МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный университет
имени Франциска Скорины»

И.В.МАСКАЕВА

МАТЕМАТИЧЕСКИЕ МОДЕЛИ ПРОЦЕССОВ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

ЗАДАНИЯ

к контрольной работе

Гомель 2005

Автор-составитель:

И. В. Маскаева, доцент, кандидат технических наук

Рецензенты:

Е.А.Дей, доцент, кандидат физико-математических наук

Кафедра автоматизированных систем обработки информации учреждения образования «Гомельский государственный университет имени Франциска Скорины»

Рекомендовано к изданию научно-методическим советом учреждения образования «Гомельский государственный университет имени Франциска Скорины» 30 марта 2005 года, протокол № 7

Задания к контрольной работе по дисциплине «Математические модели процессов информации и управления» разработаны в соответствии с учебным планом подготовки специалистов по специальности I 53 01 02 - «АСОИ». Задания адресованы студентам заочного факультета и включают теоретические сведения, варианты контрольных заданий и примеры их выполнения.

Содержание

Введение

Выполнение контрольных работ по дисциплине “Математические модели процессов информации и управления” предполагает обучение студентов основам работы с математическими моделями процессов передачи информации. Решение соответствующих задач позволит освоить основы кодирования информации и получить базовые сведения по защите информации.

Выполнение контрольной работы включает:

1. Изучение студентами необходимого теоретического материала по теме работы.
2. Постановку задачи в соответствии с заданием
3. Решение контрольного примера

Выбор вариантов определяется последней цифрой зачетной книжки.

В отчете о работе указываются:

- фамилия, имя, отчество студента;
- код группы;
- содержание поставленной задачи;
- ход решения задачи и результаты расчетов.

Структура отчета по контрольной работе:

1. Тема задания.
2. Цель работы.
3. Постановка задачи.
4. Описание хода решения задачи.
5. Результаты расчетов.
6. Выводы по полученным результатам.
7. Алгоритм и текст программы.

Тема 1 Оптимальное кодирование информации

Энтропией дискретного ансамбля $X=\{x_i,p(x_i)\}$ называется величина

$$\mathbf{H(x) = M[I(x)] = \sum I(x_i)*p(x_i) = - \sum p(x_i)*\log p(x_i)}$$

По определению получается, что энтропия представляет собой среднеожидаемое количество информации от системы.

Пусть имеется дискретный ансамбль X={x, p(x)}, требуется построить неравномерный двоичный код. Причем, необходим такой код, который обладает свойством однозначной декодируемости.

Определение: код обладает свойством однозначной декодируемости, если он допускает разделение кодовой последовательности на отдельные кодовые слова, без использования каких-либо дополнительных символов.

Пример.

Пусть имеется множество X={1,2,3,4}.Кодировать будем с помощью двоичного алфавита A={0,1}.Пусть имеются коды c_1={00,01,10,11}

c_2={1,01,001,000}

c_3={1,10,100,000}

c_4={0,1,10,01}

Из этих множеств только c_4 не обладает свойством декодируемости. Код c_2 из неравномерных кодов наиболее легко расшифровывается, т.к. ни одно кодовое слово не является началом другого слова. Такие коды называются префиксными.

Префиксные коды всегда являются однозначно декодируемыми.

Префиксность – достаточное, но не необходимое условие декодируемости (например, c_3 – код однозначно декодируемый, но не префиксный).

Префиксные коды удобно представить в виде кодовых деревьев

Определение: Код называется древовидным, если в качестве кодовых слов он содержит только кодовые слова соответствующие концевым вершинам дерева.

В дальнейшем будем рассматривать только префиксные коды.

При сравнении кодов логично в качестве критерия выбрать среднюю длину кодового слова.

Определение: Пусть имеется дискретный ансамбль X={x,p(x)} и пусть для его кодирования выбран код c= {c_1,c_2,…,c_m}. Обозначим через l_i длину i-го кодового слова. Тогда средняя длина кодового слова:

$$l_{cp}=M[l(x)]=\sum l_i*p(x_i)$$

Сравним среднюю длину кодовых слов для рассмотренного примера. Будем считать, что вероятности появления кодовых слов равны.

Код c_1 является равномерным $l_{cp}=2$.

