Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования   
**Национальный исследовательский университет «МЭИ»**

**Лабораторная работа №5**

по курсу«**Методы и средства защиты информации**»

**«Кодирование информации прямоугольным кодом»**

Выполнил:

студент группы А-07-18

Востриков Р.В.

Вариант 8

Москва  
2020

**1. Задание**

Задание 1

Закодировать сообщения методом Хаффмана

сообщение 1 2 3 4 5 6 7

вероятность 0,3 0,2 0,2 0,1 0,1 0,05 0,05

Задание 2  
Составить программу для кодирования и декодирования двоичного сообщения длины N (N четное) (или массив из N двоичных цифр 0 или 1) c помощью прямоугольного кода.  
Проверить, что при искажении одного разряда с помощью данного кода можно исправлять одинарные или обнаруживать двойные ошибки.

**2. Описание результатов**

Задание 1

0

1234567

23

14567

4567

1

4

567

0

0

1

1

1

5

67

0

1

6

7

0

1

2

3

0

1

Пусть алфавит будет X = { 1, 2, 3, 4, 5, 6, 7 }, а набор весов W = {0.3, 0.2, 0.2, 0.1, 0.1, 0.05, 0.05}.

В дереве Хаффмана будет 7 узлов:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Узел | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Вес | 0.30 | 0.20 | 0.20 | 0.10 | 0.10 | 0.05 | 0.05 |

Объединим в один узел два минимальных по весу узла 6 и 7:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Узел | 1 | 2 | 3 | 4 | 5 | 67 |
| Вес | 0.30 | 0.20 | 0.10 | 0.10 | 0.10 | 0.10 |

Затем опять объединим в один узел два минимальных по весу узла 5 и 67:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Узел | 1 | 2 | 3 | 567 | 4 |
| Вес | 0.30 | 0.20 | 0.20 | 0.20 | 0.10 |

Ещё раз повторим эту же операцию для узлов 567 и 4:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Узел | 1 | 4567 | 2 | 3 |
| Вес | 0.30 | 0.30 | 0.20 | 0.20 |

Ещё раз повторим эту же операцию для узлов 2 и 3:

|  |  |  |  |
| --- | --- | --- | --- |
| Узел | 23 | 1 | 4567 |
| Вес | 0.40 | 0.40 | 0.20 |

Ещё раз повторим эту же операцию для узлов 1 и 4567:

|  |  |  |
| --- | --- | --- |
| Узел | 14567 | 23 |
| Вес | 0.60 | 0.40 |

На последнем шаге объединм два узла 14567 и 23:

|  |  |
| --- | --- |
| Узел | 1234567 |
| Вес | 1 |

Остался один узел, значит, мы пришли к корню дерева Хаффмана (смотри рисунок). Теперь для каждого символа выберем кодовое слово (бинарная последовательность, обозначающая путь по дереву к этому символу от корня):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Узел | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Код | 10 | 00 | 01 | 110 | 1110 | 11110 | 11111 |

**3. Код программы (к заданию 2)**

import numpy as np

# функция кодирования сообщений прямоугольным кодом

# msg - массив двоичных цифр

def encode(msg):

length = len(msg)

# кодирование только сообщений четной длины

if length % 2 == 0:

rows = 2

cols = length // 2

# создаем таблицу (rows x cols) из исходного сообщения

table = np.array(msg, dtye='int8').reshape([rows, cols])

# дополняем таблицу столбцом и строкой чтобы получить прямоугольный код

alpha = table.sum(axis=1) % 2

table = np.append(table, alpha.reshape(len(s),1), axis = 1)

beta = code\_table.sum(axis=0) % 2

table = np.append(code\_table, beta, axis = 1)

# из полученной таблицы составляем сообщение

text = np.array2string(table[:-1,:-1].flatten(), separator='')

a = np.array2string(table[:-1, -1], separator='')

b = np.array2string(table[-1, :], separator='')

return '{}{}{}'.format(text[1,-1], a[1,-1], b[1,-1])

else:

raise Exception("Длина сообщение должна быть кратна 2!")

ё# функция декодирования сообщений из прямоугольного кода

# code - массив двоичных цифр

# возвращает кортеж из сообщения, флага наличия ошибок, флага исправления ошибки

def decode(code):

length = len(code)

# код, полученный с помощью функции encode(), будет кратен 3

if length % 3 == 0:

rows = 3

cols = length // 3

# исходное сообщение имеет длину 2 \* (cols-1)

msg\_len = 2 \* (cols - 1)

msg = np.array(code[:msg\_len], dtype='int8')

# слудующие за сообщением 2 бита служат для проверки четности строк

row\_check = code[msg\_len:msg\_len+2]

# оставшиеся cols бит служат для проверки четности столбцов

col\_check = code[-cols:]

# проверка на наличие ошибок

# вычисляем соллбец и строку для проверки на четность полученного сообщения

table = msg.reshape([2, msg.size // 2])

row\_actual = table.sum(axis=1) % 2 # биты четности строк полученного сообщения

col\_actual = table.sum(axis=0) % 2 # без учета столбца проверки четности строк сообщения

col\_actual = np.append(col\_actual, row\_actual.sum() % 2) # добавляем проверку последнего столбца

# проверка соответствия вычисленных контрольных битов полученным

row\_errors = np.where(row\_actual != row\_check)[0]

col\_errors = np.where(col\_actual != col\_check)[0]

# обработка ошибок (в предположении что ошибка не могла возникнуть ни в одном из контрольных битов)

if row\_errors.size == 0 and col\_errors.size == 0:

return (stringify(msg), False, False) # сообщение получено без ошибок

elif row\_errors.size == 1 and col\_errors.size == 2: # обнаружена одна ошибка

pos = row\_errors[0]\*(cols-1) + col\_errors[0] # позиция ошибки в сообщении

msg[pos] = 1 if msg[pos] == 0 else 0

return (stringify(msg), True, True)

else:

return (stringify(msg), True, False) # невозможно исправить более одной ошибки

else:

raise Exception("Некорректная длина сообщения")

**4. Вывод**

В лабораторной работе было показано кодирование сообщений с помощью алгоритма Хаффмана. Полученный код оказался оптимальнее равномерного кода. Также была написана программа, кодирующая и декодирующая сообщения с помощью прямоугольного кода. Такой способ позволяет исправить единичную ошибку и обнаружить двойную.