

Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники

Кафедра радиотехнических систем

С.Б. Саломатин

ИССЛЕДОВАНИЕ СВОЙСТВ КОДОВ БОУЗА–ЧОУДХУРИ–ХОКВИНГВЕМА

МЕТОДИЧЕСКОЕ ПОСОБИЕ
к лабораторной работе по дисциплине
КОДИРОВАНИЕ И ЗАЩИТА ИНФОРМАЦИИ
для студентов специальностей 39 01 02 «Радиоэлектронные системы»
и 39 01 03 «Радиоинформатика» дневной формы обучения

Минск 2004

УДК 681.391. 25 (076)

ББК 32.811.4 я 7

С 16

Саломатин С.Б.

С 16 Исследование свойств кодов Боуза – Чоудхури – Хоквингвема:
Метод. пособие к лаб. работе по дисц. «Кодирование и защита информации» для студ. спец. 39 01 02 «Радиоэлектронные системы» и 39 01 03 «Радиоинформатика» дневн. формы обуч./ С.Б. Саломатин. - Мн.: БГУИР, 2004.- 24 с.
ISBN 985-444-711-1

Методическое пособие содержит теоретические сведения, алгоритмы, программы моделирования процессов кодирования и декодирования, а также исследования структурных свойств блочных корректирующих кодов Боуза–Чоудхури–Хоквингвема (БЧХ). В лабораторной работе исследуются различные методы кодирования и декодирования. Оценивается эффективность кодирования информации.

УДК 681.391.25 (076)

ББК 32.811. 4 я 7

ISBN 985-444-711-1

© Саломатин С. Б., 2004

© БГУИР, 2004

Содержание

1. ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ
2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ
 - 2.1. Задание циклического кода с помощью элементов конечного поля
 - 2.2. БЧХ-коды
 - 2.3. Декодирование БЧХ-кодов
 - 2.3.1. Алгоритм Питерсона–Горенштейна–Цирлера (ПГЦ)
 - 2.3.2. Алгоритм Сугиямы
 - 2.3.3. Алгоритм Берлекэмп–Мессе
 - 2.4. Нумератор весов кода
3. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
 - 3.1. Программы кодирования
 - 3.2. Программы для исследования свойств кодов
 - 3.3. Программы декодирования кодов БЧХ
 - 3.3.1. Программа декодирования БЧХ - кода по алгоритму ПГЦ
4. СОДЕРЖАНИЕ И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ
 - 4.1. Содержание работы
 - 4.2. Порядок выполнения работы
5. СОДЕРЖАНИЕ ОТЧЕТА
- КОНТРОЛЬНЫЕ ВОПРОСЫ
- ЛИТЕРАТУРА

1. ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

1. Изучить методы помехоустойчивого кодирования и декодирования информации с помощью циклических блочных кодов Боуза–Чоудхури–Хоквинггема.
2. Исследовать свойства кодов Боуза–Чоудхури–Хоквинггема (БЧХ), изучить связь их корректирующей способности с конечными полями.
3. Приобрести навыки построения алгоритмов и программ кодирования и декодирования информации.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

2.1. Задание циклического кода с помощью элементов конечного поля

Корректирующий циклический (n, k) - код C представляется в виде вектора-строки $\mathbf{c} = (c_{n-1}, c_{n-2}, \dots, c_0) \in (\mathbf{Z}_2)^n$ или в полиномиальном виде как

$$c(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \dots + c_0 \in \mathbf{Z}_2[x],$$

где $\mathbf{Z}_2[x]$ - кольцо многочленов над полем \mathbf{Z}_2 .

Обозначим примитивный элемент конечного поля $GF(2^r)$ как α . Определим проверочную матрицу кода \mathbf{H} , столбцы которой равны $\alpha^{n-1}, \alpha^{n-2}, \dots, \alpha, 1$, где $n = 2^r - 1$. Произведению $(\mathbf{c} \mathbf{H}^T)$ соответствует полином синдрома

$$S(\alpha) = c_{n-1}\alpha^{n-1} + c_{n-2}\alpha^{n-2} + \dots + c_2\alpha^2 + c_1\alpha + c_0.$$

Для дальнейшего изложения полезно ввести следующие понятия.

Минимальный многочлен. Пусть F - поле, F_0 - подполе поля F и α - элемент мультипликативной группы $F - \{0\}$ поля F . Обозначим через I_α множество всех многочленов над F_0 , корнем которых является α . Пусть $m_\alpha(x)$ - нормированный многочлен минимальной степени в I_α ; этот многочлен называется минимальным многочленом элемента α над F_0 . Минимальный многочлен является неприводимым.

Пример. Рассмотрим поле $GF(16) = GF(2^4)$, $r = 4, n = 2^4 - 1 = 15$. Возможными коэффициентами для минимального многочлена являются 0 или 1. Получаем следующее соответствие (табл.1).

Таблица 1

Элемент поля	Минимальный многочлен
0	x
1	$x+1$
α, α^{-1}	$x^4 + x + 1$
α^3	$x^4 + x^3 + x^2 + x + 1$

a^5	$x^2 + x + 1$
-------	---------------

Сопряженные элементы и циклотомические классы. Минимальные многочлены элементов b и b^2 равны. Элементы поля, минимальные многочлены которых равны, называются сопряженными. Так, для поля $GF(16) = GF(2^4)$ минимальным многочленом элементов $a, a^2, (a^2)^2 = a^4, (a^4)^2 = a^8$ (при этом $(a^8)^2 = a$) является один и тот же многочлен. Аналогично для $a^3, a^6, a^{12}, a^{24} = a^9$ минимальным многочленом является один и тот же многочлен. Видно, что степени элемента a распадаются на непересекающиеся множества, которые называются циклотомическими классами. Когда индекс j пробегает циклотомический класс, то минимальным для всех a^j является один и тот же многочлен.

Множество целых чисел по модулю $p^m - 1$ относительно операции умножения на p распадается на подмножества, которые называются *циклотомическими классами* по модулю $p^m - 1$. Циклотомический класс, содержащий s , состоит из чисел

$$K_s = \{s, ps, p^2s, p^3s, \dots, p^{t_s}s\} = \{k_{s,0}, k_{s,1}, \dots, k_{s,t_s}\},$$

где t_s - наименьшее положительное целое число, такое, что $p^{t_s}s \equiv s \pmod{p^m - 1}$.

Например, циклотомическими классами по модулю 15 (для $p = 2$) являются $K_0 = \{0\}$; $K_1 = \{1, 2, 4, 8\}$; $K_3 = \{3, 6, 12, 9\}$; $K_5 = \{5, 10\}$; $K_7 = \{7, 14, 13, 11\}$.

Циклотомические классы позволяют вычислить минимальные полиномы с помощью выражения $m_s(x) = \prod_{i \in K_s} (x - a^i)$. Используя введенные определения, можно построить следующую конструкцию (n, k) - кода.

1. Представим кодируемую информацию (сообщение) $a(x)$ старшей частью полинома

$$a(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \dots + c_kx^k.$$

2. Найдем остаток $\text{Rem}(x)$ от деления сообщения $a(x)$ на минимальный полином $m_a(x)$:

$$\text{Rem}(x) = a(x) \bmod m(x).$$

3. Сформируем кодовое слово по правилу

$$c(x) = a(x) + \text{Rem}(x).$$

Декодирование принимаемого сигнала $y(x) = \sum_{i=0}^{n-1} y_i x^i$ можно вести в следующей последовательности. Подставим $x = a$ в $y(x)$ и вычислим синдром ошибки $S = y(a)$. Тогда если $y(a) = 0$, то ошибок в принятом сигнале нет; если $y(a) \neq 0$, то $y(a) = a^j$ для некоторого j . Полином ошибки $e(x) = a^j$. Следовательно, принимаемый символ y_j ошибочен и должен быть исправлен $c(x) = y(x) - e(x)$.