Код c_2 является неравномерным $l_{cp}=2,25$.

Код c_3 также не является равномерным $l_{cp}=2,25$.

Первый аспект: не всегда и не любой неравномерный код будет более эффективным. Однако, если бы короткие слова имели более высокую вероятность появления, то средняя длина кодового слова была бы меньше. Если вероятность появления кодовых слов неодинакова, то неравномерные коды более эффективны и дают меньшую среднюю длину.

Второй аспект при сравнения кодов – сложность реализации кодирования.

Причем, сложность зависит от области применения. Если кодирование реализуется на универсальном компьютере, то оперативная память неограниченна. Если код реализован на интегральной схеме, то память является определяющим фактором.

Кроме того, коды часто являются составной частью более сложного алгоритма, поэтому имеет значение сложность модификации кода, при изменении статических данных об источнике.

Таким образом, задача неравномерного кодирования есть построение однозначного кода с наименьшей длинной кодового слова, при заданных ограничениях на сложность.

Принципы построения оптимальных неравномерных кодов:

1) Каждый элементарный символ должен переносить максимальное количество информации. Для этого нули и единицы должны встречаться в закодированном тексте приблизительно в равном количестве.

2) Буквы, имеющие наибольшую вероятность, должны кодироваться более короткими словами.

Первый оптимальный код – **код Шеннона-Фано** имеет следующий алгоритм:

Шаг 1. Все вероятности сортируются в порядке убывания.

Шаг 2. Все сообщения разбивают на две группы так, чтобы суммарные вероятности группы были приблизительно равными. Первой группе присваиваем ноль, а второй единицу.

Шаг 3. Если в подгруппе количество символов больше единицы, то переходим к шагу два. Если во всех группах сообщений по одному, то построение кодов закончено.

Второй оптимальный код – **код Хаффмана.** Код Хаффмана строится следующим образом:

Шаг 1. Буквы располагают в порядке убывания вероятностей.

Шаг 2. Складывают вероятности двух последних букв.

Шаг 3. Ряд переписывают с учетом новой суммарной вероятности. Эти действия повторяют пока суммарная вероятность не станет равной единице.

Чтобы не упорядочивать вероятности для построения кода Хаффмана используют следующий алгоритм:

1) Инициализация. Список подлежащих обработке слов состоит из $M=M_0$ узлов, каждому из которых предписана некоторая вероятность.

2) Организуем цикл пока $M>1$.

Находим два узла с наименьшими вероятностями, исключаем их из списка.

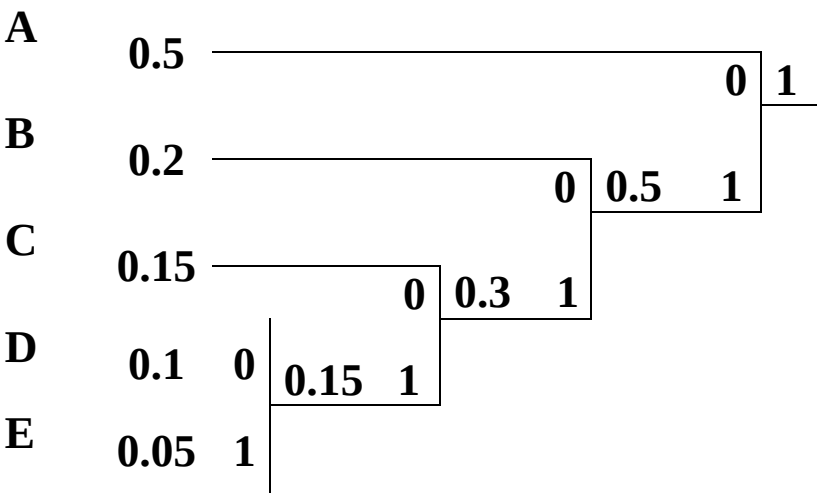
Вместо них вводим новый узел, вероятность которого равна сумме исключенных узлов.

Новый узел связывается ребрами с исключенными узлами. Уменьшаем M=M-1.

В результате получится кодовое дерево.

Существует совмещенной способ построения кода Хаффмана:

Пусть имеется пять символов, которые имеют заданные вероятности. Соответствующее дерево кода Хаффмена будет иметь вид:



Достоинство кода Хаффмана в том, что можно использовать не только двоичные алфавиты.

