САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ПЕТРА ВЕЛИКОГО

Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчет
по лабораторной работе №7
на тему
"Помехоустойчивое кодирование"

выполнил:
Кыльчик И.В.
группа: 33501/1
преподаватель:
Богач Н.В.

Санкт-Петербург
2018

1. Цель работы

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. Теоретическая часть

Физическая среда, по которой передаются данные, не может быть абсолютно надёжной. Более того, уровень шума бывает очень высоким, например в беспроводных системах связи и телефонных системах. Ошибки при передаче — это реальность, которую надо обязательно учитывать.

В разных средах характер помех разный. Ошибки могут быть одиночные, а могут возникать группами, сразу по несколько. В результате помех могут исчезать биты или наоборот — появляться лишние.

Для надёжной передачи кодов было предложено два основных метода:

Первый — добавить в передаваемый блок данных нескольких «лишних» бит так, чтобы, анализируя полученный блок, можно было бы сказать, есть в переданном блоке ошибки или нет. Это так называемые коды с обнаружением ошибок.

Второй — внести избыточность настолько, чтобы, анализируя полученные данные, можно не только замечать ошибки, но и указать, где именно возникли искажения. Это коды, исправляющие ошибки.

Коды Хемминга — простейшие линейные коды с минимальным расстоянием 3, то есть способные исправить одну ошибку. Код Хемминга может быть представлен в таком виде, что синдром $\vec{s} = \vec{r}H^T$, где $\vec{r}$ — принятый вектор, будет равен номеру позиции, в которой произошла ошибка. Это свойство позволяет сделать декодирование очень простым.

Любой код (в том числе нелинейный) можно декодировать с помощью обычной таблицы, где каждому значению принятого слова $\vec{r_i}$ соответствует наиболее вероятное переданное слово $\vec{u_i}$. Однако данный метод требует применения огромных таблиц уже для кодовых слов сравнительно небольшой длины.

Для линейных кодов этот метод можно существенно упростить. При этом для каждого принятого вектора $\vec{r_i}$ вычисляется синдром $\vec{r_i} = \vec{s_i}H^T$. Поскольку $\vec{r_i} = \vec{v_i} + \vec{e_i}$, где $\vec{v_i}$ — кодовое слово, а $\vec{e_i}$ — вектор ошибки, то $\vec{s_i} = \vec{e_i}H^T$. Затем с помощью таблицы по синдрому определяется вектор ошибки, с помощью которого определяется переданное кодовое слово. При этом таблица получается гораздо меньше, чем при использовании предыдущего метода.

Циклические коды — это подкласс линейных кодов, обладающие тем свойством, что циклическая перестановка символов в кодированном блоке дает другое возможное кодовое слово того же кода.

Циклическим кодом является линейный код, обладающий следующим свойством: если $\vec{v}$ является кодовым словом, то его циклическая перестановка также является кодовым словом.

Слова циклического кода удобно представлять в виде многочленов. Например, кодовое слово$\vec{v} = (v_0, v_1, ..., v_{n-1})$ представляется в виде полинома $v(x) = v_0 + v_1 x + ... + v_{n-1}x^{n-1}$. При этом циклический сдвиг кодового слова эквивалентен умножению многочлена на x по модулю $x^n - 1$.

Коды БЧХ являются одним из подклассов циклических блочных кодов. Их отличительное свойство — возможность построения кода БЧХ с минимальным расстоянием не меньше заданного. Это важно, потому что, вообще говоря, определение минимального расстояния кода есть очень сложная задача.

Коды Рида—Соломона являются одним из подклассов циклических блочных кодов. Это недвоичные циклические коды, позволяющие исправлять ошибки в блоках данных. Элементами кодового вектора являются не биты, а группы битов (блоки). Очень распространены коды Рида-Соломона, работающие с байтами (октетами).

Свёрточные коды, в отличие от блоковых, не делят информацию на фрагменты и работают с ней как со сплошным потоком данных.

Свёрточные коды, как правило, порождаются дискретной линейной инвариантной во времени системой. Поэтому, в отличие от большинства блоковых кодов, свёрточное кодирование — очень простая операция, чего нельзя сказать о декодировании.

Кодирование свёрточным кодом производится с помощью регистра сдвига, отводы от которого суммируются по модулю два. Таких сумм может быть две (чаще всего) или больше.

Декодирование свёрточных кодов, как правило, производится по алгоритму Витерби, который пытается восстановить переданную последовательность согласно критерию максимального правдоподобия.

4. Ход работы

1. **Коды Хэмминга** Первый способ: используя функции encode/decode.

```
count = 4;
bitNum = 4;
n = 2^(bitNum - 1) - 1;
m = n - (bitNum - 1);

msg = randerr(count, bitNum, 0:bitNum);
code = encode(msg, n, m, 'hamming/binary');
\%code(1) = xor(code(1),1);
result = decode(code, n, m, 'hamming/binary');
```

   В данном примере мы составили 4 посылки длиной 4 бита. Все посылки были закодированы кодом (7,4). Т.к. в канале нет помех то на выходе мы получаем нашу исходную посылку. Если мы раскомментируем данную строку $code(1) = xor(code(1), 1);$, то мы тем самым инвертируем первый бит нашей посылки. После раскодирования мы все равно получим исходную посылку т.к. данный код Хэмминга обнаруживает и исправляет ровно 1 ошибку.

