Лабораторная работа №5 “Циклические коды”

Индивидуальное задание, вариант №8

1)

Согласно параметрам заданного кода:

n = 15 - общее число элементов;

m = 11 - число информационных элементов;

k = 4 - число избыточных элементов;

d_{min} = 3,

Порождающий многочлен:

Символическая запись: 23

Двоичная запись: 10011

то возможно закодировать двоичные числа от нуля до $2^{11} = 2048$. Кодовые слова найдём умножением векторов-строк (1,11) на порождающую матрицу G размером (11,15)

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

Программный код:

```
import sys
import numpy as np
from tabulate import tabulate
np.set_printoptions(threshold=sys.maxsize)
g = [int(i) for i in "10011"]  # порождающий многочлен
n = 15  # общее число элементов
m = 11  # число информационных элементов
length = len(g)
for i in range(n - length):
    g.append(0)
G0 = []  # временная матрица
for i in range(m):
    G0.append(np.roll(g, i))
G = np.array(G0)  # порождающая матрица
codewords_table_headers = ["Информационное слово", "Кодовое слово"]
codewords_table = []
length = pow(2, m)
for i in range(0, length):
    d = np.array([int(i) for i in np.binary_repr(i, m)])
    codewords_table.append([''.join(map(str, d.tolist())), ''.join(map(str, np.mod(d.dot(G), 2)))])
with open('output1.txt', 'w') as file:
    file.write("Разрешенные кодовые комбинации: ")
    file.write("\n")
    file.write(tabulate(codewords_table, codewords_table_headers, tablefmt="grid",
numalign='center'))
    file.close()
```

(смотреть output1.txt
в этом файле записаны фрагменты таблицы)

Программный код:

```
import sys
import numpy as np
import itertools
np.set_printoptions(threshold=sys.maxsize)
def hammingDist(str1, str2, fillchar='-'):
    return sum([ch1 != ch2 for (ch1, ch2) in itertools.zip_longest(str1, str2, fillvalue=fillchar)])
g = [int(i) for i in "10011"]  # порождающий многочлен
n = 15  # общее число элементов
m = 11  # число информационных элементов
length = len(g)
for i in range(n - length):
    g.append(0)
G0 = []  # временная матрица
for i in range(m):
    G0.append(np.roll(g, i))
G = np.array(G0)  # порождающая матрица
codewords_table = np.array("00000000000")
length = pow(2, m)
for i in range(1, length):
    d = np.array([int(i) for i in np.binary_repr(i, m)])
    codewords_table = np.append(codewords_table, ''.join(map(str, np.mod(d.dot(G), 2))))
length = len(codewords_table)
hamming_distances_table = np.zeros((length, length), dtype=int)
with open('output2.txt', 'w') as file:
    file.write("Таблица кодовых расстояний:\n")
    file.write("\n")
    for i in range(0, length):
        for j in range(i + 1, length):
            hamming_distances_table[i][j] = hammingDist(codewords_table[i], codewords_table[j])
            hamming_distances_table[j][i] = hamming_distances_table[i][j]
        file.write(str(hamming_distances_table[i]))
        file.write("\n")
    file.write("\n")
    file.write("Минимальное кодовое расстояние:\n")
    file.write(str(np.amin(np.where(hamming_distances_table == 0, 100,
hamming_distances_table))))
    file.close()
```

(смотреть output2.txt
в этом файле записаны фрагменты таблицы)

2)
Определим характеристики заданного кода в режиме исправления ошибок:
а)
Определим кратность гарантированно исправляемых кодом ошибок
Код исправляет в сообщении t ошибок, если кодовое расстояние d_{min} не меньше, чем 2t+1, то есть $d_{min} \geq 2t + 1$

$d_{min} = 3 \geq 2t + 1 => t = 1$– кратность гарантированно исправляемых кодом ошибок
б)
Найдем число различных векторов ошибок, которые код может исправить
Общее число различных векторов ошибок, которые может исправить циклический код, равно $2^{n-m} = 2^k$. 2^4=16 – число различных векторов ошибок, которые может исправить данный код
в)
Для одного из векторов ошибок, исправляемых кодом, найдем соответствующий этому вектору синдром (Синдромом ошибки в этих кодах является наличие остатка от деления принятой кодовой комбинации на производящий полином. Если синдром равен нулю, то считается, что ошибок нет. В противном случае, с помощью полученного синдрома можно определить номер разряда принятой кодовой комбинации, в котором произошла ошибка, и исправить ее). Найдем несколько из возможных векторов ошибок, при декодировании которых получается тот же синдром, и, следовательно, происходит ошибочное декодирование