Свойства оптимальных кодов:

1) Если все сообщения равновероятны, то оптимальным будет равномерный код. При этом, если количество сообщений равно целой степени двойки, то средняя длина кодового слова будет равна энтропии.

2) Слова с одинаковой вероятностью имеют одинаковую длину.

3) Кодовые слова с наименьшей вероятностью имеют наибольшую длину.

4) Если кодовый алфавит состоит из k символов, тогда средняя длина кодового слова $l_{ср}=H/\log_2 K$.

5) Можно рассчитать коэффициент сжатия кода

$$\mu=H_{max}/(l_{ср}*\log_2 K)$$

для двоичных кодов $\mu=H_{max}/l_{ср}$.

Коэффициент относительной эффективности кода $k_{оэ}= H/(l_{ср}*\log_2 K)$.

Задание №1 Построение оптимальных кодов

Цель работы: выработка практических навыков построения оптимальных кодов и оценки характеристик кодов

Работа выполняется на персональном компьютере с использованием языков программирования высокого уровня.

1. Постройте программу, которая бы позволила ввести сообщение произвольной длины.
   a. Программа должна выделить буквы алфавита введенного текста, подсчитать и выдать частоту появления этих букв.
   b. Определить энтропию, приходящуюся в среднем на одну букву; длину кода при равномерном кодировании и избыточность.
   c. Программа должна обладать некоторым интерфейсом для удобства работы пользователя.
2. Имея программу необходимо запустить ее и, пользуясь подсказками меню, ввести произвольный связный текст на русском языке. Это может быть пословица, стихотворение или произвольный текст.

Используя результаты, выданные компьютером, следует:

1) проанализировать алфавит введенного сообщения: количество символов алфавита, значение энтропии H, длину кода при равномерном кодировании, избыточность R при однобуквенном кодировании;

2) построить вручную коды Шеннона-Фано и Хаффмена (используя полученные статистические данные);

3) определить среднюю длину кода $l_{ср}$ для кодов Шеннона-Фано и Хаффмена;

4) сравнить полученное значение $l_{ср}$ с оптимальным H, выданным компьютером;

5) определить коэффициент сжатия и коэффициент относительной эффективности.

Пример выполнения задания

Распечатка результатов решения задачи:

```
КУКУШКА_КУКУШОНКУ_КУПИЛА_КАПЮШОН

Символ Вероятность
     К     0.25000
     У     0.18750
     Ш     0.09375
     А     0.09375
     _     0.09375
     О     0.06250
     Н     0.06250
     П     0.06250
     И     0.03125
     Л     0.03125
     Ю     0.03125

Длина сообщения: 32
Кол-во символов алфавита: 11
Энтропия: 3.13205
Энтропия на 1 символ: 0.09788
Длина кода при равномерном кодировании: 4
Избыточность: 0.21699
```

Код Шеннона-Фано:

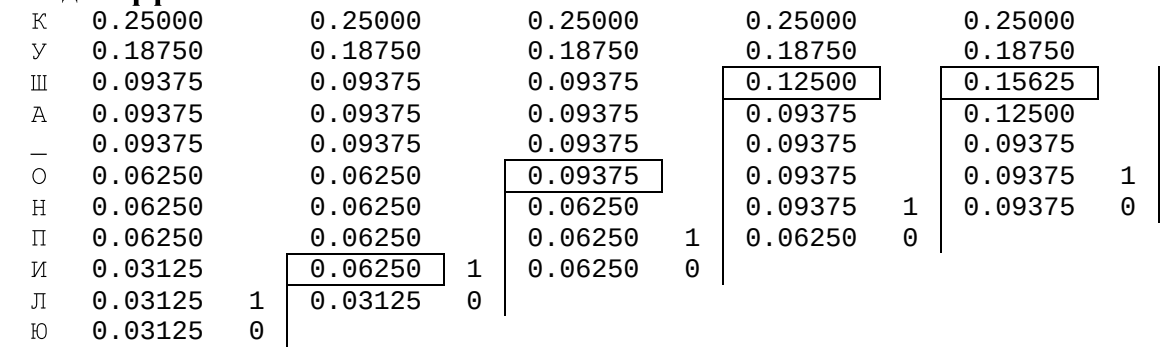
К	0.25000	0	0				00
У	0.18750		1				01
Ш	0.09375	1	0	0			100
А	0.09375			1	0		1010
_	0.09375				1		1011
О	0.06250		1	0	0		1100
Н	0.06250				1		1101
П	0.06250			1	0	0	11100
И	0.03125					1	11101
Л	0.03125				1	0	11110
Ю	0.03125					1	11111