Пример. Пусть двоичное сообщение имеет вид $a = (1,1,0,1,1,0,1,0,0,0,0)$. Применим код (15,11). Образует полином $a(x) = x^{14} + x^{13} + x^{11} + x^{10} + x^8$.

Разделим сообщение $a(x)$ на минимальный полином $m(x) = x^4 + x + 1$, $m(x) \rightarrow a \in GF(2^4)$: $a(x) = (x^{10} + x^9 + x^6 + x^5 + x^4 + x^3)m(x) + x^3$, $Rem(x) = x^3$.

Кодовое слово равно $c(x) = a(x) + Rem(x) = x^{14} + x^{13} + x^{11} + x^{10} + x^8 + x^3$.

Пусть вектор принимаемого сигнала равен $y = (1,1,0,1,1,0,1,0,0,0,1,1,0,0)$ и $y(x) = x^{14} + x^{13} + x^{11} + x^{10} + x^8 + x^3 + x^2$. Если α - корень $m(x)$, то, используя правила сложения в поле $GF(2^4)$, получим

$$y(a) = a^{14} + a^{13} + a^{11} + a^{10} + a^8 + a^3 + a^2 = a^2.$$

Следовательно, ошибка произошла на позиции $j = 2$. Полином ошибки $e(x) = a^2$. Символ y_2 должен быть исправлен с 1 на 0.

Рассмотрим случай, когда требуется корректировать две ошибки. Минимальный полином $m_1(x)$ элемента α будет иметь своими корнями также все элементы α^j , $j \in K_1$. Использование полинома $m_1(x)$ достаточно для коррекции одной ошибки, но недостаточно для исправления двух ошибок. Обратимся к коду Хэмминга (15,11), $p^k = 16$, $K_1 = \{1,2,4,8\}$. Возьмем элемент поля $a^3 \in GF(2^4)$, которому соответствует минимальный неприводимый в $\mathbb{Z}_2[x]$ полином $m_3(x) = x^4 + x^3 + x^2 + x + 1$. Элемент α^3 является корнем полинома $(x^5 - 1)$. Элементы α и α^3 являются корнями полинома

$$m_{13}(x) = m_1(x)m_3(x) = (x^4 + x + 1)m_3(x) = x^8 + x^7 + x^6 + x^4 + 1.$$

Если полином сообщения $a(x)$ разделить на $m_{13}(x)$, то для кода с $n = 15$ получим остаток $Rem(x)$ степени не выше 7:

$$a(x) = c_{14}x^{14} + c_{13}x^{13} + \dots + c_9x^9 + c_8x^8,$$

$$Rem(x) = a(x) \bmod(m_{13}(x)) = c_7x^7 + \dots + c_1x + c_0.$$

В этом случае сообщение длиной 7 может быть закодировано кодовым словом, $c(x) = a(x) + Rem(x) = c_{14}x^{14} + \dots + c_1x + c_0$, состоящим из 15 символов.

Процесс декодирования подобен процедуре исправления одной ошибки, но при исправлении двух ошибок требуется решить квадратное уравнение. Запишем принимаемый сигнал в виде $y(x) = c(x) + e(x)$, где $y(x)$ – полином ошибок.

Для рассматриваемого примера $c(a) = c(a^3) = 0$. Ошибки в принимаемом сигнале отсутствуют в том случае, если $e(x) = 0$ или $y(a) = y(a^3) = 0$. Если произошла одна ошибка, то $e(x) = x^e$ для различных значений e . В этом случае $y(a) = a^e$ и $y(a^3) = a^{3e} = y(a)^3$. Если в принимаемом сигнале два символа ошибочны, то

$e(x) = x^e + x^f$ для различных значений $e, f, e \neq f$, при этом $y(a) = a^e + a^f$ и $y(a^3) \neq y(a)^3$.

Определим уравнение $z^2 + s_1 z + s_2 = 0$, которое имеет два корня $b_1 = a^e$ и $b_2 = a^f$, при этом $s_1 = b_1 + b_2$, $s_2 = b_1 b_2$. Заметим, что корни уравнения однозначно связаны с координатами e и f ошибок. Решение уравнения позволяет определить местоположение ошибок в декодируемом сигнале. Многочлен в левой части уравнения может быть найден с использованием синдромов ошибок следующим образом. Компоненты S_i вектора синдромов $\mathbf{S} = (S_1, S_2, \dots, S_{2t-1})$ и коэффициенты полинома позиций ошибок связаны соотношениями

$$S_1 = y(a) = e(a) = b_1 + b_2 = s_1; S_2 = y(a^2) = e(a^2) = s_1^2;$$

$$S_3 = y(a^3) = b_1^3 + b_2^3 = (b_1 + b_2)^3 + (b_1 + b_2)b_1 b_2 = S_1 S_2 + S_1 s_2 = s_2 S_1 + s_2 S_2.$$

По рассмотренным примерам можно сделать обобщение и дать определение кодов Боуза–Чоудхури–Хоквингвема (БЧХ).

2.2. БЧХ-коды

Определение примитивного БЧХ-кода. Пусть m и $t < p^m - 1$ - целые числа, тогда q -ичный БЧХ-код имеет параметры $n = p^m - 1$; $k \geq n - m \cdot t$; $d_{\min} \geq 2t + 1$. Код формируется генераторным полиномом $g(x)$, имеющим корнями примитивный элемент поля $GF(p^m)$ a^i , $i = 1, \dots, 2t$. Соответственно

$$g(x) = \text{НОК}(m_a(x), m_{a^2}(x), \dots, m_{a^{2t}}(x)), \quad (1)$$

где m_{a^i} - минимальный полином, соответствующий элементу поля a^i .

Для $p = 2$ в поле $GF(2^m)$ элементы a и a^2 имеют одинаковые минимальные полиномы, поэтому для двоичного БЧХ-кода

$$g(x) = \text{НОК}(m_a(x), m_{a^3}(x), \dots, m_{a^{2t-1}}(x)). \quad (2)$$

Расширенные БЧХ-коды формируются по следующему правилу:

$$g(x) = \text{НОК}(m_{a^b}(x), m_{a^{b+1}}(x), \dots, m_{a^{b+2t-1}}(x)). \quad (3)$$

Алгоритм формирования кода имеет следующий вид:

1. Строится поле $GF(p^m)$, для которого a - примитивный элемент поля.

2. Для a^i , $i = 1, \dots, 2t$ определяются примитивные полиномы $m_i(x) = m_{a^i}(x)$.

3. Находится $g(x)$ как $g(x) = \text{НОК}(m_1(x), m_2(x), \dots, m_{2t}(x))$.

Основные свойства кода

Код БЧХ может исправлять t ошибок.

Если полином $y(x)$ принимаемого сигнала содержит не более t ошибок и синдромы ошибок равны $S_i = y(a^i)$ для $i = 1, 2, \dots, 2t$, то количество ошибок равно рангу ν матрицы

$$\begin{bmatrix} S_1 & S_2 & \mathbf{L} & S_t \\ S_2 & S_3 & \mathbf{L} & S_{t+1} \\ \mathbf{M} & \mathbf{M} & \mathbf{O} & \mathbf{M} \\ S_t & S_{t+1} & \mathbf{L} & S_{2t-1} \end{bmatrix}.$$

Если в принимаемом сигнале имеется n ошибок, тогда полином ошибок может быть представлен в виде

$$e(x) = y(x) - c(x) = x^{m_1} + x^{m_2} + \dots + x^{m_n},$$

при этом степени примитивного элемента $a^{m_1}, a^{m_2}, \dots, a^{m_n}$ являются корнями полинома позиций ошибок

$$z^n + s_1 z^{n-1} + \dots + s_{n-1} z + s_n,$$

коэффициенты которого связаны с компонентами синдрома ошибок системой уравнений

$$\begin{bmatrix} S_1 & S_2 & \mathbf{L} & S_n \\ S_2 & S_3 & \mathbf{L} & S_{n+1} \\ \mathbf{M} & \mathbf{M} & \mathbf{O} & \mathbf{M} \\ S_n & S_{n+1} & \mathbf{L} & S_{2n-1} \end{bmatrix} \begin{bmatrix} s_n \\ s_{n-1} \\ \mathbf{M} \\ s_1 \end{bmatrix} = - \begin{bmatrix} S_{n+1} \\ S_{n+2} \\ \mathbf{M} \\ S_{2n} \end{bmatrix}.$$

