol{\xi})$ при распределении вероятностей $\{p_f\}$ на f. Среднее значение не больше максимального. По условию теоремы, среднее значение больше $n-d$, а значит и одно из скалярных произведений в правой части (3.28) больше $n-d$. Следовательно, искомое решение будет вынесено при одном из $f \in \{0, 1, ..., n\}$.

С другой стороны, декодер, исправляющий ошибки и стирания, может выносит решения только при числе стираний меньшем d, отсюда следует ограничение на f в формулировке теоремы. □

Отметим, что при четной разности $d-f$ решение декодера будет таким же как и при на единицу большем числе стираний. Поэтому максимальное число попыток декодирования примерно равно $d/2$.

Кроме того, как только выполнено неравенство $(\boldsymbol{\alpha}, \boldsymbol{\xi}_m) > n - d$, декодирование может быть остановлено. Поэтому среднее число попыток будет заметно меньше числа $d/2$.

Задачи

1. Дана последовательность (0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1) над полем $GF(2)$. Постройте минимальный РЛОС, генерирующий эту последовательность.

2. Какие коды можно построить над полями GF(5), GF(7)? Приведите пример кода, проверьте на нем работу алгоритмов ПГЦ, БМ.

3. Постройте код Рида-Соломона длины 7, исправляющий двойные ошибки. Приведите пример вектора ошибок веса 2 и выполните декодирование по алгоритму ПГЦ.

4. Решите предыдущую задачу с использованием алгоритма БМ.

5. Для кода из задачи 3 приведите пример вектора ошибок веса 1 и вектора стираний веса 2. Примените алгоритм БМ для нахождения кодового слова.

6. Оцените сложность в числе операций в расширении поля для каждого из этапов декодирования при декодировании РС-кодов по алгоритму ПГЦ и БМ.

7. Постройте двоичный БЧХ-код длины 31, исправляющий 3 ошибки. Приведите пример вектора, содержащего 3 ошибки. Выполните декодирование, используя алгоритмы ПГЦ и БМ.

8. Используя опыт решения предыдущей задачи, предложите упрощения, возможные при использовании описанных в разделе алгоритмов для декодирования двоичных БЧХ-кодов. Изменится ли асимптотическое поведение сложности декодирования как функции длины кодов и числа исправляемых ошибок для двоичных кодов по сравнению с q-ичными?

9. БЧХ-коды и коды РС могут быть укорочены удалением информационных символов. При этом расстояние кода не уменьшается, но код теряет свойство цикличности. Как воспользоваться описанными в данной главе алгоритмами для декодирования укороченных циклических кодов?

Приложение. Линейная сложность последовательностей

Пусть имеется последовательность $\boldsymbol{s} = (s_1, s_2, ...)$ конечной или бесконечной длины с элементами из конечного поля $GF(q)$ и нужно построить линейную схему, которая ее порождает. Этой линейной схемой служит линейный регистр с обратной связью (ЛРОС). Его можно описать разностным уравнением, выражающим очередное значение последовательности в виде линейной комбинации предыдущих значений

$$s_{n+1} + c_1 s_n + ... + c_1 s_{n-L+1} + 0, \tag{3.29}$$

где L – длина регистра, а вектор $\boldsymbol{c} = (1, c_1, ..., c_L)$ задает весовые коэффициенты умножителей в цепи обратной связи (см. рис.3.2). Альтернативной формой описания регистра служит полином $c(x) = 1 + c_1 x + ... + c_L x^L$.

Если вся последовательность может быть порождена регистром длины L и не может быть порождена более коротким регистром (разностным уравнением более низкого порядка), то L называется *линейной сложностью последовательности.*

Пример 3.2. *Простейшее разностное уравнение*

$$s_n - c_1 s_{n-1} = 0$$

нетривиально при $c_1 \neq 0$. *При начальном состоянии* s_1 *оно порождает последовательность* $(c_1 s_1, c_1^2 s_1, ..., c_1^n s_1, ...)$. *Сложность последовательности равна 1, ей соответствует полином* $c(x) = 1 - c_1 x$.