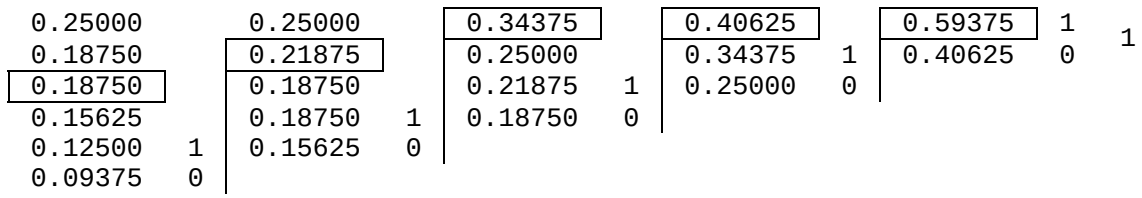
$$\bar{l} = \sum_{i=1}^{k} p_i \cdot l_i = 3.1875$$

$$\mu = \frac{H_{\max}}{\bar{l} \cdot \log_2 k} = 0.36275$$

$$K_{ОЭ} = \frac{H}{\bar{l} \cdot \log_2 k} = 0.28404$$

Код Хаффмана:





К	0.25000	10
У	0.18750	00
Ш	0.09375	010
А	0.09375	1110
_	0.09375	1111
О	0.06250	1100
Н	0.06250	0110
П	0.06250	0111
И	0.03125	11010
Л	0.03125	110111
Ю	0.03125	110110

$$\bar{l} = \sum_{i=1}^{k} p_i \cdot l_i = 3.1875$$

$$\mu = \frac{H_{\max}}{\bar{l} \cdot \log_2 k} = 0.36275$$

$$K_{ОЭ} = \frac{H}{\bar{l} \cdot \log_2 k} = 0.28404$$

Вывод: коды Шеннона-Фано и Хаффмана
являются оптимальными префиксными кодами, т.к. ни одно кодовое слово в них не является началом другого, и слова с большей вероятностью содержат меньшее число символов.

Исходный текст программы:

```
program V2;
uses crt;
var ch:array[1..32] of char;
    p:array[1..32] of real;
    i,j,k,lc:byte;
    st:string;
    n:byte absolute st;
    h,p1,lcr:real;
function upcaser(ch:char):char;
  const b:array[1..32] of char=('А','Б','В','Г','Д','Е','Ж','З','И',
                                'К','Л','М','Н','О','П','Р','С','Т',
                                'У','Ф','Х','Ц','Ч','Ш','Щ','Ъ','Ы',
                                'Ь','Э','Ю','Я','_');
        m:array[1..32] of char=('а','б','в','г','д','е','ж','з','и',
                                'к','л','м','н','о','п','р','с','т',
                                'у','ф','х','ц','ч','ш','щ','ъ','ы',
                                'ь','э','ю','я',' ');
  var i:byte;
  begin
    i:=1;
    while (i<=32) and (m[i]<>ch) do
      inc(i);
    upcaser:=b[i];
```

```
  end;

 .  .  .  .  .  .  .  .  .  .

  writeln('Длина кода при равномерном кодировании: ',lc);
  writeln('Избыточность: ',1-h/lc:1:5);
  readkey;
end.
```

Тема 2 Помехоустойчивое кодирование

Линейные блочные коды

Определение: Линейным блочным кодом называется множество последовательности длины n названных кодовыми словами, которое характеризуется тем, что сумма двух кодовых слов есть кодовое слово. Такие коды называются линейными ((n,k)- коды).

n – длина кода

k – информационная часть

Операция сложения и умножения для кодовых слов осуществляются по модулю два (mod 2).