Примеры:

1. Пусть дано поле $GF(2^4)$ $m=4$, $q=2$. Зададим параметры кода $t=2$, $n=2^4-1=15$, $k \geq n - m \cdot t = 15 - 4 \cdot 2 = 7$, тогда

$$g(x) = \text{НОК}(m_1(x), m_2(x), m_3(x), m_4(x)) = x^8 + x^7 + x^6 + x^4 + 1.$$

В результате получается БЧХ-код $(15, 7, 5)$, у которого $d_{\min} \geq 2 \cdot 2 + 1 = 5$. При степени полинома $\deg(d(x)) = 8$ количество проверочных символов в коде равно $r = n - k = 8$, а число информационных символов — $k = 7$.

2. Рассмотрим возможность построения БЧХ-кода (15, 7), корректирующего три ошибки. Определим конечное поле $GF(2^4)$, для которого существует четыре циклотомических класса:

$$K_1 = \{1, 2, 4, 8\}; K_2 = \{3, 6, 12, 9\}; K_5 = \{5, 10\}; K_7 = \{7, 14, 13, 11\}.$$

Степень примитивного элемента поля $(a^5)^3 = a^{15} = 1$ и, следовательно, a^5 является корнем многочлена $x^3 - 1 = x^3 + 1 = (x + 1)(x^2 + x + 1)$; и кодовые слова должны быть кратны полиному

$$m_{135}(x) = m_{13}(x) m_5(x) = m_{13}(x) (x^2 + x + 1) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1.$$

Возможности поля $GF(2^4)$ позволяют построить код, корректирующий четыре ошибки. Действительно, структура циклотомических классов показывает, может ли быть определен минимальный полином $m_7(x)$, корнями которого являются степени $a^7, a^{14}, a^{13}, a^{11}$. Требуемый полином $m(x)$, на основе которого формируется код, определяется через произведение $m(x) = m_{1357}(x) = m_1(x)m_3(x)m_5(x)m_7(x)$, имеет степень, равную 14. Получается БЧХ - код (15, 1) с одним информационным символом.

Для формирования кодов с лучшими корректирующими способностями необходимо использовать конечные поля большего размера. Так, если выбрать поле $GF(2^5) = \mathbb{Z}_2[x]/(x^5 + x^2 + 1)$, то можно получить следующие коды (табл. 2).

Таблица 2

Код	Циклотомический класс	Количество информационных символов k	Число исправляемых ошибок t
C_1	$\{1, 2, 4, 8, 16\}$	26	1
C_2	$\{3, 6, 12, 24, 17\}$	21	2
C_3	$\{5, 10, 20, 9, 18\}$	16	3
C_4	$\{7, 14, 28, 25, 19\}$	11	5
C_5	$\{11, 22, 13, 26, 21\}$	6	7
C_6	$\{15, 30, 29, 27, 23\}$	1	15

Так, код C_4 формируется с помощью полинома

$$m(x) = m_{1357}(x) = m_1(x)m_3(x)m_5(x)m_7(x),$$

степень которого равна 20. Количество проверочных символов равно 20, а число информационных символов $31 - 20 = 11$. Степени $\alpha, \alpha^2, \dots, \alpha^{10}$ являются корнями полинома $m(x)$, но α^{11} не является корнем $m(x)$. Следовательно, код C_4 исправляет 5 ошибок. Минимальный полином циклотомического класса K_7 получается из выражения $m_7(x) = (x - a^7)(x - a^{14})(x - a^{28})(x - a^{25})(x - a^{19})$.

2.3. Декодирование БЧХ-кодов

Предположим, что задан БЧХ-код, корректирующий t ошибок и при приеме

произошло n ошибок, причём $0 \leq n \leq t$. Ошибки располагаются на позициях (i_1, i_2, \dots, i_n) , вектор ошибок $e(x) = e_{i_1} \cdot x^{i_1} + e_{i_2} \cdot x^{i_2} + \dots + e_{i_n} \cdot x^{i_n}$. Декодер решает две основные задачи. Первая – определяет координаты ошибок, т.е. значения $\{i_j\}$. Вторая - оценивает величины (амплитуды) ошибок.

Введем следующие обозначения: $i_j \rightarrow a^{ij}$; $X_e = a^{ie}$ - локатор ошибки; $e_{i_e} = Q_e$ - величина ошибки. Тогда компоненты синдрома могут быть вычислены как

$$S_j = y(a^j) = c(a^j) + e(a^j) = e(a^j), \quad j = 1, \dots, 2t.$$

С другой стороны, компоненты синдрома можно описать следующей системой уравнений:

$$\begin{aligned} S_1 &= Q_1 \cdot a^{i_1} + Q_2 \cdot a^{i_2} + \dots + Q_n \cdot a^{i_n}, \\ S_2 &= Q_1 \cdot a^{2i_1} + Q_2 \cdot a^{2i_2} + \dots + Q_n \cdot a^{2i_n}, \\ &\vdots \\ S_{n_i} &= Q_1 \cdot a^{2ti_1} + Q_2 \cdot a^{2ti_2} + \dots + Q_n \cdot a^{2ti_n}. \end{aligned}$$

или
$$S_j = \sum_{l=1}^n Q_l \mathbf{a}^{j \cdot i_l}, \quad j = 1, \dots, 2t.$$

Для функции ошибки определим дискретное преобразование Фурье в конечном поле $[4]$:

$$E_k = \sum_{l=0}^{n-1} e_l w^{lk}, \quad k=1, \dots, 2t.$$

Заметим, что компоненты синдрома можно трактовать как коэффициенты Фурье-преобразования $S_j = E_j, j = 1, \dots, 2t$ при $w^i = a^i$.

Оценки параметров ошибки можно получить путем решения полученной нелинейной системы уравнений, что достаточно сложно. Используем искусственный приём. Введем так называемый *полином локаторов ошибок* $S(x)$:

$$S(x) = S_n \cdot x^n + S_{n-1} \cdot x^{n-1} + \dots + S_1 \cdot x + S_0, \quad \text{или}$$

$$S(x) = (1 - x \cdot X_1) \cdot (1 - x \cdot X_2) \cdot \dots \cdot (1 - x \cdot X_n) = \prod_i (1 - x \cdot X_i),$$

нулям которого соответствуют элементы a^{il} , $l = 1, \dots, v$.