Пример 3.3. *Разностное уравнение*

$$s_n - s_{n-L} = 0$$

(в полиномиальной записи $c(x) = 1 - x^L$ *) при начальном состоянии* $($ $\underbrace{0, 0, ..., 0}_{L-1 \text{ нулей}}$ $a)$ *периодически повторяет эту последовательность. Сложность этой бесконечной последовательности равна* L *и совпадает с ее периодом.*

Пример 3.4. *Нетрудно проверить, что в поле* $GF(2)$ *при любом ненулевом начальном состоянии уравнение*

$$s_n = s_{n-1} + s_{n-3}$$

(в полиномиальной записи $c(x) = 1 + x + x^3$ *) порождает последовательность с периодом 7. Это максимально возможный период для последовательностей, задаваемых полиномом третьей степени, поскольку общее число ненулевых состояний регистра равно 7.*

В общем случае, если в качестве $c(x)$ *выбрать примитивный многочлен степени* L, *получим на выходе регистра последовательность длины* $2^L - 1$, *которая представляет собой кодовое слово кода*

максимальной длины, двойственного коду Хэмминга (см. параграф 1.2). Действительно, выписав соответствующее рекуррентного уравнение, можно заметить, что сдвиги примитивного многочлена $c(x)$ являются проверочными соотношениями для любой последовательности, порождаемой данным регистром. Следовательно, проверочная матрица линейного пространства таких последовательностей совпадает с порождающей матрицей кода Хэмминга.

В этих примерах бесконечная (периодическая) последовательность задавалась полиномом конечной степени. Нетрудно догадаться, что в общем случае при длине регистра L период последовательности не может быть больше, чем $q^L - 1$ (число различных ненулевых состояний регистра).

С другой стороны, любая последовательность длины (периода) n может быть порождена тривиальным РЛОС: достаточно записать последовательность в регистр и соединить выход регистра с его входом.

Таким образом, сложность L последовательности длины (периода) n может находиться в широких пределах от $L = \log_q(n+1)$ до $L = n$.

Рассмотрим задачу нахождения минимального регистра, порождающего заданную последовательность.

Предположим сначала, что L известно и нужно только найти коэффициенты обратной связи (коэффициенты уравнения (3.29). Задача решается просто. Выпишем (3.29) при $n = 1, 2, ...2L - 1$. Запишем полученную систему уравнений в матричном виде

$$\begin{pmatrix} s_1 & s_2 & \cdots & s_L \\ s_2 & s_3 & \cdots & s_{L+1} \\ \vdots & \vdots & \ddots & \vdots \\ s_L & s_{L+1} & \cdots & s_{2L-1} \end{pmatrix} \begin{pmatrix} c_L \\ c_{L-1} \\ \vdots \\ c_1 \end{pmatrix} \begin{pmatrix} -s_{L+1} \\ -s_{L+2} \\ \vdots \\ -s_{2L} \end{pmatrix} \tag{3.30}$$

Если гипотеза о линейной сложности L верна, то существует единственное решение этой системы уравнений, оно и является решением этой задачи. Приходим к следующему утверждению.

Теорема 3.8. *Для вычисления коэффициентов ЛРОС, описывающего последовательность сложности L достаточно иметь любые $2L$ последовательных значений на выходе фильтра.*

Возвращаясь к примеру 3.4, заметим, что последовательность, порожденная регистром, заданным примитивным многочленом степени L, может быть восстановлена полностью по любому отрезку длины $2L$. Этот факт препятствует применению таких ЛРОС в криптографии.

То, что на сложность L неизвестна заранее, не намного усложняет дело, поскольку можно по очереди проверять гипотезы $L = 1, 2, ...$ Решение системы из L уравнений имеет сложность не более L^3, поэтому сложность общая нахождения регистра не превышает L^4, т.е. остается в любом случае полиномиальной.

Алгоритм Берлекэмпа-Месси решает совместно обе задачи – определение линейной сложности и нахождение коэффициентов полинома ЛРОС со сложностью порядка L^2. Это упрощение обусловлено тем, что матрица коэффициентов имеет специфический вид: строки коэффициентов являются сдвигами предыдущих строк.