Пример:

1+1+1(mod 2)=1

1+1+0(mod 2)=0

1*1+1*0(mod 2)=1

Желательным качеством этих кодов является систематичность. Это означает, что в качестве кодового слова идут k информационных символов, а затем идут n-k проверочных.

Определение: Блочный линейный n,k-код, обладающий свойством систематичности называется линейным систематическим n,k- кодом.

Простейшим случаем линейного блочного кода является код с проверкой на четность.

Способы задания линейных кодов. Свойства линейных кодов

Первый способ задания – перечисление кодовых слов.

Второй способ задания – система проверочных условий. Для всех проверочных символов записывается:

$$e_j=\sum x_i * h_{ij}, \quad j=1,n-k$$

e_j – проверочный символ

x_j – символ информационного сообщения

h_{ij} – коэффициент, принимающий значение 0 или 1, в соответствии с правилами конкретного кода.

Пример: пусть имеется (5,3) – код

$e_1 = 0*x_1+1*x_2+1*x_3 = x_2+x_3$

$e_2 = 1*x_1+1*x_2+0*x_3 = x_1+x_2$

Кодовое слово 011 → $x_1=0$ $e_1=0$

$x_2=1$ $e_2=1$

$x_3=1$ $e_3=1$

011→01101

Третий способ задания – матричный. Он основан на построении порождающей и проверочной матрицы.

Определение: Алфавит из двух символов {0,1} вместе со сложением и умножением по mod 2 называется полем из двух элементов.

Обозначается GF(2).

Векторное пространство над полем GF(2) включает в себя 2^n векторов. Его подпространством Vk является множество из 2^k кодовых слов.

Любое векторное пространство однозначно определяется базисом. Для Vk базис состоит из k линейно независимых векторов.

Таким образом, линейный n,k – код полностью определяется набором из k кодовых слов принадлежащих этому коду. Этот набор обычно представляется в виде матрицы, которую называют **порождающей**. Ее обозначение G(n,k).

Пример: (5,3) – код

$$G(5,3)=\begin{pmatrix} 1\,0\,0\,0\,1 \\ 1\,1\,0\,1\,0 \\ 0\,1\,1\,0\,1 \end{pmatrix}$$

Возможно множество вариантов для порождающей матрицы. Количество возможных вариантов определяется по формуле:

$$\mathbf{M_{n,k}=\prod(2^k*2^i)}$$

Для удобства работы в порождающей матрице, ее записывают в канонической или систематической форме:

$$G(n,k)=||I_k\ R_{k,r}||, \text{ где } r=n-k$$

$$G(5,3)=\left\|\begin{matrix}1\ 0\ 0\ 0\ 1\\0\ 1\ 0\ 1\ 1\\0\ 0\ 1\ 1\ 1\end{matrix}\right\|$$

Порождающая матрица в канонической форме может быть получена из любой другой порождающей матрицы посредством перестановки строк и заменой строк на их линейные комбинации.

Проверочная матрица строится на основе порождающей и в общем виде имеет вид:

$$H(n,k)=||\ R^{T}_{k,r,}\ I_r||$$

$$H(5,3)=\left\|\begin{matrix}0\ 1\ 1\ 1\ 0\\1\ 1\ 0\ 0\ 1\end{matrix}\right\|$$

Свойства кодов:

1) Произведение любого кодового слова на транспонированную проверочную матрицу есть нулевой вектор из r элементов, r=n-k

$$X*H^{T}(n,k)=[0\ldots0]$$

Пример: пусть кодовое слово X=10001

$$X*H^{T}=(10001)\left|\begin{matrix}0\ 1\\1\ 1\\1\ 0\\1\ 0\\0\ 1\end{matrix}\right|=(00)$$

2) Произведение ошибочного кодового слова X на транспонированную проверочную матрицу будет ненулевой вектор.

Вектор, полученный в результате умножения, называется синдромом и обозначается s(x).

$$\mathbf{s(x)=X*}H(n,k)\neq\mathbf{0}$$

3) Между порожденной и проверочной матрицей существует однозначное соответствие: $\mathbf{G(n,k)*\ H^{T}(n,k)=0}$

4) Произведение информационного сообщения на порожденную матрицу дает кодовое слово кода.

5) Вес любой строки подматрицы $P_{k,r}$ должен быть $W_i \geq d_{min} - 1$.