Корнями полинома $S(x)$ являются величины $l = X_i^{-1}$. Если коэффициенты $S(x)$

известны, то соответственно для определения локаторов ошибок X_i нужно найти корни $S(x)$, после чего вычислить обратные к ним величины. При этом решение системы нелинейных уравнений сводится к решению системы линейных уравнений относительно коэффициентов полинома $S(x)$. Действительно, умножим левую и правую части выражения для полинома локаторов ошибок на $Q_i \cdot x_i^{j+n}$, а также положим, что при $x = X_e^{-1}$ имеем $S(X_e^{-1}) = 0$. Образует систему

$$0 = Q_e \cdot X_e^{j+n} \cdot (1 + s_1 \cdot X_e^{-1} + s_2 \cdot X_e^{-2} + \dots + s_n \cdot X_e^{-n}), e = 1, \dots, n.$$

Просуммировав уравнения по индексу e от 1 до n , для каждого индекса j получим

$$\sum_{e=1}^n Q_e \cdot X_e^{j+n} + s_1 \cdot \sum_{e=1}^n Q_e \cdot X_e^{j+n-1} + \dots + s_n \cdot \sum_{e=1}^n Q_e \cdot X_e^j = 0.$$

Учитывая, что $S_i = \sum_{e=1}^n Q_e \cdot X_e^j$, система уравнений представляется в виде

$$s_1 \cdot S_{j+n-1} + s_2 \cdot S_{j+n-2} + \dots + s_n \cdot S_j = -S_{j+n}, j = 1, \dots, n,$$

или в матричной форме:

$$\begin{bmatrix} S_1 & S_2 & S_3 & \dots & S_{n-1} & S_n \\ S_2 & S_3 & S_4 & \dots & S_n & S_{n+1} \\ S_3 & S_4 & S_5 & \dots & S_{n+1} & S_{n+2} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ S_n & S_{n+1} & S_{n+2} & \dots & S_{2n-2} & S_{2n-1} \end{bmatrix} \cdot \begin{bmatrix} s_n \\ s_{n-1} \\ s_{n-2} \\ \dots \\ s_1 \end{bmatrix} = \begin{bmatrix} -S_{n+1} \\ -S_{n+2} \\ -S_{n+3} \\ \dots \\ -S_{2n} \end{bmatrix}.$$

Система связывает компоненты синдрома S_i с коэффициентами полинома локаторов ошибок, задача вычисления которых решается достаточно просто, если можно найти обратную матрицу синдромов.

2.3.1. Алгоритм Питерсона–Горенштейна–Цирлера (ПГЦ)

1. Вычисляются компоненты синдрома: $S_i = y(a^i)$, $i = 1, 2, \dots, 2t$. Задается величина $n = t$.

2. Составляется матрица синдромов $M_n = [S_{l+k+1}]$, $l, k = 0, 1, \dots, v-1$.

3. Вычисляется определитель матрицы $\Delta = \det M_n$. Если $\Delta = 0$, то матрица является сингулярной, и обратной матрицы не существует. В этом случае изменяется размер матрицы $v := (v - 1)$ до тех пор, пока определитель не станет отличным от нуля.

После чего повторяется второй шаг.

4. Определяются коэффициенты полинома локаторов ошибок:

$$\begin{bmatrix} s_n \\ s_{n-1} \\ s_{n-2} \\ \dots \\ s_1 \end{bmatrix} = M_n^{-1} \cdot \begin{bmatrix} -s_{n+1} \\ -s_{n+2} \\ -s_{n+3} \\ \dots \\ -s_{2n} \end{bmatrix}$$

5. Определяются корни $S(x) = 0$ и через их инверсию вычисляют значения X_i .

6. Оценивают величины ошибок $\{Q_i\}$, решая систему

$$\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ \dots \\ Q_m \end{bmatrix} = \begin{bmatrix} X_1 & \dots & X_m \\ X_1^2 & \dots & X_m^2 \\ X_1^3 & \dots & X_m^3 \\ \dots & \dots & \dots \\ X_1^m & \dots & X_m^m \end{bmatrix}^{-1} \cdot \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ \dots \\ s_m \end{bmatrix}.$$

7. Корректируют принятый вектор, подставляя вычисленные значения Q_i .

При количестве ошибок $t > 2$, данный алгоритм характеризуется большой вычислительной сложностью. В этих случаях для вычисления полинома локаторов ошибок используют алгоритмы Берлекэмп–Мессис, Сугиямы, а для оценки значений ошибок для q-ичных кодов – алгоритм Форни.

2.3.2. Алгоритм Сугиямы

Алгоритм Сугиямы позволяет решить систему уравнений в поле F

$$\begin{bmatrix} E_{t-1} & E_{t-2} & \mathbf{L} & E_0 \\ E_t & E_{t-1} & \mathbf{L} & E_1 \\ \mathbf{M} & \mathbf{M} & \mathbf{O} & \mathbf{M} \\ E_{2t-2} & E_{2t-3} & \mathbf{L} & E_{t-1} \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \mathbf{M} \\ s_t \end{bmatrix} = - \begin{bmatrix} E_t \\ E_{t+1} \\ \mathbf{M} \\ E_{2t-1} \end{bmatrix},$$

что разрешает центральную проблему декодирования локаторов ошибок.

Предполагаем, что $s_0 = 1$, полином локаторов ошибок $S(x) = \sum_{j=0}^t s_j x^j$ и

полином спектральных коэффициентов $\langle E_j \rangle$ функции ошибок $E(x) = \sum_{j=0}^{2t-1} E_j x^j$.

Считается, что коэффициенты полинома произведения $S(x)E(x)$ равны нулю для $j = t, \dots, 2t-1$.

Решение матричной системы уравнений эквивалентно решению полиномиального уравнения $S(x)E(x) = \Gamma(x) \bmod x^{2t}$ для степени полинома $S(x)$ не больше, чем t , и степени $\Gamma(x)$ не больше, чем $t-1$.

Рассматриваемый алгоритм решает полиномиальное уравнение с использованием алгоритма Евклида следующим образом. Обозначим начальные условия как $a^{(0)}(x) = x^{2t}$ и $b^{(0)} = E(x)$. Верхний индекс обозначает номер итерации (этапа) алгоритма. Для r -й итерации вычисления алгоритма Евклида можно записать

$$a^{(r-1)}(x) = \Theta^{(r)}(x)b^{(r-1)}(x) + b^{(r)} \quad \text{при } \deg(b^{(r)}(x)) < \deg(b^{(r-1)}(x)).$$

Определим условие выполнения равенства $a^{(r)}(x) = b^{(r-1)}(x)$:

$$\begin{bmatrix} a^{(r)}(x) \\ b^{(r)}(x) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -\Theta^{(r)}(x) \end{bmatrix} \begin{bmatrix} a^{(r-1)}(x) \\ b^{(r-1)}(x) \end{bmatrix}.$$

Матрицу $A^{(r)}(x)$ r -й итерации по индукции можно определить как

$$A^{(r)}(x) = \begin{bmatrix} 0 & 1 \\ 1 & -\Theta^{(r)}(x) \end{bmatrix} A^{(r-1)}(x) \quad \text{и} \quad A^{(0)}(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Для $r \geq 0$ имеем

$$\begin{bmatrix} a^{(r)}(x) \\ b^{(r)}(x) \end{bmatrix} = \begin{bmatrix} A_{11}^{(r)}(x) & A_{12}^{(r)}(x) \\ A_{21}^{(r)}(x) & A_{22}^{(r)}(x) \end{bmatrix} \begin{bmatrix} x^{2t} \\ E(x) \end{bmatrix}.$$

Откуда получаем желаемую форму уравнения

$$b^{(r)}(x) = A_{22}^{(r)}(x)E(x) \bmod x^{2t}, \quad S(x) \leftarrow A_{22}^{(r)}(x).$$

Для решения задачи декодирования требуется найти такое значение r , при котором $\deg(A_{22}^{(r)}(x)) \leq t$ и $\deg(b^{(r)}(x)) \leq t-1$. Это требование удобно выполнить, выбирая r^t как значения r , удовлетворяющие условиям $\deg(b^{(r^t-1)}(x)) \geq t$ и $\deg(b^{(r^t)}(x)) \leq t-1$.

2.3.3. Алгоритм Берлекэмп-Мессе

Позволяет найти полином локаторов ошибок. В основе алгоритма лежит адаптивная авторегрессионная модель генератора (рис.1), который формирует компоненты синдрома по правилу

$$S_r = - \sum_{j=1}^n S_j \cdot S_{r-j},$$

где n - количество произошедших ошибок; $r = n+1, \dots, 2n$

и перестраивает свою структуру в зависимости от возможной ошибки в структуре формируемого сигнала.

