Лабораторная работа № 2

по ТИПиС

студента группы ИТ-32

Манукова Давида Альбертовича

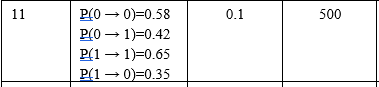
Выполнение: \_\_\_\_\_\_\_\_\_\_ Защита: \_\_\_\_\_\_\_\_\_\_

**Моделирование передачи сообщения по каналу с помехами**

Задание к работе

Задание А.

Исходя из распределения вероятностей символов первичного алфавита (для языка эсперанто), полученных в задании работы №1, и заданных характеристик бинарного канала связи, рассчитать априорные характеристики передачи информации по каналу с помехами.



А.1. Исходя из кодировки, полученной в п.А.11 работы №1, и заданной модели помех, построить канальную матрицу передачи сообщений на языке, определенном в задании лабораторной работы №1.

А.2. Рассчитать априорные характеристики:

А.2.1. Энтропию источника информации. Сравнить с результатом, полученным в п. А.8 работы №1.

А.2.2. Энтропию приемника информации.

А.2.3. Энтропию шума.

А.2.4. Утечку (ненадежность) информации.

А.2.5. Количество полезной информации.

А.2.6. Скорость передачи информации.

А.3. В программе «Бинарный канал с помехами» смоделировать прохождение сообщений по каналу связи с помехами (без помехоустойчивого кодирования), построить экспериментальную канальную матрицу и рассчитать апостериорные характеристики:

А.3.1. Энтропию источника информации.

А.3.2. Энтропию приемника информации.

А.3.3. Энтропию шума.

А.3.4. Утечку (ненадежность) информации.

А.3.5. Количество полезной информации.

А.3.6. Скорость передачи информации.

А.4. Сравнить и пояснить результаты, полученные в п.п. А.2 и А.3.

Задание Б.

Вместо выполнения п. А.3 написать программу, моделирующую прохождение сообщений по каналу с помехами.

Б.1. Смоделировать генерацию К сообщений источника информации в соответствии с заданным распределением вероятности символов первичного алфавита. Результатом должна стать апостериорные вероятности (частоты) входных сообщений.

1. Разбить на 10 интервалов.

2. Подсчитать количество переходов из одного символа в другой . то есть 400 условно бросков.

Б.2. Смоделировать искажения при прохождении входных сообщений по каналу с помехами в соответствии с заданными вероятностными характеристиками. Результатом должна стать апостериорная канальная матрица и апостериорные вероятности (частоты) выходных сообщений.

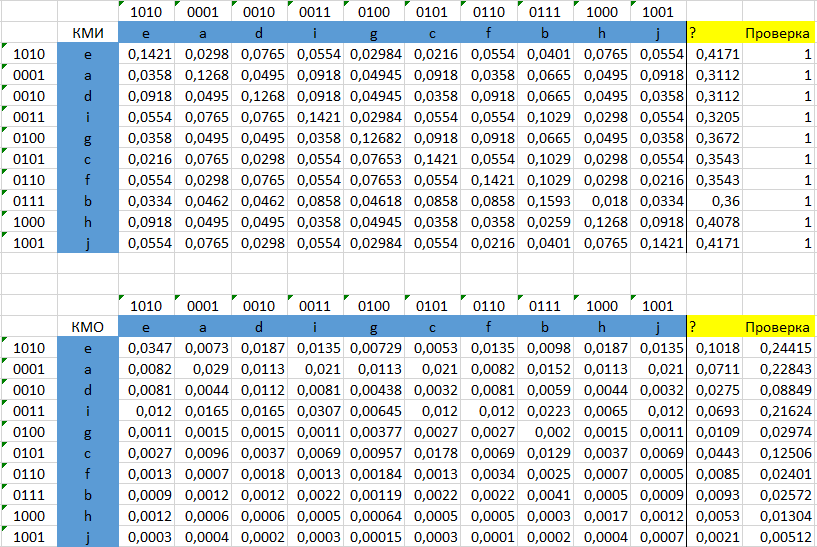
Б.3. На основании моделирования вычислить экспериментальные характеристики: количество переданной информации (энтропию источника), количество принятой информации (энтропию приемника), потери информации (утечку), излишнюю информацию (шумы), количество полезной информации и скорость передачи информации.

Б.4. Сравнить экспериментальные результаты с теоретическими и пояснить причину отличий.

