Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Лабораторная работа №7
по теме
"Помехоустойчивое кодирование"

Выполнил студент группы 33501/3
_____ Кисличенко Б. Д

Руководитель
_____ Богач Н. В

Санкт-Петербург
2018

1 Цель

Изучение методов помехоустойчивого кодирования и сравнение их свойств.

2 Постановка задачи

1) Провести кодирование/декодирование сигнала, полученного с помощью функции randerr кодом Хэмминга 2-мя способами; с помощью встроенных функций encode/decode, а также через создание проверочной и генераторной матриц и вычисление синдрома. Оценить корректирующую способность кода.

2) Выполнить кодирование/декодирование циклическим кодом, кодом БЧХ, кодом Рида-Соломона. Оценить корректирующую способность кода.

3 Коды кодирования

3.1 Циклические коды

Циклические коды - подкласс линейных кодов, обладающие следующим свойством: циклическая подстановка символов в кодированном блоке дает другое возможное кодовое слово того же кода. Для работы с циклическими кодами в пакете Communications есть две функции. С помощью функции cyclpoly можно получить порождающий полином циклического кода. Для этого предварительно нужно задать число символов в кодируемом и закодируемом блоках. С помощью функции cyclegen и полученного раннее полинома можно получить порождающую и проверочную матрицы для данного кода.

3.2 Коды БЧХ

Коды БЧХ (Боуза — Чоудхури — Хоквингема) - являются подклассом циклических юлочных кодов. Для работы с ними есть функции bchenco (кодирование) и bcddeco (декодирование). Функция bchpoly позволяет расчитывать и считывать параметры или порождающий полином для двоичных кодов БЧХ.

3.3 Коды Хэмминга

Коды Хэмминга - подкласс циклических блочных кодов. Порождающий полином для кода Хэмминга - примитивен. Длина кодированного блока равна 2m-1. Порождающая и проверочная матрицы для кодов Хэмминга генерируются функцией hammgen.

3.4 Коды Рида-Соломона

Коды Рида-Соломона - подкласс циклических блочных кодов. Это единственные поддерживаемые пакетом Communications недвоичные коды. Для работы с этим кодом есть функции rsenco (кодирование) и rsdeco (декодирование). Функции rsencof и rsdencof осуществляют кодирование и декодирование текстового файла. Функция rspoly генерирует порождающие полиномы для кодов Рида-Соломона.

4 Ход работы

4.1 Кодирование/декодирование кодом Хэмминга

```
%msg - передаваемое сообщение (11 бит)
msg=[1 1 0 0 1 0 1 0 1 1 1];
%Кодируем msg с помощью кода Хэмминга
%encode(msg,n,k), где n - длина кодового слова, k - длина
%блока сообщения
%Для кода Хэмминга n=2^m-1, k=n-m
%Для примера возьмем m=4=>n=15,k=11
m=4; n=15; k=11;
code_hamming=encode (msg,n,k,'hamming/binary')
%Декодируем, воссоздавая исходное сообщение
decoded_hamming = decode(code_hamming,n,k,'hamming/binary')
%сделаем ошибку при помощи инвертирования одного бита
code_hamming(7) = not(code_hamming(7));
%декодируем сообщение с ошибкой
[decoded_hamming,err] = decode(code_hamming,n,k,'hamming/binary')

%Проведем кодирование/декодирование с помощью создания проверочной
%и генераторной матрицы, а также вычислим синдром
msg=[1 1 0 0 1 0 1 0 1 1 1];
%[h,g,n,k] = hammgen(...) возвращает проверочную, порождающую
%матрицы и длину кодового слова n и длину блока исходного
%сообщения k
[h,g,n,k] = hammgen(m);
%столбцы порождающей матрицы с номерами не степенями 2 образуют
% единичную подматрицу, а остальные столбцы соответствуют
% проверочным уравнениям кода. Такая матрица при кодировании
% будет копировать биты сообщения в позиции, не степени 2,
% и заполнять другие позиции кода согласно системе вычисления
% контрольных разрядов.
new_msg=msg*g;
%приведем к бинарному виду путем поэлементного деления массива
```

Рис. 1: Код Matlab (код Хэмминга) часть 1

```
%сообщения на 2 и получения таким образом остатка от деления
new_msg=rem(new_msg,ones(1,n).*2)

%сделаем ошибку при помощи инвертирования одного бита
new_msg(7) = not(new_msg(7));
%вычисоим синдром сообщения с ошибкой
syndrom=new_msg*h';
syndrom=rem(syndrom,ones(1,n-k).*2);

%с помощью матрицы синдрома выявим ошибочный бит и исправим его
% Сначала вычислим таблицу декодирования для кода Хэмминга
%по проверочной таблице
num_error=syndtable(h);

%Преобразуем вектор-строку двоичных цифр матрицы синдрома
%в неотрицательное целое число.
%left-msb-первый столбец - старший разряд
tmp=bi2de(syndrom,'left-msb');
z=num_error(tmp+1,:);
new_msg;
rez=xor(new_msg,z);
decode_msg=decode(code_hamming,n,k,'hamming/binary');
```