6) Два кода называются эквивалентными, если их порождающие матрицы получаются одна из другой перестановкой столбцов и линейными комбинациями строк.

7) Кодовое расстояние любого (n,k) – кода удовлетворяет неравенству $d_{min} \leq n\text{-}k+1$

Код, удовлетворяющий равенству, называется кодом с максимальным расстоянием.

Пример: пусть требуется построить помехоустойчивый код, позволяющий передовать15 сообщений. Код должен обнаруживать и исправлять одну ошибку. Требуется передать 15 сообщений. Информационная часть кода 4 бита

$k=4$, $2^r \geq n+1$

$2^r \geq 4+r+1$

$2^r \geq 5+r \rightarrow r=3$

Таким образом, имеем (7,4)- код, $d_{min}=1+1+1$. Это значение позволяет определить количество единиц в проверочной части матрицы.

$W_i = d_{min} - 1 \rightarrow$ записываем порождающую матрицу

$$G(7,4) = \begin{Vmatrix} 1000\ 110 \\ 0100\ 011 \\ 0010\ 101 \\ 0001\ 111 \end{Vmatrix}$$

Проверочная матрица

$$H(7,4) = \begin{Vmatrix} 1\ 0\ 1\ 1\ \ 100 \\ 1\ 1\ 0\ 1\ \ 010 \\ 0\ 1\ 1\ 1\ \ 001 \end{Vmatrix}$$

Исходная последовательность: x=1010 101

По проверочной матрице можно записать формулы для вычисления проверочных символов

$e_1 = x_1 + x_3 + x_4$

$e_2 = x_1 + x_2 + x_4$

$e_3 = x_2 + x_3 + x_4$

Исправление ошибок

Для описания возникших ошибок, используется вектор ошибки. Это есть последовательность длиной n с единицами в тех позициях, где произошли ошибки.

Пусть принята последовательность с ошибкой $\tilde{x} = x + er$, где

x – переданная последовательность

er – вектор ошибки

Для обнаружения ошибок $s = \tilde{x} * H^T = (x + er) * H^T = x * H^T + er * H^T = er * H^T$

Это означает, что синдром принятой последовательности зависит от ошибки и не зависит от переданного кодового слова. Задача декодера, используя эту зависимость, определить координаты ошибки. Для этого составляется таблица в соответствии с синдромом и координат ошибки.

Понятие циклического кода

Циклические коды являются разновидностью линейных кодов.

Определение: Линейный код называется циклическим, если для любого кодового слова $(x_0, x_1, \ldots, x_n)$ циклическая перестановка символов $(x_1, x_2, \ldots, x_n, x_0)$ также является кодовым словом.

Для построения циклических кодов требуется перейти от векторного описания к полиноминальному. Для этого символы, составляющие кодовые слова интерпретируются как коэффициенты полинома в порядке возрастания степени.

Кодирования сообщения в кодовое слово производится умножением исходного полинома на другой.

Рассмотрим, как производится деление полиномом и операция mod.

1) Деление

$$\begin{array}{l|l} x^5+x^2+x+1 & x^2+1 \\ \underline{x^5+x^3} & x^3+x+1 \\ x^3+x^2+x+1 & \\ \underline{x^3+x} & \\ x^2+1 & \\ \underline{x^2+1} & \\ 0 & \end{array}$$

2)Пусть необходимо взять модуль $x^5+x^2(\text{mod } x^2+1)$

Когда записывает так, то результатом является остаток от деления первого полинома на второй. Если степень первого полинома меньше степени второго, то результатом будет просто первый полином.

Важным свойством полиномиального представления кодов является возможность осуществить циклический сдвиг вправо, путем умножения кодового слова на x и нахождение остатка от деления x^n+1.

Пример: требуется осуществить сдвиг кодового слова(1011)

n=4

1) Записываем кодовое слово $1+x^2+x^3$ в виде полинома
2) Умножаем на x: $x+x^3+x^4$
3) $\begin{array}{l|l} x^4+x^3+x & \underline{x^4+1} \\ \underline{x^4+1} & 1 \\ x^3+x+1 & \\ (1101) & \end{array}$

Порождающий полином для циклических кодов

Возникает вопрос: «Как получить кодовое слово из исходного сообщения?» Для этого полином, соответствующий сообщению, умножается на специальный полином, который называется порождающим для данного кода.

