



**Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

**Факультет “Информатика и системы управления”
Кафедра “Системы обработки информации и управления”**

**Отчет по домашнему заданию №2
по дисциплине
Сети и телекоммуникации**

**Выполнил:
студент группы
ИУ5-54Б
Шараев В.А.**

**Проверил:
Галкин В.А**

2025 г.

Вариант 21

Требуется, используя кодирование кодом Хэмминга $X [7,4]$, определить корректирующую способность этого кода C_k .

21	1011	$X [7,4]$	C_k
----	------	-----------	-------

1. Метод решения задачи для варианта задания

Для оценки корректирующей способности кода необходимо перебрать все возможные векторы ошибок, сгруппировав их по кратности (числу единиц i).

Имитируя канал связи, передаём закодированное сообщение (для информационного вектора 1011 — кодовое слово 0110011), накладывая последовательно все вектора ошибок каждой кратности.

Имитируя приёмник, вычисляем синдром ошибки. Если синдром ненулевой, пытаемся исправить ошибку, инвертируя бит в позиции, указанной синдромом. После исправления проверяем, совпадает ли полученное кодовое слово с исходным. Если совпадает — ошибка исправлена (увеличиваем счётчик N_k для данной кратности).

После перебора всех ошибок кратности i вычисляем $C_k = \frac{N_k}{C_n^i}$, где $C_n^i = \binom{7}{i}$

2. Модель канала связи. Алгоритмы кодирования, декодирования, вычисления корректирующей способности кода для ошибок всех возможных кратностей

Передатчик имитируется вычислением кодового слова Хэмминга для фиксированного информационного вектора $1011 \rightarrow 0110011$.

Канал связи имитируется наложением (XOR) векторов ошибок всех возможных кратностей.

Приёмник имитируется вычислением синдрома и попыткой исправления одиночной ошибки (стандартный декодер Хэмминга $[7,4]$).

Алгоритм кодирования кодом Хэмминга $[7,4]$:

1. Информационный вектор 1011 размещается в позициях 3,5,6,7.
2. Вычисляются проверочные биты в позициях 1,2,4 по чётности соответствующих групп.
3. Результат: кодовое слово 0110011.

Алгоритм декодирования кодом Хэмминга $[7,4]$:

1. Для принятого вектора g вычисляется синдром E (3 бита):
 $h1 = \oplus$ битов в позициях с 1 в младшем бите номера (1,3,5,7)
 $h2 = \oplus$ битов в позициях с 1 во втором бите (2,3,6,7)
 $h3 = \oplus$ битов в позициях с 1 в старшем бите (4,5,6,7)
2. Если $E = 0$ — ошибка отсутствует или не обнаружена.
3. Если $E \neq 0$ — инвертировать бит в позиции E (десятичное значение).
4. Полученное слово считается исправленным

Алгоритм вычисления корректирующей способности кода C_k для ошибок всех возможных кратностей:

Формула: $C_k^i = \frac{N_k}{C_7^i}$

По определению: корректирующая способность кода C_k для ошибок кратности i — отношение числа исправленных ошибок N_k к общему числу ошибок данной кратности $\binom{7}{i}$.

Для каждой кратности i вводится счётчик $N_k = 0$, который увеличивается на 1, если после декодирования получено исходное кодовое слово.

Алгоритм повторяется для i от 1 до 7, формируется таблица с i , $\binom{7}{i}$, N_k и C_k (в долях единицы или процентах).

Реализация на Python:

```
import math

from itertools import combinations
```

```
def hamming_encode(info):
```

```
    data = [int(b) for b in info]
```

```
    k = len(data)
```

```
    r = 0
```

```
    while 2 ** r < k + r + 1:
```

```
        r += 1
```

```
    n = k + r
```

```
    code = [0] * n
```

```
    j = 0
```

```
    for i in range(1, n + 1):
```

```
        if i & (i - 1) != 0:
```

```
            code[i - 1] = data[j]
```

```
            j += 1
```

```

for p in range(r):

    pos = 2 ** p

    parity = 0

    for i in range(1, n + 1):

        if i & pos:

            parity ^= code[i - 1]

    code[pos - 1] = parity

return ".join(map(str, code))