Решение задачи мы уже описали в параграфе 3.2. Остаток параграфа посвящен доказательству его оптимальности, т.е. минимальности длины регистра, получаемого по алгоритму БМ.

Лемма 3.9. *Пусть* $\boldsymbol{c}^{(r-1)}$ *длины* $L_{(r-1)}$ *– регистр минимальной длины, генерирующий* $\boldsymbol{s}_{r-1} = (s_1, s_2, ..., s_{r-1})$, *а* $\boldsymbol{c}^{(r)}$ *длины* $L_{(r)}$ *– регистр минимальной длины, генерирующий* $\boldsymbol{s}_r = (s_1, s_2, ..., s_r)$ *и пусть* $\boldsymbol{c}^{(r-1)} \neq \boldsymbol{c}^{(r)}$. *Тогда*

$$L_r \geq \max\left\{L_{r-1}, r - L_{r-1}\right\}.$$

Доказательство. Неравенство $L_{(r)} \geq L_{(r-1)}$ очевидно, поскольку регистр, генерирующий некоторую последовательность, генерирует также любую последовательность, продолжением которой эта последовательность является.

Докажем неравенство $L_{(r)} \geq r - L_{(r-1)}$. Для сокращения записи обозначим $\boldsymbol{a} = \boldsymbol{c}^{(r-1)}$, $\boldsymbol{b} = \boldsymbol{c}^{(r)}$, $L_a = L_{r-1}$, $L_b = L_r$.

Доказательство от противного. Предположим, что $r > L_a + L_b$.

Из условий леммы имеем

$$s_j = -\sum_{i=1}^{L_a} a_i s_{j-i}, \quad j = L_{a+1}, ..., r-1, \tag{3.31}$$

$$s_r \neq -\sum_{i=1}^{L_a} a_i s_{r-i}, \tag{3.32}$$

$$s_j = -\sum_{k=1}^{L_b} b_k s_{j-k}, \quad j = L_{b+1}, ..., r, \tag{3.33}$$

поскольку $\boldsymbol{a}$ порождает последовательность длины $r-1$, но не порождает последовательность длины r, а $\boldsymbol{b}$ порождает последовательность длины r.

Применяя (3.33) при $j = r$, получаем после подстановки (3.31)

$$s_r = -\sum_{k=1}^{L_b} b_k s_{r-k} = -\sum_{k=1}^{L_b} b_k \sum_{i=1}^{L_a} a_i s_{r-k-i}. \tag{3.34}$$

Здесь существенно то, что индексы при s остаются положительными в силу предположения $r > L_a + L_b$. С другой стороны, подстановка (3.33) в (3.32) дает

$$s_r \neq -\sum_{i=1}^{L_a} a_i s_{r-i} = -\sum_{i=1}^{L_a} a_i \sum_{k=1}^{L_b} b_k s_{r-i-k}. \tag{3.35}$$

Очевидно, (3.34) противоречит (3.35). □

Хотя доказательство несложное, приведем еще одно, опирающееся больше на интуицию, чем на формальные вычисления. Для этого сформулируем вспомогательные утверждения.

Лемма 3.10. *Линейная сложность суммы последовательностей не превышает суммы их линейных сложностей.*

Доказательство. Устройство, порождающее отдельные слагаемые и суммирующее их – неоптимальная (возможно, не самая короткая) версия минимального ЛРОС. □

Лемма 3.11. *Линейная сложность последовательности* $\boldsymbol{s}$ *длины* r *вида* $\boldsymbol{s} = (\underbrace{0, 0, ..., 0}_{r-1 \text{ нулей}}\ a)$ *равна* r .