Рис. 2: Код Matlab (код Хэмминга) часть 2

```
msg =

     1     1     0     0     1     0     1     0     1     1     1

code_hamming =

     0     0     1     1     1     1     0     0     1     0     1     0     1     1     1

code_hamming =

     0     0     1     1     1     1     1     0     1     0     1     0     1     1     1
```

Рис. 3: Сообщение, закодированное сообщение и закодированное сообщение с ошибкой в 7м разряде

```
error_col =

     0     0     0     0     0     0     1     0     0     0     0     0     0     0     0

rez =

  1×15 logical array

   0   0   1   1   1   1   0   0   1   0   1   0   1   1   1
```

Рис. 4: Номер разряда, в котором ошибка и исправленный результат

4.2 Кодирование/декодирование циклическим кодом

```
%Выполним кодирование/декодирование циклическим кодом
msg=[1 1 0 0 1 0 1 0 1 1 1];
pol=cyclpoly(n,k);
[h,g]=cyclgen(n,pol);
code=msg*g;
code=rem(code,ones(1,n).*2);
code(7)=not(code(7))
syndrom=code*h';
syndrom=rem(syndrom, ones(1,n-k).*2);

%с помощью матрицы синдрома выявим ошибочный бит и исправим его
% Сначала вычислим таблицу декодирования для кода Хэмминга
%по проверочной таблице
num_error=syndtable(h);

%Преобразуем вектор-строку двоичных цифр матрицы синдрома
%в неотрицательное целое число.
%left-msb-первый столбец - старший разряд
tmp=bi2de(syndrom,'left-msb');
z=num_error(tmp+1,:)
code;
rez=xor(code,z);
msg;
msg = decode(code,n,k,'cyclic/binary');
```

Рис. 5: Код Matlab (Циклический код)

```
msg =

     1     1     0     0     1     0     1     0     1     1     1


pol =

     1     0     0     1     1


code =

     1     1     1     1     1     1     1     0     1     0     1     0     1     1     1


error_col =

     0     0     0     0     0     0     1     0     0     0     0     0     0     0     0


rez =

  1×15 logical array

   1   1   1   1   1   1   0   0   1   0   1   0   1   1   1
```

Рис. 6: Сообщение, полнином, сообщение в циклическом коде, номер разряда с ошибкой, исправленное закодированное сообщение

4.3 Кодирование/декодирование кодом БЧХ

```
%Произведем кодирование/декодирование при помощи кодов БЧХ
msg=[1 1 0 0 1 0 1 0 1 1 1];

codebch=comm.BCHEncoder(n,k);
decbch=comm.BCHDecoder(n,k);
temp=msg';
code=step(codebch,temp(:))';

code(7)=not(code(7));
decode=step(decbch,code')';
```

Рис. 7: Код Matlab (код БЧХ)

```
code =

     1     1     0     0     1     0     0     0     1     1     1     1     1     1     1

decode =

     1     1     0     0     1     0     1     0     1     1     1

msg =

     1     1     0     0     1     0     1     0     1     1     1
```

Рис. 8: сообщение в коде БЧХ, с ошибкой, декодированное сообщение

4.4 Кодирование/декодирование кодом Рида-Соломона

```
%Произведем кодирование/декодирование с помощью кодов Рида-Соломона
m=3;%число бит на символ
n=2^m-1;%длина кода слова
k=3;%длина сообщения

msg=gf([2 7 3;4 0 6],m)
%сгенерируем код Рида-Соломона
code=rsenc(msg,n,k)

%добавим ошибки к закодируемому сообщению
errs=gf([0 5 5 0 0 0 0; 0 7 7 7 0 0 0],m);
code=code+errs;

%Раскодируем сообщение с ошибками
[dec,errnum]=rsdec(code,n,k)
```

Рис. 9: Код Matlab (код Рида-Соломона)

```
msg = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

   2   7   3
   4   0   6


code = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

   2   7   3   3   6   7   6
   4   0   6   4   2   2   0


code = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

   2   2   6   3   6   7   6
   4   7   1   3   2   2   0


dec = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

   2   7   3
   4   7   1


errnum =

     2
    -1
```

Рис. 10: Передаваемое сообщение, сообщение в коде Рида-Соломона, с ошибкой, декодирование, кол-во ошибок

5 Вывод

в ходе данной лабораторной работы были изучены методы помехоустойчивого кодирования и были сравнены их свойства. Исправляющая способность кодов Хэмминга, БЧХ, циклического кода равна 1. А исправляющая способность Рида-Соломона равна 2. Код Хэмминга используется в некоторых прикладных программах в области хранения данных. Коды Рида — Соломона имеют очень широкую область применения благодаря их способности находить и исправлять многократные пакеты ошибок.