Теорема: Полином g(x) является порождающим для линейного циклического кода длины n тогда и только тогда, когда g(x) делит многочлен x^n+1. Это означает, что для получения

порождающего полинома требуется разложить на множители x^n+1. Из этих множителей выбираем полином степени n-k, где k – длина сообщения.

Пример: требуется найти порождающий многочлен линейного циклического кода n=7. Этот код должен осуществлять кодирование сообщений длины k=4, тогда количество проверочных символов r = n – k = 3.

1) Разложим x^n+1 на множители для n=7:

$x^7+1=(x+1)*(x^3+x+1)*(x^3+x^2+1)$

2) Надо выбрать порождающий полином. Выбираем, например, $g(x)=x^3+x+1$

3) Закодируем для примера V=1010.

$c(V)=V*g(x)=(x^3+x+1)*(1+x^2)=x^3+x+1+x^5+x^3+x^2=x^5+x^2+x+1=1+x+x^2+x^5$

Кодовое слово c(V)=(1110010)

Рассмотрим алгоритм декодирования и нахождения ошибок.

Процедура декодирования следующая:

1) Находим синдромный полином

$S(V)=c(V)(\mathrm{mod}\ g(x))$

Если S(V)=0, то слово передано правильно, если $S(V)\neq 0$, требуется определить и исправить ошибки.

2) Начиная с i=1 рассчитаем $s_i(V)=x^i*s(V)(\mathrm{mod}\ g(x))$. Нахождение s_i продолжаем до тех пор, пока вес s_i не станет меньше либо равен количеству ошибок, который способен найти код.

Пусть $s_m(V)$ имеет требуемый вес, тогда полином ошибки

$e(x)=x^{n-m}*s_m(x)(\mathrm{mod}\ x^{n+1})$.

Для исправления ошибок надо сложить полученное кодовое слово с ошибкой c’(v)=c(V)+e(x) – правильное кодовое слово.

Задание № 2 Построение помехоустойчивых кодов

Цель работы: выработка практических навыков построения помехоустойчивых кодов и определения ошибок.

Задача 1. Линейные блочные (n,k)-коды

- Определить длину информационной части кода и проверочной части.
- Построить порождающую матрицу.
- Построить проверочную матрицу в систематическом виде (для кодов Хэмминга и в упорядоченном виде).
- Сформировать системы проверочных и синдромных уравнений.
- Составить таблицу синдромов.
- Сформировать кодовое слово исследуемого кода, проверить его на правильность.
- Ввести в кодовое слово одну ошибку. Показать, как определить ошибку в кодовом слове и как ее исправить.
- Показать на примере, сколько ошибок код не может обнаружить.

Задача 2. Циклические коды

- Определить порождающий полином для кода
- Выбрать любое слово и закодировать его.
- Внести одну или несколько ошибок в кодовое слово. Построить синдром и через синдромы определить полином ошибки. Построить правильное кодовое слово.
- Построить порождающую и проверочную матрицы .
- Задать другое слово, закодировать его с помощью порождающей матрицы и проверить правильность с помощью проверочной матрицы.

Варианты задания :

0. Код Хэмминга для кодирования 6 сообщений
1. Код, исправляющий 1 ошибку и способный передать 7 сообщений

2. Код, исправляющий 1 ошибку и способный передать 20 сообщений
3. Код Хэмминга для кодирования 10 сообщений
4. Код, обнаруживающий 2 ошибки и исправляющий 1 ошибку, способный передать 50 сообщений
5. Код, обнаруживающий 2 ошибки и исправляющий 1 ошибку, способный передать 15 сообщений
6. Код, обнаруживающий 2 ошибки и исправляющий 1 ошибку, способный передать 30 сообщений
7. Код, обнаруживающий 2 ошибки и исправляющий 1 ошибку, способный передать 7 сообщений
8. Код, обнаруживающий и исправляющий 1 ошибку и способный передать 15 сообщений
9. Код, обнаруживающий и исправляющий 1 ошибку и способный передать 120 сообщений

Пример выполнения задания

Задача 1. Линейные блочные (n,k)-коды.
Код, обнаруживающий и исправляющий 1 ошибку и способный передать 15 сообщений

