МИНИСТЕРСТВО НАУКИ ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационный технологий и программирования

**Лабораторная работа №02**

**По дисциплине «Введение в цифровую культуру и программирование»**

**Эффективное кодирование**

**Вариант №1**

Выполнил студент группы №M3109  
Бабурин Тимур Сергеевич

Проверил  
Хлопотов Максим Валерьевич

САНКТ-ПЕТЕРБУРГ

2019

Исходное изображение и фотография в заданном формате.

Изображение выглядит как кот, животное, сидит, млекопитающее

Автоматически созданное описание

Исходное изображение.

Изображение выглядит как кот, сидит, животное, смотрит

Автоматически созданное описание

Фотография в заданном формате.

Изображение было обработано с помощью программы: "paint.net".

1.Обрезание происходило с помощью инструмента: "Выбор прямоугольной области".

2.Изменение цвета происходило с помощью функций: "Коррекция"»"Сделать чёрно-белым".

3.Глубина цвета была изменена при сохранений изображения(8 бит):

Изображение выглядит как снимок экрана

Автоматически созданное описаниеИзображение выглядит как снимок экрана

Автоматически созданное описаниеИзображение выглядит как снимок экрана

Автоматически созданное описание

3.

2.

1.

Полученная цифровая последовательность.

Программа.

#include <iostream>  
#include <cmath>  
#include "bitmap\_image.hpp" //Библиотека растровых изображений C++  
// для растрового изображения с 24-битным разрешением на пиксель.  
  
using namespace std;  
  
int main()  
{  
 bitmap\_image image("image\_ch\_b.bmp"); //Объект, где хранится изображение.  
 int picture[128][128]; //Массив для данных изображения.  
 for (int i = 0; i < image.width(); i++)  
 {  
 for (int j = 0; j < image.height(); j++)  
 {  
 rgb\_t colour; //Структура, где хранится значение характеристики RGB.  
 image.get\_pixel(i, j, colour); //Получаем данные.  
 picture[i][j] = colour.red; //Значения RGB в нашем формате(ч/б) равны, поэтому записать  
 //можем любое.  
 }  
 }  
  
 cout << "Sequence:" << endl;  
  
 int freq[261]; //Массив для частоты встречаемости, позже пригодится для вычисления энтропий.  
 for (int i = 0; i < 128; i++)  
 {  
 picture[i][64] = round(double(picture[i][64]) / 20) \* 20; //Квантование.  
 freq[picture[i][64]]++;  
 cout << picture[i][64] << ' '; //Выводим на консоль среднюю строку.  
 }  
  
 cout << endl << endl << "Count and frequency:" << endl;  
  
 float entropy = 0;  
 for (int i = 0; i < 261; i++)  
 {  
 if (freq[i] != 0)  
 {  
 cout << i << ' ' << freq[i] << endl; //Выводим на консоль частоту встречаемости.  
 entropy -= (double(freq[i]) / 128) \* log2(double(freq[i]) / 128); //Считаем энтропию  
 //с помощью формулы.  
 }  
 }  
  
 cout << endl;  
  
 cout << "Entropy = " << entropy; //Выводим на консоль энтропию.  
 return 0;  
}

Последовательность.

180 200 200 200 220 220 200 200 180 180 180 200 220 220 220 220 220 200 180 200 240 240 260 240 240 240 240 240 220 220 220 220 220 200 180 160 160 140 140 160 160 140 140 120 80 60 80 100 100 80 60 80 60 40 60 60 60 60 40 40 40 60 60 60 40 40 40 40 80 100 100 100 100 100 120 100 100 100 120 120 120 120 140 140 140 140 120 120 140 140 120 100 100 80 80 80 80 60 60 100 120 120 140 140 160 180 200 200 200 200 180 180 180 200 200 200 220 220 240 240 260 260 260 260 260 240 240 240

Вероятность появления.количество символов алфавита.энтропия.расчётная длина двоичного кода.

|  |  |
| --- | --- |
| Символ | Вероятность |
| 160 |  |
| 260 |  |
| 40 |  |
| 80 |  |
| 180 |  |
| 120 |  |
| 60 |  |
| 140 |  |
| 240 |  |
| 100 |  |
| 220 |  |
| 200 |  |

Количество символов: 12.