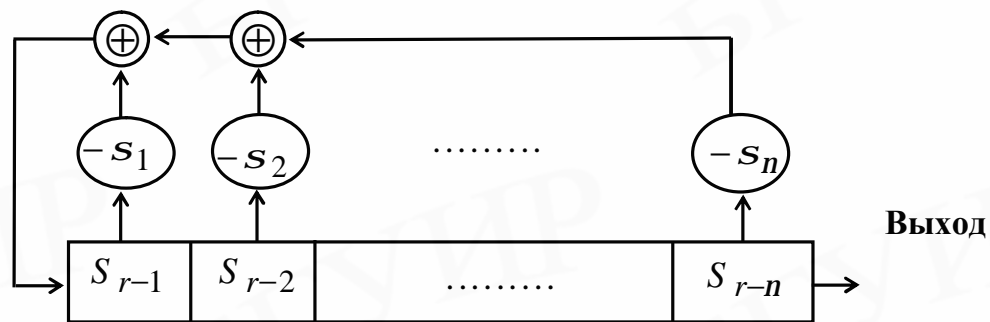


Рис. 1

Для нахождения полинома локатора ошибок используется итерационная процедура. На каждой итерации вычисляется модель регистра с обратными связями, генерирующего первые r компонентов синдрома, где r – номер этапа. Длина регистра на r -м этапе обозначается L_r . На r -м этапе оценка \hat{S}_r r -й компоненты синдрома равна

$$\hat{S}_r = - \sum_{j=1}^{L_r} s_j^{(r-1)} \cdot S_{r-j},$$

где $s_j^{(r-1)}$ – весовые коэффициенты на $(r-1)$ -м предыдущем этапе.

Ошибка (невязка) вычислений Δ_r определяется как

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

Если $\Delta_r = 0$, то параметры модели не меняются

$$(L_r, S^{(r)}(x)) = (L_{r-1}, S^{(r-1)}(x)).$$

Если $\Delta_r \neq 0$, то параметры модели меняются по правилу

$$S^{(r)}(x) = S^{(r-1)}(x) + A \cdot x^l \cdot S^{(m-1)}(x),$$

где $S(x)$ – веса; x^l – некоторое целое число; A – элемент поля.

Новая невязка может быть вычислена как

$$\Delta'_r = \sum_{j=0}^{L_r} s_j^{(r)} \cdot S_{r-j} = \sum_{j=0}^{L_r} s_j^{(r-1)} \cdot S_{r-j} + A \cdot \sum_{j=0}^{L_r} s_j^{(m-1)} \cdot S_{r-j-1}.$$

Определим величины A, m и r так, чтобы новая невязка Δ'_r была равна нулю.

Выберем следующие значения $m < r$ так, чтобы $\Delta_m \neq 0$; $l = r - m$ и

$A = -\Delta_m^{-1} \cdot \Delta_r = 0$. После подстановки получаем, что $\Delta'_r = \Delta_r - \frac{\Delta_r}{\Delta_m} \cdot \Delta_m = 0$. В этом

случае новая модель регистра сдвига будет генерировать последовательность S_1, \dots, S_{r-1}, S_r . Если выберем $L_m > L_{m-1}$, то получим регистр с обратной связью минимальной длины.

Теорема Берлекэмпа –Мессии. Пусть заданы компоненты синдрома S_1, \dots, S_{2t} из некоторого поля, тогда при начальных условиях $S^{(0)}(x) = 1$; $B^{(0)}(x) = 1$; $L_0 = 1$ выполняются следующие рекуррентные равенства, используемые для вычисления

$$S(x):$$

1. $\Delta_r = \sum_{j=0}^{n-1} S_j^{(r-1)} \cdot S_{r-j};$
2. $L_r = d_r \cdot (r - L_{r-1}) + (1 - d_r) \cdot L_{r-1};$
3. $\begin{bmatrix} S^{(r)}(x) \\ B^{(r)}(x) \end{bmatrix} = \begin{bmatrix} 1 & -\Delta_r \cdot x \\ \Delta_r^{-1} \cdot d_r & (1 - d_r) \cdot x \end{bmatrix} \cdot \begin{bmatrix} S^{(r-1)}(x) \\ B^{(r-1)}(x) \end{bmatrix},$

где $r = 1, 2, \dots, 2t$ и $\begin{cases} d_r = 1, & \text{если одновременно } \begin{cases} \Delta_r \neq 0, \\ 2L_{r-1} \leq r-1. \end{cases} \\ d_r = 0 & \text{в остальных случаях.} \end{cases}$

При выполнении таких условий полином $S(x)$ является многочленом наименьшей степени, коэффициенты которого удовлетворяют равенству

$$S_j + \sum_{j=1}^{n-1} S_j^{(2t)} \cdot S_{r-j} = 0, \text{ где } r = L_{2t} + 1, \dots, 2t.$$

Граф-схема алгоритма Берлекэмпа–Мессии приведена на рис. 2.

Процедура Ченя. Для вычисления инверсии корней полинома ошибок удобно использовать следующий алгоритм. Воспользуемся тем свойством, что

сумма $\sum_{k=0}^n S_k \cdot a^{-ik} = 0$ только в том случае, если символ, располагающийся на

$(n-i)$ -й позиции, оказывается ошибочным. Если определить $S_k^{(i)} = S_k \cdot a^{-ik}$, тогда

$$S_k^{(i-1)} = S_k^{(i)} \cdot a^k, \text{ а } S(a^{-i}) = \sum_{k=0}^n S_k^{(i)}.$$

В итоге получается простая переборная схема вычислителя корней (рис.3).