Доказательство. Один (не единственный возможный) полином обратной связи, генерирующий $\boldsymbol{s}$, равен $a - x^r$ (см. пример 3.3). Следовательно, линейная сложность не больше r. С другой стороны, регистр длины меньше r не может породить серию из $r-1$ нулей. Значит, сложность равна r . □

Доказательство леммы 3.9. Пусть $\boldsymbol{s}_r$ – последовательность длины r и $\hat{\boldsymbol{s}}_r$ – последовательность длины r, порождаемая фильтром длины L_{r-1}. По условию леммы 3.9 $\boldsymbol{s}_r - \hat{\boldsymbol{s}}_r = (0, 0, ..., 0, \Delta)$, где $\Delta \neq 0$ – невязка. Сложность суммы, в соответствии с леммой 3.11, оказалась равной r. Из леммы 3.10 имеем $L_r + L_{r-1} \geq r$, что и требовалось доказать. □

Из леммы 3.9 и теоремы 3.3, следует, что алгоритм Берлекэмпа-Месси строит регистр минимально возможной длины, т.е. для заданной последовательности синдромов он отыскивает вектор ошибок минимального веса.

Пример 3.5. *Предположим, что на выходе троичной линейной схемы наблюдается последовательность* $x = (2\ 1\ 0\ 1\ 2\ 1\ 0\ 1\ 2\ 1\ 0\ 1\ ...)$ *Построим минимальный ЛРОС, порождающий эту последовательность. Результаты вычислений, выполняемых по алгоритму Витерби шаг за шагом, приведены в таблице 3.4, а соответствующие схемы фильтров показаны на рис.3.6.*

Поскольку в алгоритме БМ (см. 3.3) качестве фильтра нулевой длины выбран фильтр $\Lambda = 1$*, невязка на первом шаге подсчитывается относительно единицы. Это разумное соглашение, поскольку любое поле содержит единицу. Фильтр первого порядка после первых двух итераций успешно сформировал первые два символа последовательности, но третий (он выделен жирным шрифтом на рис. 3.6а) неверен. На следующих итерациях в качестве исходного состояния фильтра принимаем уже аппроксимированные элементы последовательности. На пятой итерации приходится увеличивать порядок фильтра до трех. В результате оказалось,*

r	s	Δ	$B(x)$	$\Lambda(x)$	L
0	–	0	1	1	0
1	2	2	2	$1+x$	1
2	1	0	$2x$	$1+x$	1
3	0	1	$1+x$	$1+x+x^2$	2
4	1	2	$x+x^2$	$1+2x+2x^2$	2
5	2	1	$1+2x+2x^2$	$1+2x+x^2+2x^3$	3
6	1	0	$x+2x^2+2x^3$	$1+2x+x^2+2x^3$	3
7	0	0	$x^2+2x^3+2x^4$	$1+2x+x^2+2x^3$	3

Таблица 3.4: Построение ЛРОС по последовательности на его выходе

что заданная последовательность длины 12 описывается фильтром третьего порядка.