Энтропия по формуле: 3.5204.

Расчётная длина двоичного кода: 4.

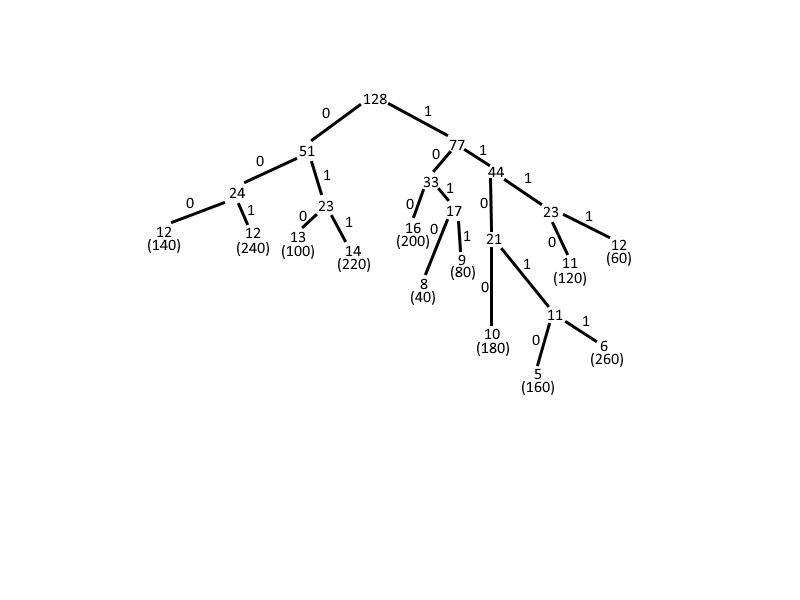
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Символ | 160 | 260 | 40 | 80 | 180 | 120 | 60 | 140 | 240 | 100 | 220 | 200 |
| Код | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 |

коды шеннона-фано.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Символ | Вероятность | 1 | 2 | 3 | 4 |
| 200 |  | 0 | 0 | 0 | |
| 220 |  | 1 | |
| 100 |  | 1 | 0 | |
| 140 |  | 1 | 0 |
| 240 |  | 1 |
| 60 |  | 1 | 0 | 0 | |
| 120 |  | 1 | 0 |
| 180 |  | 1 |
| 80 |  | 1 | 0 | 0 |
| 40 |  | 1 |
| 260 |  | 1 | 0 |
| 160 |  | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Символ | 200 | 220 | 100 | 140 | 240 | 60 | 120 | 180 | 80 | 40 | 260 | 160 |
| Код | 000 | 001 | 010 | 0110 | 0111 | 100 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |

кодовое дерево хаффмана.



коды хаффмана для символов.

|  |  |
| --- | --- |
| Символ | Код |
| 200 | 100 |
| 220 | 011 |
| 100 | 010 |
| 60 | 1111 |
| 240 | 001 |
| 140 | 000 |
| 120 | 1110 |
| 180 | 1100 |
| 80 | 1011 |
| 40 | 1010 |
| 260 | 11011 |
| 160 | 11010 |

Закодированное равномерным кодом сообщение, длина кодового слова, количество переданной информации при передаче сообщения.

Последовательность, которая была закодирована:

180 200 200 200 220 220 200 200 180 180 180 200 220 220 220 220 220 200 180 200 240 240 260 240 240 240 240 240 220 220 220 220 220 200 180 160 160 140 140 160 160 140 140 120 80 60 80 100 100 80 60 80 60 40 60 60 60 60 40 40 40 60 60 60 40 40 40 40 80 100 100 100 100 100 120 100 100 100 120 120 120 120 140 140 140 140 120 120 140 140 120 100 100 80 80 80 80 60 60 100 120 120 140 140 160 180 200 200 200 200 180 180 180 200 200 200 220 220 240 240 260 260 260 260 260 240 240 240

Закодированное сообщение:

01001011101110111010101010111011010001000100101110101010101010101010101101001011100010000001100010001000100010001010101010101010101010110100000000000111011100000000011101110101001101100011100110010011011000110110001001100110011001100010001000100110011001100010001000100010001110011001100110011001010110011001100101010101010101010111011101110111010101010111011101011001100100110011001100110110011010010101010101110111000001001011101110111011010001000100101110111011101010101000100000010001000100010001100010001000

Длина кодового слова: 4.