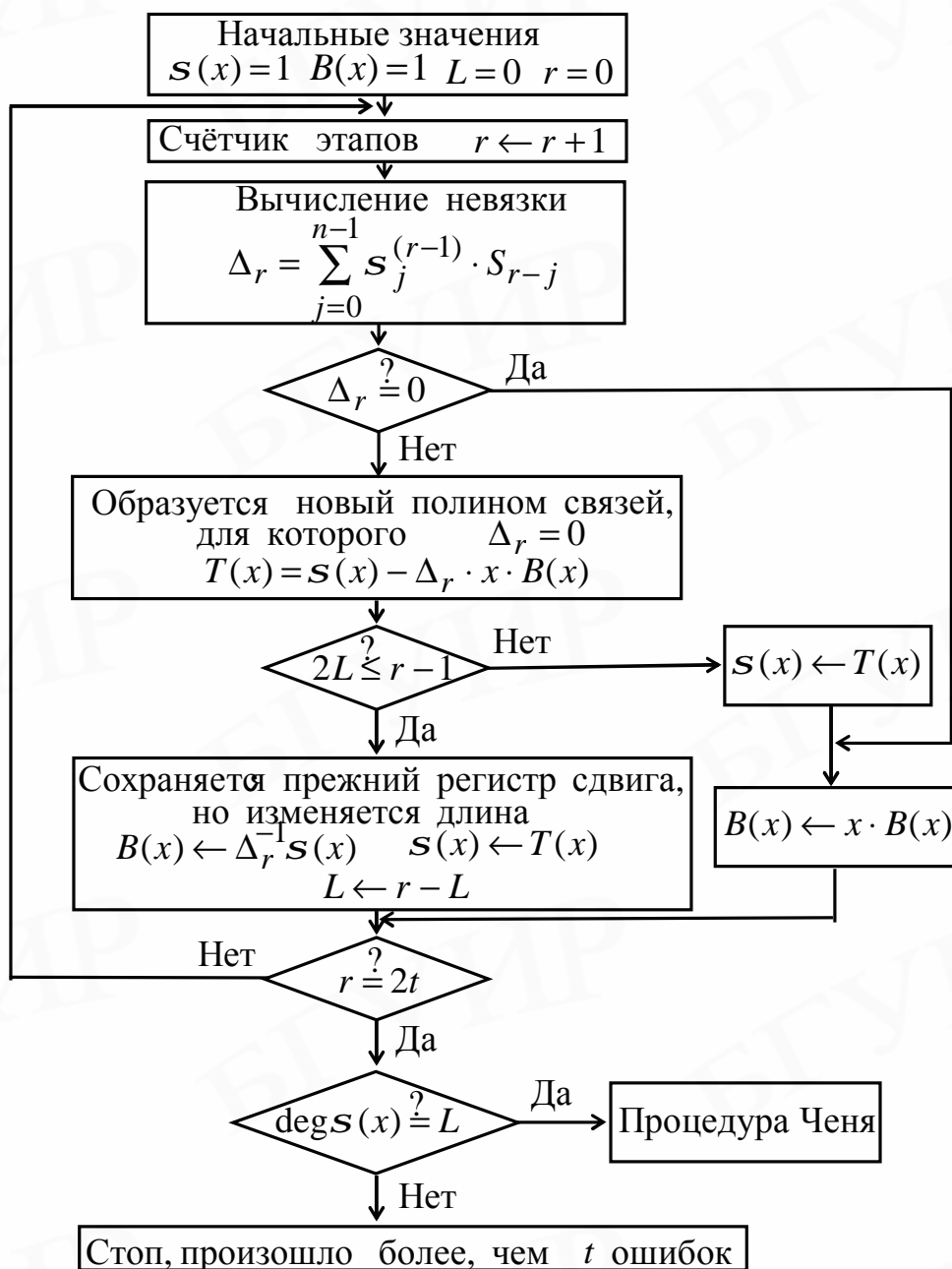


Рис. 2

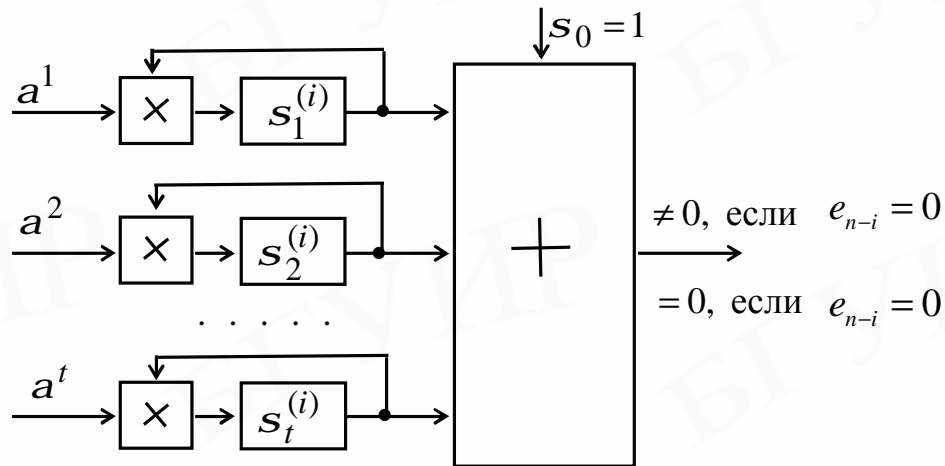


Рис. 3

Алгоритм Форни. Позволяет оценить величины ошибок Q_i . Известно, что

$$S(x) = \sum_{j=1}^{n=2t} S_j \cdot x^j = \sum_{j=1}^{2t} \sum_{i=1}^n Q_i \cdot X_i^j \cdot x^j.$$

Полином локатора ошибок имеет вид

$$S(x) = \sum_{j=0}^n S_j \cdot x^j = \prod_{l=1}^n (1 - x \cdot X_l),$$

где X_l^{-1} - корни полинома. Составим ключевое уравнение

$\Gamma(x) = (1 + S(x)) \cdot S(x) \bmod x^{2t+1}$. Если в ключевое уравнение подставить выражения для $S(x)$, $\sigma(x)$ и решить его относительно Q_i , то получим оценку величины ошибки:

$$Q_l = \frac{-X_l \cdot \Gamma(X_l^{-1})}{S'(X_l^{-1})} \bmod x^{2t+1},$$

где S' - формальная производная полинома.

Пример. Троичный БЧХ-код (7, 3) формируется над полем $GF(8) = GF(2^3)$, $f(x) = x^3 + x + 1$, для которого $a^0 \rightarrow 100$; $a^1 \rightarrow 010$; $a^2 \rightarrow 001$; $a^3 \rightarrow 110$; $a^4 \rightarrow 011$; $a^5 \rightarrow 111$; $a^6 \rightarrow 101$. Порождающий полином имеет вид

$$g(x) = x^4 + a^3 \cdot x^3 + x^2 + a \cdot x + a^3.$$

Предположим, что на вход декодера поступает сигнал

$$y(x) = a^2 \cdot x^6 + a^2 \cdot x^4 + x^3 + a^5 \cdot x^2,$$

компоненты синдрома ошибок равны $S_1 = a^6$; $S_2 = a^3$; $S_3 = a^4$; $S_4 = a^3$.

Алгоритм Берлекэмп–Мессе вычисляет полином локатора ошибок:

$$s(x) = 1 + a^2 \cdot x + a \cdot x^2, \quad s'(x) = a^2 + 2a \cdot x = a^2,$$

который имеет следующие корни: $l_1 = a^4$, $l_2 = a^2$. Тогда номера позиций ошибок определяются как 3 и 5. Ключевое уравнение запишется как

$$w(x) = (1 + a^2 \cdot x + a \cdot x^2) \cdot (1 + a^6 \cdot x + a^3 \cdot x^2 + a^4 \cdot x^3 + a^3 \cdot x^4) \bmod x^{(2t+1)=5} = \\ = 1 + x + a^3 \cdot x^2.$$

Откуда

$$Q_l = \frac{-X_l \cdot w(X_l^{-1})}{s'(X_l^{-1})} = \frac{-X_l \cdot (1 + X_l^{-1} + a^3 \cdot X_l^{-2})}{a^2} = X_l \cdot a^5 + a^5 + a \cdot X_l^{-1};$$

$$Q_3 = Q_3(X_1 = a^3) = a; \quad Q_5 = Q_3(X_2 = a^5) = a^5.$$

Оценка вектора ошибки равна $\hat{e}(x) = a \cdot x^3 + a^5 \cdot x^5$. Оценка кодового слова равна

$$y(x) - \hat{e}(x) = a^2 \cdot x^6 + a^5 \cdot x^5 + a^2 \cdot x^4 + a^3 \cdot x^3 + a^5 \cdot x^2 = \hat{c}(x).$$

2.4. Нумератор весов кода

Рассмотрим весовые характеристические функции линейного кода, содержащие полную информацию о структуре кода. Используются каналы, вносящие независимые ошибки в передаваемые символы с вероятностью p_0 . Безошибочно символы принимаются с вероятностью $(1 - p_0)$.

Пусть задан линейный (n, k) -код, символы которого принадлежат q -ичному множеству из $GF(q)$, и пусть этот код содержит A_i векторов весом i . Тогда совокупность чисел $A_0, \dots, A_i, \dots, A_n$ называют распределением весов, а многочлен

$$A(z) = \sum_{i=0}^n A_i \cdot z^i \text{ называют характеристической функцией или нумератором весов}$$

кода C . На рис. 4 приведено распределение нумераторов веса для кода БЧХ.