```

```

def hamming_syndrome(received):

    received = [int(b) for b in received]

    n = len(received)

    r = 0

    while 2 ** r <= n:

        r += 1

    syndrome = 0

    for p in range(r):

        pos = 2 ** p

        parity = 0

        for i in range(1, n + 1):

            if i & pos:

                parity ^= received[i - 1]

        if parity:

            syndrome += pos

    return syndrome

```

```
info_vector = '1011'
```

```
v = hamming_encode(info_vector)
```

```
print("Кодовое слово:", v)
```

```
n = 7
```

```
print("i\tCin\tNo\tCo\tNk\tCk")
```

```
for i in range(1, n + 1):
```

```
    Cin = math.comb(n, i)
```

```
    Nk = 0
```

```
    No = 0
```

```
    for err_pos in combinations(range(n), i):
```

```
        r_list = list(v)
```

```
        for p in err_pos:
```

```
            r_list[p] = '1' if r_list[p] == '0' else '0'
```

```
        r = ''.join(r_list)
```

```
        synd = hamming_syndrome(r)
```

```
    if synd != 0:
```

```
        No += 1
```

```
    corrected = r
```

```
    if 1 <= synd <= n:
```

```
        corr_list = list(r)
```

```
        corr_list[synd - 1] = '1' if corr_list[synd - 1] == '0' else '0'
```

```
        corrected = ''.join(corr_list)
```

else:

corrected = r

if corrected == v:

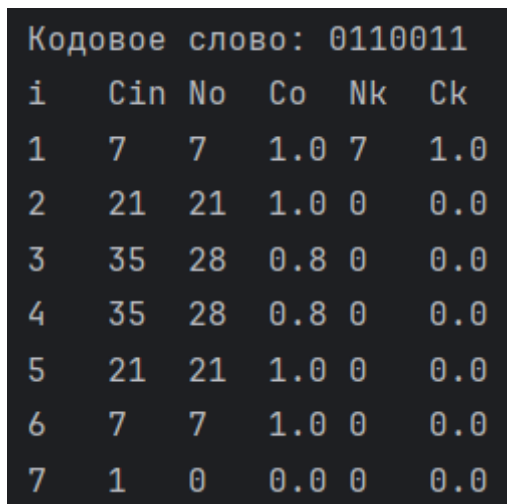
Nk += 1

Co = No / Cin if Cin > 0 else 0

Ck = Nk / Cin if Cin > 0 else 0

print(f"{i}\t{Cin}\t{No}\t{round(Co, 3)}\t{Nk}\t{round(Ck, 3)}")

3. Заполненная программно таблица результатов:



Кодовое слово: 0110011					
i	Cin	No	Co	Nk	Ck
1	7	7	1.0	7	1.0
2	21	21	1.0	0	0.0
3	35	28	0.8	0	0.0
4	35	28	0.8	0	0.0
5	21	21	1.0	0	0.0
6	7	7	1.0	0	0.0
7	1	0	0.0	0	0.0

4. Выводы

В ходе выполнения домашнего задания получены практические навыки по реализации алгоритмов кодирования и декодирования кодом Хэмминга [7,4], а также программного определения корректирующей способности кода.

Код Хэмминга [7,4] обладает полной корректирующей способностью для одиночных ошибок ($C_k = 1$ при $i=1$), что соответствует теории ($d=3$, $t=1$). Для ошибок кратности $i \geq 2$ корректирующая способность равна нулю — декодер либо не обнаруживает ошибку, либо исправление приводит к дополнительным ошибкам.

Это подтверждает, что код Хэмминга предназначен именно для исправления одиночных ошибок.

5. Список используемой литературы и URL-ссылок

- Галкин В.А., Григорьев Ю.А. Телекоммуникации и сети: Учеб. Пособие для вузов.- М.: Изд-во МГТУ им.Н.Э.Баумана, 2003
- http://www.opennet.ru/docs/RUS/inet_book/

6. Электронная копия отчёта и программы (включая исходные модули)

Электронная копия отчёта и исходный код программы доступны на удалённом репозитории github: <https://github.com/jayus-9/Network>