Количество информаций при передаче сообщений: 512 бит.

Закодированная полученным кодом Шеннона–Фано последовательность, длина кодового слова, количество переданной информации при передаче сообщения.

Последовательность, которая была закодирована:

180 200 200 200 220 220 200 200 180 180 180 200 220 220 220 220 220 200 180 200 240 240 260 240 240 240 240 240 220 220 220 220 220 200 180 160 160 140 140 160 160 140 140 120 80 60 80 100 100 80 60 80 60 40 60 60 60 60 40 40 40 60 60 60 40 40 40 40 80 100 100 100 100 100 120 100 100 100 120 120 120 120 140 140 140 140 120 120 140 140 120 100 100 80 80 80 80 60 60 100 120 120 140 140 160 180 200 200 200 200 180 180 180 200 200 200 220 220 240 240 260 260 260 260 260 240 240 240

Закодированное сообщение:

1011000000000001001000000101110111011000001001001001001000101100001110111111001110111011101110111001001001001001000101111111111011001101111111101100110101011001001100010010110010011001001101100100100100110111011101100100100110111011101110111000100100100100101010010010010101010101010101001100110011001101010101001100110101001001011001100110011001001000101010101001100110111110110000000000001011101110110000000000010010111011111101110111011101110011101110111

Количество информаций при передаче сообщений: 457 бит.

Закодированная полученным кодом Хаффмана последовательность, длина кодового слова, количество переданной информации при передаче сообщения.

Последовательность, которая была закодирована:

180 200 200 200 220 220 200 200 180 180 180 200 220 220 220 220 220 200 180 200 240 240 260 240 240 240 240 240 220 220 220 220 220 200 180 160 160 140 140 160 160 140 140 120 80 60 80 100 100 80 60 80 60 40 60 60 60 60 40 40 40 60 60 60 40 40 40 40 80 100 100 100 100 100 120 100 100 100 120 120 120 120 140 140 140 140 120 120 140 140 120 100 100 80 80 80 80 60 60 100 120 120 140 140 160 180 200 200 200 200 180 180 180 200 200 200 220 220 240 240 260 260 260 260 260 240 240 240

Закодированное сообщение:

110010010010001101110010011001100110010001101101101101110011001000010011101100100100100100101101101101101110011001101011010000000110101101000000011101011111110110100101011111110111111101011111111111111111010101010101111111111111010101010101010101101001001001001011100100100101110111011101110000000000000111011100000001110010010101110111011101111111111010111011100000001101011001001001001001100110011001001001000110110010011101111011110111101111011001001001

Количество информаций при передаче сообщений: 456 бит.

Расчеты по п. 8 – 10 задания на лабораторную работу.

Средняя длина кодовой комбинаций:

Шеннон-Фано:

Хаффман:

Степень сжатия сообщений:

Шеннон-Фано:

Хаффман:

Избыточность для сформированных кодов:

Шеннон-Фано:

Хаффман:

вывод.

Методика Шеннона-Фано не всегда приводит к однозначному построению кода. Так как при разбиений на подгруппы можно сделать большей по вероятности как верхнюю, так и нижнюю подгруппу. В результате среднее число символов на букву окажется другим. При этом метод Хаффмана гарантирует однозначное построение кода с наименьшим для данного распределения вероятностей средним числом символов на букву. Метод Хаффмана производит такое сжатие, что сжимает данные до их энтропий, если вероятности символов точно равны отрицательным степеням двойки.По моему мнению метод Хаффмана вссегда будет лучше результатов по методу Шеннона-Фано.