$e \epsilon \{0, 1\}^n$ - вектор ошибки:
e_i ={1 (в i-ом бите произошла ошибка); 0 (ошибки нет) }

Пусть вектор ошибки e = 100000000000000, то есть ошибка в 15-ом бите
Рассмотрим кодовое слово: $u_{14}(x) =$ 011000010111100

Рассмотрим кодовое слово: $u'_{14} =$ 011000010111100

Найдём синдром ошибки:

1. $u'_{14}(x) = q(x)g_4(x) \oplus s(x)$(Операция деления является обычным делением многочленов с остатком, только вместо вычитания используется сложение по модулю 2 (mod 2))

Найдем $u'_{14}(x)$: $g_4(x)$ $(g_4(x) = x^4 + x + 1)$

$$
\begin{array}{l|l}
x^{14}+x^{13}+x^{12}+x^{7}+x^{5}+x^{4}+x^{3}+x^{2} & x^{4}+x+1 \\
\hline
\oplus\ x^{14}+x^{11}+x^{10} & x^{10}+x^{9}+x^{8}+x^{7}+x+1 \\
\hline
x^{13}+x^{12}+x^{11}+x^{10}+x^{7}+x^{5}+x^{4}+x^{3}+x^{2} & \\
\oplus\ x^{13}+x^{10}+x^{9} & \\
\hline
x^{12}+x^{11}+x^{9}+x^{7}+x^{5}+x^{4}+x^{3}+x^{2} & \\
\oplus\ x^{12}+x^{9}+x^{8} & \\
\hline
x^{11}+x^{8}+x^{7}+x^{5}+x^{4}+x^{3}+x^{2} & \\
\oplus\ x^{11}+x^{8}+x^{7} & \\
\hline
x^{5}+x^{4}+x^{3}+x^{2} & \\
\oplus\ x^{5}+x^{2}+x & \\
\hline
x^{4}+x^{3}+x & \\
\oplus\ x^{4}+x+1 & \\
\hline
x^{3}+1 &
\end{array}
$$

2. $e(x) = q(x)g_4(x) \oplus s(x)$

Найдем $e(x)$: $g_4(x)$

$$
\begin{array}{l|l}
\oplus\ x^{14} & x^{4}+x+1 \\
\hline
x^{14}+x^{11}+x^{10} & x^{10}+x^{7}+x^{6}+x^{4}+x^{2}+x+1 \\
\hline
\oplus\ x^{11}+x^{10} & \\
\oplus\ x^{11}+x^{8}+x^{7} & \\
\hline
x^{10}+x^{8}+x^{7} & \\
\oplus\ x^{10}+x^{7}+x^{6} & \\
\hline
x^{8}+x^{6} & \\
\oplus\ x^{8}+x^{5}+x^{4} & \\
\hline
x^{6}+x^{5}+x^{4} & \\
\oplus\ x^{6}+x^{3}+x^{2} & \\
\hline
x^{5}+x^{4}+x^{3}+x^{2} & \\
\oplus\ x^{5}+x^{2}+x & \\
\hline
x^{4}+x^{3}+x & \\
\oplus\ x^{4}+x+1 & \\
\hline
x^{3}+1 &
\end{array}
$$

Результат: s(x)=$x^3 + 1$=1001

3) Определим возможности заданного кода в режиме обнаружения ошибок:

а) Определим кратность σ гарантированно обнаруживаемых кодом ошибок

Код обнаруживает в сообщении t ошибок, если кодовое расстояние d_{min} не меньше, чем t+1, то есть $d_{min} \geq t + 1$

$d_{min} = 3 \geq t + 1 => t = 2$– кратность гарантированно обнаруживаемых кодом ошибок

б) Найдём вектора ошибок, которые не могут быть обнаружены заданным кодом.
Так как слова любого линейного кода обладают свойством замкнутости по отношению к операции сложения, то есть сумма двух и более кодовых слов тоже является кодовым словом, то векторы ошибок, совпадающие с кодовыми словами, не могут быть обнаружены декодером циклического кода.