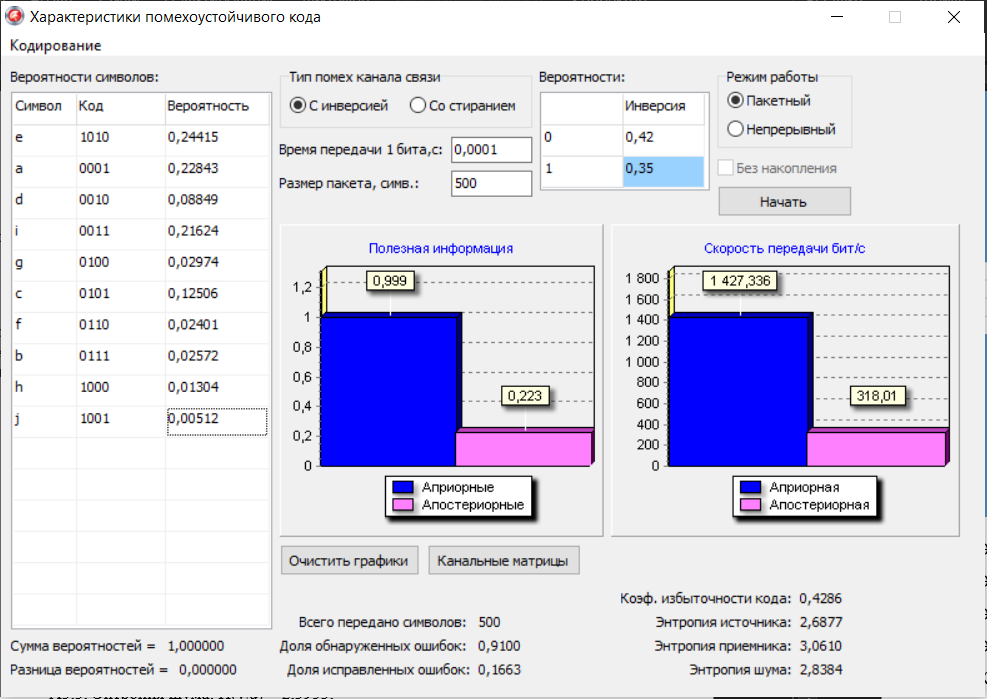
Ход работы

Задание А

1. Построил КМИ и КМО



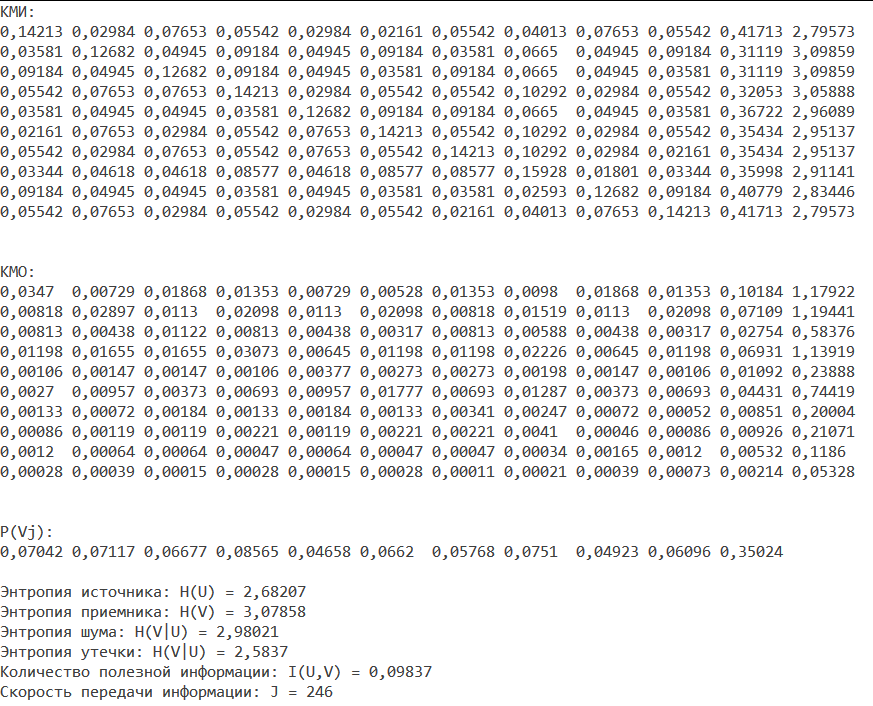
1. Энтропия источника: *H(U) = 2.6821*
2. Энтропия приемника: *H(V) = 3.0786*
3. Энтропия шума: *H(V|U) = 2.9802*
4. Энтропия утечки: *H(U|V) = 2.5837*
5. Количество полезной информации: *I(U,V) = 0.0984*
6. Скорость передачи информации: *J = 246*
7. С помощью программы характеристики помехоустойчивости кода рассчитал апостериорные характеристики и смоделировал прохождение сообщений по каналу связи с помехами.



1. Энтропия источника: *H(U) = 2.6877*
2. Энтропия приемника: *H(V) = 3.0610*
3. Энтропия шума: *H(V|U) = 2.8384*
4. Энтропия утечки: *H(U|V) = 2.6877 - 3.0610 + 2.8384 = 2.4651*
5. Количество полезной информации: *I(U,V) = 0.223*
6. Скорость передачи информации: *J = 318.01*

Задание Б

1. Результат работы программы



1. Листинг программы

class Letters

{

public char c;

public int[] bincode = new int[4];

public double prob;

}

class Program

{

static void Main(string[] args)

{

List<Letters> letters = new List<Letters>();

double[] p = { 0.58, 0.42, 0.65, 0.35 };

double[,] kmi = new double[10, 12];

double[,] kmo = new double[10, 12];

double[] PVj = new double[11];

int height = 10;

int width = 12;

for (int i = 0; i < height; i++)

{

for (int j = 0; j < width-1; j++)