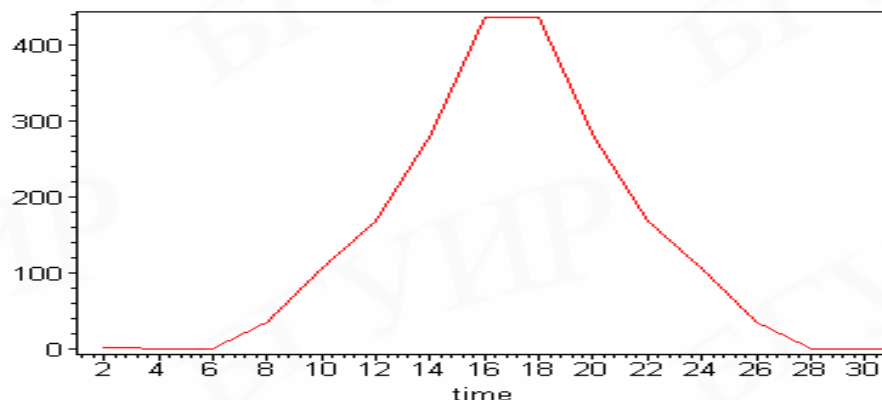


Рис. 4

Если выполняется неравенство $2t + 1 \leq d_{\min}$ и декодер исправляет все конфигурации ошибок, вес которых не превышает t , то вероятность ошибочного декодирования равна

$$P_e = \sum_{h=0}^n [p_0/(1-q)]^h (1-p_0)^{n-h} \sum_{s=0}^t \sum_{l=1}^n A_l N(l, h, s),$$

где $N(l, h, s)$ – число конфигураций ошибок весом h , находящихся на расстоянии s от кодового слова веса l .

3. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Выполнение работы проводится с использованием программной среды Maple. Программное обеспечение содержит программы, моделирующие алгоритмы, необходимые для кодирования, исследования свойств и декодирования кодов БЧХ.

3.1. Программы кодирования

CyclotomicCoset - вычисляет циклотомический класс;

AllCyclotomicCoset - вычисляет множества циклотомических классов;

IrredPoly - выбирает неприводимый полином степени r над конечным полем путем факторизации полинома $x^{p^r-1}-1 \bmod p$;

MinPoly - вычисляет минимальный полином β^s , где β является примитивным элементом поля $GF(p^m)$, $m = \deg(f)$ и корнем полинома f ;

BCHGenPoly - вычисляет генераторный полином БЧХ-кода длиной $n=2^m-1$, с требуемым расстоянием $2t+1$ над полем $F(p)$. Полином $f(x)$ должен быть неприводимым и его корни должны быть примитивными элементами для $GF(p^m)$

ToVector - конвертирует форму полинома в вектор коэффициентов длины n ;

ToPoly – преобразует форму вектора в полином с коэффициентами, равными элементам вектора;

CyclicGenMat - формирует генераторную матрицу циклического кода длиной n с генераторным полиномом g (если он существует), необходимое условие - полином g должен делить (x^n-1) ;

VMP - вычисляет произведение вектора v на матрицу A (vA) $\bmod p$;

GenMat - вычисляет генераторную матрицу ступенчатой формы;

CheckMat - вычисляет проверочную матрицу "стандартной" формы.

3.2. Программы для исследования свойств кодов

HamWeight - вычисляет вес Хэмминга исследуемого кода;

WeightEnumerator - вычисляет нумератор весов кода, образованного строками матрицы G по $\bmod p$, над полем $F(p)$;

ListCode - вычисления списка кодовых слов для кода, образованного строками матрицы G по $\bmod p$, над $F(p)$;

syndroom – вычисляет компоненты синдрома ошибок (спектр Фурье принимаемого сигнала).

3.3. Программы декодирования кодов БЧХ

bm – вычисляет полином локаторов ошибок по алгоритму Берлекэмп–Мессис;
sugiyama - вычисляет полином локаторов ошибок по алгоритму Сугияма;
syndroom – вычисляет компоненты синдрома ошибок;
forney - вычисляет оценки (амплитуды) ошибок по алгоритму Форни, для q –
ичного кода БЧХ.

3.3.1. Программа декодирования БЧХ - кода по алгоритму ПГЦ

Комментарии. Программа декодирования “ВСН” определяет корректность кодового слова и обрабатывает входной сигнал, представленный в виде бинарного n -кода или полинома. При этом первый элемент n -кода соответствует свободному члену полинома. Процедура декодирования предварительно проверяет корректность принимаемого кода и если она нарушена, то определяет местоположение ошибки.

Программа работает с конечным полем F над кольцом $K[x]/(p(x))$, где $p(x)$ неприводимый полином над K . Если поле содержит n ненулевых элементов и если t положительное целое число, тогда код состоит из всех полиномов степени меньше, чем n , для которых степени $(2*t)$ примитивного элемента альфа являются корнями, $n = 2^{\deg(p)} - 1$. В процессе работы полином p и целое число t необходимо ввести в память программы. Примитивный элемент поля находится с помощью процедуры **isprimitive**, которая проверяет, является ли данный элемент примитивным и вводит примитивный элемент в память программы под именем **alpha**. Целое число t не может быть больше чем $n/2$. Процедура вычисления позиций ошибок выдает результат в полиномиальном виде: так, если ошибки расположены на 2 и 5 позициях, то полином ошибок равен $x^2 + x^5$.

Процедура **generator** формирует генераторный полином БЧХ - кода для заданного полинома $p(x)$. Процедура задается в виде **generator(t)**, где t число исправляемых ошибок.

Программа декодирования

> **restart: with(linalg):**

*Программа "isprimitive" осуществляет нахождение и проверку
примитивного элемента alpha*

> **isprimitive := proc(f) local i,j,q,n,FAC,test: global alpha:**

> **test := 0:**

> **n := 2^degree(p)-1:**

> **readlib(ifactors):**

> **FAC := ifactors(n):**

> **for j from 1 to nops(FAC[2]) do**

> **q := FAC[2][j][1]:**

```

> if Rem(f^(n/q),p,x) mod 2 = 1 then
>   printf("%A не является примитивным элементом Z2[x]/( %A)",f,p):
>   test := 1:
>   break: fi: od:
> if test = 0 then
>   printf("%A является примитивным элементом Z2[x]/( %A)",f,p):
>   alpha := f:
>   printf("\n альфа = %A",f): fi: end:

```

Программа "BCH": процедура декодирования БЧХ - кода

```

> BCH:=proc(rr) local c,cc,i,j,l,n,r,v,A,B,C,D,S,SS,T,inv,abc,remm,count,beta:

```

Инициализация переменных

```

> beta := 'beta':
> r := 0:
> remm := f->Rem(f,p,x) mod 2:
> n := 2^degree(p)-1:
> count := 0:

```

Конвертация rr в полиномиальную форму (если это необходимо)

```

> if type(rr,list) = false then
>   r := rr:
> else
>   for i from 1 to n do
>     r := r + rr[i]*x^(i-1): od: fi:
> printf("%A", "Принятый полином равен"):
> print(sort(r)):

```

Вычисление оценки r как степени альфа

```

> printf("Синдромы s_1 = s_2t \n"):
> S := array(1..2*t):
> for i from 1 to 2*t do
>   S[i] := Rem(subs(x=alpha^i,r),p,x) mod 2:
>   SS[i] := subs(x=beta,S[i]):
>   if S[i] <> 0 then
>     count := 1: fi: od:
> S:=convert(S,list):
> print(convert(SS,list)):
> printf("\n"):
> if count = 0 then
>   printf("%A", "Принятое кодовое слово не содержит ошибок"):
> else
>   A:=array(1..t,1..t):

```

```

> for i from 1 to t do
>   for j from 1 to t do
>     A[i,j] := S[i+j-1]: od: od:

```

Определение v , количества ошибок

```

> D := Rem(det(A),p,x) mod 2:
> v := t:
> while D = 0 and v > 1 do
>   A := delrows(A,v..v):
>   A := delcols(A,v..v):
>   D := Rem(det(A),p,x) mod 2:
>   v := v-1: od:
> if D <> 0 then
>   printf("%A","Количество ошибок равно"):
>   print(v):

