

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и
автоматизированных систем

Лабораторная работа №4
по дисциплине: «Теория информации»

Выполнил: ст. группы ПВ-211

Чувилко Илья Романович

Проверил:

Твердохлеб В.В.

Белгород 2023 г.

Тема: «Исследование кода Гилберта-Мура »

Содержание отчета

1. Построить обработчик, выполняющий компрессию по алгоритму Гилберта-Мура
- 2 Создать генераторы данных, работающих как источники Хартли и Бернулли (в двоичном алфавите)
- 3 Построить коды Гилберта-Мура для каждой из последовательностей, для чего предварительно сегментировать каждую цепочку по 8 символов
4. Вычислить полученные коэффициенты сжатия и величину дисперсии для каждой последовательности

Ход работы:

- 1 Построить обработчик, выполняющий компрессию по алгоритму Гилберта-Мура.

```
std::vector<bool> GetFloatNBits(  
    float number,  
    int n  
) {  
    std::vector<bool> res(n);  
    for (int i = 0; i < n; i++) {  
        number *= 2;  
        res[i] = int(number);  
        number -= int(number);  
    }  
  
    return res;  
}  
  
std::vector<FanoCode> GetMooreHilbertCode(const std::unordered_map<char, int> &counters) {  
    if (counters.empty()) {  
        return {};  
    }  
  
    if (counters.size() == 1) {  
        return { { counters.begin()->first, 1, std::vector{true} } };  
    }  
  
    std::vector<FanoCode> codes;  
    std::vector<float> d;  
    std::vector<float> sigma;  
    size_t total = 0;  
  
    for (auto &i: counters) {  
        codes.push_back({i.first, i.second});  
        total += i.second;  
    }  
  
    for (int i = 0; i < codes.size(); i++) {  
        double p = static_cast<float>(codes[i].count) / total;  
  
        if (i == 0) {  
            d.push_back(0);  
        } else {  
            double prev_p = static_cast<float>(codes[i - 1].count) / total;
```

```

    d.push_back(d[i - 1] + prev_p);
}

sigma.push_back(d[i] + p / 2);
int length = ceil(-std::log2(p)) + 1;
codes[i].code = GetFloatNBits(sigma[i], length);
}

return codes;
}

```

2 Создать генераторы данных, работающих как источники Хартли и Бернулли (в двоичном алфавите).

```

std::string HartliGenerator(
    int n,
    int n_bits
) {
    std::string t;
    std::random_device rd;
    std::mt19937 gen(rd());
    std::uniform_int_distribution<unsigned char> d(0, std::min((1 << n_bits) - 1, 127));
    for (int i = 0; i < n; i++) {
        t.push_back(d(gen));
    }

    return t;
}

std::string BernoulliGenerator(
    int n,
    float p
) {
    std::string t;
    std::random_device rd;
    std::mt19937 gen(rd());
    std::bernoulli_distribution d(p);
    for (int i = 0; i < n; i++) {
        t.push_back(d(gen));
    }

    return t;
}

```

3. Построить коды Гилберта-Мура для каждой из последовательностей, для чего предварительно сегментировать каждую цепочку по 8 символов.

Сообщение сгенерированное источником Хартли:

~&j@~s~)~7#~==aBMKR,~|~H*\zymC~m 5]V!~ ^Fm NSf~e

<f>		1		0000001	
<S>		1		0000011	
<N>		1		0000110	
<F>		1		0001000	
<	>		1		0001011
<V>		1		0001110	
<]>		1		0010000	
<5>		1		0010011	
< >		2		001011	
<m>		3		001110	
<~>		2		010001	
<~>		1		0100111	
<=>		1		0101010	
<^>		1		0101100	
<#>		1		0101111	
<@>		1		0110001	
<Z>		1		0110100	
<s>		1		0110111	
<)>		1		0111001	
<~>		1		0111100	
<j>		1		0111110	
<M>		1		1000001	
<a>		1		1000011	
<&>		1		1000110	
<~>		1		1001000	
		1		1001011	
<C>		1		1001110	
<~>		2		101000	
>		1		1010101	
< >		1		1011000	
<K>		1		1011010	
<~>		1		1011101	
<R>		1		1011111	
<7>		1		1100010	

4. Вычислить полученные коэффициенты сжатия и величину дисперсии для каждой последовательности.

Алгоритм Гильберта-Мура для источника Хартли:
Длина исходного сообщения в битах: 400
Длина закодированного сообщения в битах: 339
Коэффициент сжатия: 1.1799411
Средняя длина кодового слова: 6.7800002
Дисперсия: 0.4099

Алгоритм Гильберта-Мура для источника Бернулли:
Длина исходного сообщения в битах: 400
Длина закодированного сообщения в битах: 118
Коэффициент сжатия: 3.3898306
Средняя длина кодового слова: 2.3599999
Дисперсия: 0.4656

Алгоритм Гильберта-Мура для сообщения:
Длина исходного сообщения в битах: 720
Длина закодированного сообщения в битах: 482
Коэффициент сжатия: 1.493776
Средняя длина кодового слова: 5.3555555
Дисперсия: 1.081