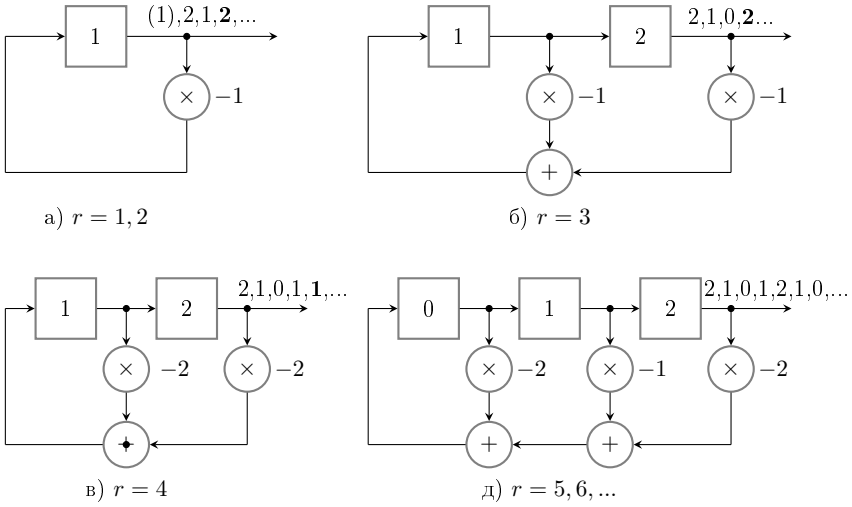

Рис. 3.6: Построение фильтра по заданной последовательности

Глава 4

Введение в сверточные коды

Глава 5

Алгебраическое описание сверточных кодов

Глава 6

Длинные коды из коротких кодов

Глава 7

Коды с малой плотностью проверок на четность

Литература

[1] Сергей Георгиевич Влэдуц, Г Л Кацман, and Михаил Анатольевич Цфасман. Модулярные кривые и коды с полиномиальной сложностью построения. *Проблемы передачи информации*, 20(1):47–55, 1984.

[2] Полтырев Г. Ш. Колесник В. Д. *Курс теории информации.* М.: Наука, 1982.

[3] Р Дж. Галлагер. *Коды с малой плотностью проверок на четность.* Мир, 1966.

[4] Е А Крук. Граница для сложности декодирования линейных блоковых кодов. *Проблемы передачи информации*, 25(3):103–107, 1989.

[5] Галлагер Р. *Теория информации и надежная связь.* М.: Советское радио, 1974.

[6] Ф Дж. Мак-Вильямс, Н. Дж. А. Слоэн, Теория кодов, исправляющих ошибки, Москва,". *Связь*, 1979.

[7] Digital video broadcasting (DVB), August 2009.

[8] Barg A. *Complexity Issues in Coding Theory*, volume 1, chapter Complexity Issues in Coding Theory". North-Holland, 1998.

[9] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Transactions on Information Theory*, IT-20(1):284–287, Mar. 1974.

[10] Toby Berger. *Rate-Distortion Theory.* Wiley Online Library, 1971.

[11] Claude Berrou, Alain Glavieux, and Punya Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. (1). In *1993. ICC 93. Geneva. Technical Program, Conference Record, IEEE International Conference on Communications*, volume 2, pages 1064–1070. IEEE, 1993.

[12] Joseph J. Boutros and Kalpesh Patel. Performance of optimal codes at finite length. 2006.

[13] John T Coffey and Rodney M Goodman. The complexity of information set decoding. *IEEE Transactions on Information Theory*, 36(5):1031–1037, 1990.

[14] Gérard Cohen. A nonconstructive upper bound on covering radius. *IEEE Transactions on Information Theory*, 29(3):352–353, 1983.

[15] Robert Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory,*, 8(1):21–28, 1962.

[16] Sason Igal and Shamai Shlomo. *Performance analysis of linear codes under maximum-likelihood decoding: a tutorial.* Now Publishers Inc, 2006.

[17] Lev Levitin and Carlos Hartmann. A new approach to the general minimum distance decoding problem: The zero-neighbors algorithm. *IEEE Transactions on Information Theory*, 31(3):378–384, 1985.

[18] Robert McEliece. *The theory of information and coding.* Addison-Wesley, Reading, Mass., 1977.

[19] Gregory Poltyrev. Bounds on the decoding error probability of binary linear codes via their spectra. *IEEE Transactions on Information Theory*, 40(4):1284–1292, 1994.

[20] Claude E Shannon. Probability of error for optimal codes in a gaussian channel. *Bell System Technical Journal*, 38(3):611–656, 1959.

[21] Claude Elwood Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27(1):379–423, 1948.

[22] Gottfried Ungerboeck. Channel coding with multilevel/phase signals. *IEEE Transactions on Information Theory*, 28(1):55–67, 1982.

[23] Antoine Valembois and Marc PC Fossorier. Sphere-packing bounds revisited for moderate block lengths. *EEE Tranctions on Information Theory*, 50(12):2998–3014, 2004.

[24] Alexander Vardy. Trellis structure of codes. *Handbook of coding theory*, 2:1989–2117, 1998.

[25] Stephen B Wicker and Vijay K Bhargava. *Reed-Solomon codes and their applications*. Wiley. com, 1999.

[26] J Wolf. Efficient maximum likelihood decoding of linear block codes using a trellis. *IEEE Transactions on Information Theory*, 24(1):76–80, 1978.

Предметный указатель