{

kmi[i, j] = 1;

}

}

char[] c = { 'e', 'a', 'd', 'i', 'g', 'c', 'f', 'b', 'h', 'j' };

int[,] bincodes = {

{1, 0, 1, 0 },

{0, 0, 0, 1 },

{0, 0, 1, 0 },

{0, 0, 1, 1 },

{0, 1, 0, 0 },

{0, 1, 0, 1 },

{0, 1, 1, 0 },

{0, 1, 1, 1 },

{1, 0, 0, 0 },

{1, 0, 0, 1 }

};

double[] probs = { 0.24415, 0.22843, 0.08849, 0.21624, 0.02974, 0.12506, 0.02401, 0.02572, 0.01304, 0.00512 };

for (int i = 0; i < 10; i++)

{

Letters letter = new Letters();

letter.c = c[i];

letter.prob = probs[i];

for (int j = 0; j < 4; j++)

{

letter.bincode[j] = bincodes[i, j];

}

letters.Add(letter);

}

double sum = 0;

for (int i = 0; i < height; i++)

{

for (int j = 0; j < width; j++)

{

if (j != 10)

{

if (j != 11)

{

for (int k = 0; k < 4; k++)

{

if ((letters[i].bincode[k] == 0) && (letters[j].bincode[k] == 0))

{

kmi[i, j] \*= p[0];

}

if ((letters[i].bincode[k] == 1) && (letters[j].bincode[k] == 1))

{

kmi[i, j] \*= p[2];

}

if ((letters[i].bincode[k] == 0) && (letters[j].bincode[k] == 1))

{

kmi[i, j] \*= p[1];

}

if ((letters[i].bincode[k] == 1) && (letters[j].bincode[k] == 0))

{

kmi[i, j] \*= p[3];

}

}

sum += kmi[i, j];

kmi[i, 11] += kmi[i, j] \* Math.Log2(kmi[i, j]);

}

}

else

{

kmi[i, j] = 1 - sum;

sum = 0;

kmi[i, 11] += kmi[i, j] \* Math.Log2(kmi[i, j]);

}

}

}

Console.WriteLine("КМИ:");

for (int i = 0; i < height; i++)

{

for (int j = 0; j < width; j++)

{

kmi[i, j] = Math.Round(kmi[i, j], 5);

kmi[i, j] = Math.Abs(kmi[i, j]);

Console.Write(kmi[i,j] + "\t");

}

Console.WriteLine();

}

Console.WriteLine("\n\nКМО:");

for (int i = 0; i < height; i++)

{

for (int j = 0; j < width; j++)

{

if (j != 11)

{

kmo[i, j] = kmi[i, j] \* letters[i].prob;

kmo[i, 11] += kmo[i, j] \* Math.Log2(kmo[i, j]);

}

kmo[i, j] = Math.Round(kmo[i, j], 5);

kmo[i, j] = Math.Abs(kmo[i, j]);

Console.Write(kmo[i, j] + "\t");

}

Console.WriteLine();

}

Console.WriteLine("\n\nP(Vj):");

for (int i = 0; i < width-1; i++)

{

for (int j = 0; j < height; j++)

{

PVj[i] += kmo[j, i];

PVj[i] = Math.Round(PVj[i], 5);

}

Console.Write(PVj[i] + "\t");

}

double entropySrc = 0;

for (int i = 0; i < probs.Length; i++)

{

entropySrc += probs[i] \* Math.Log2(probs[i]);

}

entropySrc = Math.Round(entropySrc, 5);

entropySrc = Math.Abs(entropySrc);

Console.WriteLine("\n\nЭнтропия источника: H(U) = " + entropySrc);

double entropyRcv = 0;

for (int i = 0; i < PVj.Length; i++)

{

entropyRcv += PVj[i] \* Math.Log2(PVj[i]);

}

entropyRcv = Math.Round(entropyRcv, 5);

entropyRcv = Math.Abs(entropyRcv);

Console.WriteLine("Энтропия приемника: H(V) = " + entropyRcv);

double entropyNoise = 0;

for (int i = 0; i < probs.Length; i++)

{

entropyNoise += probs[i] \* kmi[i,11];

}

entropyNoise = Math.Round(entropyNoise, 5);

entropyNoise = Math.Abs(entropyNoise);

Console.WriteLine("Энтропия шума: H(V|U) = " + entropyNoise);

double entropyLeak = entropySrc - entropyRcv + entropyNoise;

Console.WriteLine("Энтропия утечки: H(V|U) = " + entropyLeak);

double countofUsefulInfo = entropyRcv - entropyNoise;

countofUsefulInfo = Math.Round(countofUsefulInfo, 5);

Console.WriteLine("Количество полезной информации: I(U,V) = " + countofUsefulInfo);

double sendingSpeed = countofUsefulInfo / 0.0001 / 4;

sendingSpeed = Math.Round(sendingSpeed, 0);

Console.WriteLine("Скорость передачи информации: J = " + sendingSpeed);

Console.ReadLine();

}

}

Вывод: В ходе в ходе выполнения лабораторной работы были приобретены навыки моделирования передачи сообщения по каналу с помехами.