```

Вычисление полинома позиций ошибок

```

> T := S[v+1..2*v]:
> T := convert(T,vector):
> Gcdex(D,p,x,'inv','abc') mod 2:
> B := map(`remm`,evalm(inv*adj(A))):
> C := map(`remm`,evalm(B&*T)):
> l := 0:
> for i from 1 to v do
>   l := l + C[i]*z^(i-1): od:
> l := l+z^v:
> c := r:
> printf("%A","Полином позиций ошибок равен"):
> print(sort(subs(x=beta,l),z)):
> printf("%A","Позиции ошибок равны"):
> count := 0:
> for i from 0 to n-1 do
>   if Rem(subs(z=alpha^i,l),p,x) mod 2 = 0 then
>     print(i):
>     c := c+x^i:
>     count := count + 1:
>   fi: od:
> if count = v then
>   printf("%A","Скорректированный полином равен"):
>   print(sort(c mod 2)):
> else

```

```

> printf("\n%A\n%A","Полином позиций ошибок имеет слишком много
различных корней,", "имеется слишком много ошибок для исправления");
> fi:
> else printf("\n%A","Существует слишком много ошибок для исправления"):
fi: fi: end:

```

Программа "generator": нахождение генераторного полинома БЧХ - кода

```

> generator:=proc(t) local i,j,f,g,n,lc,Polys:
> n := degree(p):
> Polys := Factors(x^(2^n)-x) mod 2:
> for j from 1 to nops(Polys[2]) do
> f[j] := Polys[2][j][1]: od:
> for i from 1 to 2*t do
> for j from 1 to nops(Polys[2]) do
> if Rem(subs(x=x^i,f[j]),p,x) mod 2 = 0 then g[i] := f[j]: fi:
> od: od:
> lc := g[1]:
> for i from 2 to 2*t do
> lc := Lcm(g[i],lc) mod 2: od:
> sort(lc): end:

```

Пример

```

> p:=x^4+x+1;
> t:=3;
> BCH([1,1,1,1,1,1,1,0,0,1,0,0,0]);

```

$$p := x^4 + x + 1$$

$$t := 3$$

Принятый полином равен

$$x^{11} + x^9 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$

Синдромы $s_1 = s_{2t}$

$$[\text{beta}, \text{beta}^2, \text{beta}^3 + \text{beta}^2 + \text{beta}, 1 + \text{beta}, 1 + \text{beta} + \text{beta}, \text{beta}^3 + \text{beta} + 1]$$

Количество ошибок равно

$$2$$

Полином позиций ошибок равен

$$z^2 + \text{beta} z + 1 + \text{beta}$$

Позиции ошибок равны

$$0$$

$$4$$

Скорректированный полином равен

$$x^{11} + x^9 + x^6 + x^5 + x^3 + x^2 + x$$

4. СОДЕРЖАНИЕ И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

4.1. Содержание работы

1. Изучить принцип построения, свойства и алгоритмы декодирования БЧХ-кодов.
2. Изучить правила пользования программного пакета Maple и программного обеспечения к лабораторной работе.
3. Провести моделирование с помощью программного обеспечения алгоритмов формирования БЧХ кодов в полиномиальной и матричной форме представления.
4. Провести моделирование алгоритмов декодирования кодов БЧХ.
5. Провести анализ спектральных и весовых характеристик кодов.
6. Оценить достоверность передачи информации с помощью БЧХ-кодов .

4.2. Порядок выполнения работы

Составить и отладить моделирующие алгоритмы программного обеспечения.

4.2.1. Моделирование алгоритмов формирования БЧХ-кодов

1. Получить от преподавателя исходные данные кодов, построить минимальные полиномы и циклотомические классы, необходимые для кодирования сообщений.
2. Задавая различные виды сообщений, провести кодирование информации различными методами, используя матричную, полиномиальную и символьную формы представления информации.

4.2.2. Исследование весовых характеристик кода и оценка достоверности передачи информации

1. Вычислить распределение весов и определить нумератор весов исследуемых кодов.
2. Оценить вероятность ошибки при заданной вероятности искажения символов в канале передачи информации.

4.2.3. Моделирование алгоритма декодирования ПГЦ

1. Используя данные п. 4.2.1, провести моделирование аддитивного канала добавив к кодовым словам векторы ошибок.
2. Провести декодирование полученных сигналов с помощью алгоритма ПГЦ.
3. Получив от преподавателя контрольные коды, декодировать их методом ПГЦ.

4.2.4. Моделирование алгоритмов Берлекэмпа–Месси и Сугиямы для нахождения полинома локаторов ошибок

1. Используя данные п.4.2.2, вычислить полиномы локаторов ошибок, используя алгоритмы Берлекэмпа–Месси и Сугиямы.
2. Определить местоположение ошибок в обрабатываемых сигналах.
3. Оценить эффективность алгоритмов вычисления.

4.2.5. Моделирование алгоритма Форни оценки значений ошибок

1. Получить от преподавателя исходные данные для декодирования многозначного кода БЧХ.

2. Провести моделирование алгоритма Форни, вычислив значения ошибок по исходным данным.

5. СОДЕРЖАНИЕ ОТЧЕТА

1. Формулировка цели работы.
2. Схемы алгоритмов формирования и декодирования кодов.
3. Результаты моделирования.
4. Анализ свойств исследуемых кодов и их влияние на эффективность декодирования.
5. Оценка достоверности передачи информации с помощью БЧХ - кодов.
6. Выводы.

6. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Пояснить сущность метода формирования циклических кодов с помощью элементов конечного поля.
2. Что такое циклотомические классы и как они влияют на способность кода исправлять ошибки ?
3. Как выбрать минимальные полиномы для формирования БЧХ-кода, обеспечивающего коррекцию t ошибок и заданную скорость передачи информации?
4. Как связаны функция синдрома ошибки и спектральная характеристика принимаемого сигнала, закодированного кодом БЧХ?
5. Поясните принципы работы алгоритмов Сугиямы и Берлекэмпа–Мессе.
6. Как оценить значение ошибки из ключевого уравнения?

ЛИТЕРАТУРА

1. Касами Т., Токура Н., Ивадари Ё., Инагаки Я. Теория кодирования: Пер. с яп.- М.: Мир, 1978.
2. Блейхут Р. Теория и практика кодов, контролирующих ошибки: Пер. с англ. - М.: Мир, 1986.
3. Лосев В. В. Помехоустойчивое кодирование в радиотехнических системах передачи информации. Ч.2. – Мн.: МРТИ , 1984.
4. Кларк Дж., Кейн Дж. Кодирование с исправлением ошибок в системах цифровой связи: Пер. с англ.-М.: Радио и связь, 1987.
5. Мак-Вильямс Ф. Дж., Слоэн Н.Дж. Теория кодов, исправляющих ошибки. – М.: Связь, 1979.
6. Склар Б. Цифровая связь. Теоретические основы и практическое применение.: Пер. с англ.-М.: Изд. дом «Вильямс», 2003.-1104 с.

Учебное издание

Саломатин Сергей Борисович

**ИССЛЕДОВАНИЕ СВОЙСТВ КОДОВ
БОУЗА–ЧОУДХУРИ–ХОКВИНГВЕМА**

МЕТОДИЧЕСКОЕ ПОСОБИЕ
к лабораторной работе по дисциплине
КОДИРОВАНИЕ И ЗАЩИТА ИНФОРМАЦИИ
для студентов специальностей
39 01 02 «Радиоэлектронные системы»
и 39 01 03 «Радиоинформатика»
дневной формы обучения

Редактор Т.Н. Крюкова
Корректор Е.Н. Батурчик
Компьютерная верстка М.В. Шишло

Подписано в печать 14.10.2004.
Гарнитура «Таймс».
Уч.-изд. л. 1,0.

Формат 60x84 1/16.
Печать ризографическая.
Тираж 100 экз.

Бумага офсетная.
Усл. печ. л. 1,63.
Заказ 139.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
Лицензия на осуществление издательской деятельности №02330/0056964 от 01.04.2004.
Лицензия на осуществление полиграфической деятельности №02330/0133108 от 30.04.2004.
220013, Минск, П. Бровки, 6