Второй способ: используя проверочную и генераторную матрицу.

```
count = 1;
m = 3;
[h, g] = hammgen(m, [1 0 1 1]);
msg = randerr(count, m + 1, 0:(m + 1));
code = msg*g;
for j = 1:count
    for i = 1:(2^m - 1)
        code(j, i) = mod(code(j, i), 2);
    end
end
%code(7) = xor(code(7),1);

syndrom = code*h';
for j = 1:count
    for i = 1:m
        syndrom(j, i) = mod(syndrom(j, i), 2);
    end
end
```

Данный код выполняет ту же роль, что и предыдущий. Исключение составляет тот факт, что мы н исправляем ошибку сразу а получаем номер ошибочного бита в массиве syndrom. При отсутствии ошибки в массиве syndrom записаны нули.

2. **Циклический код**

```
count = 1;
bitNum = 4;
n = 2^(bitNum - 1) - 1;
m = n - (bitNum - 1);

c = cyclpoly(n,m,'max');
[h,g] = cyclgen(n,c);

msg = randerr(count, bitNum, 0:bitNum);
code = msg*g;
for j = 1:count
    for i = 1:n
        code(j, i) = mod(code(j, i), 2);
    end
end
%code(7) = xor(code(7),1);

syndrom = code*h';
for j = 1:count
    for i = 1:(bitNum - 1)
        syndrom(j, i) = mod(syndrom(j, i), 2);
    end
end
```

В данном примере мы генерируем циклический код. С помощью функции *cyclpoly* мы получаем вектор коэффициентов, а с помощью *cyclgen* опять получем прове-

рочную и генераторную матрицу. Циклический код также может обнаружить и исправить 1 ошибку.

3. **Код БЧХ**

```
nwords = 1;
M = 4;
n = 2^bitNum - 1;
k = 5; \%message lenght
m = n - (bitNum - 1);

t = bchnumerr(n,k); \% Find the error-correction capability.

msg = gf(randi([0 1],nwords,k));
code = bchenc(msg, n, k);

noisycode = code + randerr(nwords,n,1:t); \%from 1 to t erro

msgRx = bchdec(noisycode,n,k);
```

Коды БЧХ исправляют больше 1 ошибки. В случае сообщения длины 5 мы можем исправить 3 ошибки. Количество ошибок, которые мы можем исправить находит функция *bchnumerr*. Зашумляя посылку произвольным количеством ошибок от 1 до t мы все равно получаем верное сообщение. Однако если количество ошибок будет больше, то декодер будет ошибаться.

4. **Код Рида-Соломона**.

```
nwords = 3;
m = 3;              \% Number of bits per symbol
n = 2^m - 1;        \% Codeword length
k = 3;              \% Message length

msg = gf(randi([0 (2^m - 1)],nwords,k),m); \%Create messages
\%Encode the message with a (7,3) RS code
code = rsenc(msg,n,k);

errors = gf([2 0 0 0 0 0 0; 3 4 0 0 0 0 0; 5 6 7 0 0 0 0],m);
noisycode = code + errors;

[result,cnumerr] = rsdec(noisycode,n,k);
\% RS code cannot correct more than two errors.
```

Отличие данного кода в том, что он позволяет исправлять ошибки не только в бинарных последовательностях. В данном примере показан пример кода Рида-Соломона (7,3). Мы можем в посылке из 3 символов, по 3 бита, обнаружить 2 ошибки. В массиве cnumerr, после выполнения программы, будет записано [1;2;-1]. Эти цифры говорят о том, что в первой посылке есть 1 ошибка, во второй 2, а в третей больше 2.

5. **Сверточные коды**

```
count = 4;
```

```
t = poly2trellis(3, [5 7]);

msg = randi([0 1],count,1);
code = convenc(msg, t);

result = vitdec(code, t, 2, 'trunc', 'hard');
```

Сверточные коды работают с каждым битом индивидуально. Они увеличивают разрядность схемы. Функция *poly2trellis* строит решетку кода, используя ее мы можем закодировать любое сообщение. В данном примере у нас код 2:1.

5. Вывод

В данной лабораторной работе мы изучили основные методы и свойства помехоустойчивого кодирования.

Кодировать сообщения можно разными способами. Во-первых, мы рассмотрели коды Хэмминга, которые очень просты в использовании. При умножении на проверочную матрицу мы стразу получаем номер ошибочного бита. К недостаткам относится то, что мы можем обнаружить лишь 1 ошибку.

Во-вторых, циклические коды. Они незаменимы при необходимости передавать информацию в каналах связи, в которых отсутствует возможность повторной передачи данных. Циклические коды применяются при записи и считывании на HDD, CD и DVD, при использовании USB-портов для обмена информацией, при передаче аудио и видео информации.

Коды БЧХ исправляют сразу несколько ошибок. Коды Рида-Соломона позволяют корректировать не только битовые данные, но и недвоичные коды. Свёрточные коды используются для надежной передачи данных: видео, мобильной связи, спутниковой связи. Они используются вместе с кодом Рида-Соломона и другими кодами подобного типа.