Т.к требуется передать 15 сообщений, то k = 4 – информационная часть кода
r = 3 – проверочная часть кода
Таким образом имеем (7,4) – код.
$d_{min} = 1 + 1 + 1 = 3$

Порождающая матрица

$$G(7,4) = \begin{Vmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{Vmatrix}$$

Проверочная матрица в систематическом виде

$$H(7,4)=\begin{Vmatrix}1&0&1&1&1&0&0\\1&1&0&1&0&1&0\\0&1&1&1&0&0&1\end{Vmatrix}$$

Проверочная матрица для кодов Хэмминга

$$H(7,4)=\begin{Vmatrix}0&1&1&1&1&0&0\\1&1&0&1&0&1&0\\1&0&1&1&0&0&1\end{Vmatrix}$$

Проверочная матрица в упорядоченном виде

$$H(7,4)=\begin{Vmatrix}0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\1&0&1&0&1&0&1\end{Vmatrix}$$

Система проверочных уравнений

$$e_1 = x_1 + x_3 + x_4$$
$$e_2 = x_1 + x_2 + x_4$$
$$e_3 = x_2 + x_3 + x_4$$

Таблица синдромов

$$er_1 = 1000000 \quad s_1 = er_1 * H^T = 110 = 6$$
$$er_2 = 0100000 \quad s_2 = er_2 * H^T = 011 = 3$$
$$er_3 = 0010000 \quad s_3 = er_3 * H^T = 101 = 5$$
$$er_4 = 0001000 \quad s_4 = er_4 * H^T = 111 = 7$$
$$er_5 = 0000100 \quad s_5 = er_5 * H^T = 100 = 4$$
$$er_6 = 0000010 \quad s_6 = er_6 * H^T = 010 = 2$$
$$er_7 = 0000001 \quad s_7 = er_7 * H^T = 001 = 1$$

Пусть требуется передать слово 1001. Составим кодовое слово кода:

$$x = x_4 * G(7,4) = 1001001$$

$$e_1 = x_1 + x_3 + x_4 = 0$$
$$e_2 = x_1 + x_2 + x_4 = 0$$
$$e_3 = x_2 + x_3 + x_4 = 1$$

Пусть мы получили кодовое слово 10010**1**1. Требуется проверить, есть ли ошибки?

$s = \tilde{x} * H^T = 010$ => $er = 0000010$ => ошибка в 6-м символе

Проверим, исправит ли код 2 ошибки:

$er' = 0100010$

$s' = er' * H^T = 001$ => $er = (0000001)$

Код не исправит 2-х ошибок

Задача 2. Циклические (n,k)-коды.
Код, обнаруживающий и исправляющий 1 ошибку и способный передать 15 сообщений

Т.к требуется передать 15 сообщений, то k = 4 – информационная часть кода

$2^k \le \frac{2^n}{n+1}$ => n = 7 => r = 3

r = 3 – проверочная часть кода
Таким образом имеем (7,4) – код.

Определяем порождающий полином для кода

1). Разлагаем $x^7 + 1$ на множители
$x^7 + 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1)$

2). Выбираем порождающий многочлен

$g(x) = x^3 + x + 1$

Закодируем слово 0011 => $x^2 + x^3$

$c(v) = v*g(x) = (x^3 + x + 1)(x^2 + x^3) = x^5 + x^6 + \underline{x}^3 + x^4 + x^2 + \underline{x}^3 = x^2 + x^4 + x^5 + x^6$
c(v) = 0010111

Внесём ошибку в кодовое слово. Пусть мы передали c(v) = 0010111, а

получили последовательность c’(v) = 0**1**10111 = $x + x^2 + x^4 + x^5 + x^6$

$g(x) = x^3+x+1$

Построим синдром

$s(v) = c'(v)(\text{mod } g(x)) = x \neq 0$

$s_1(v) = x*s(v) \text{ (mod } g(x)) = x^2$
$s_2(v) = x^2*s(v) \text{ (mod } g(x)) = x + 1$
$s_3(v) = x^3*s(v) \text{ (mod } g(x)) = x^2 + x$
$s_4(v) = x^4*s(v) \text{ (mod } g(x)) = x^2 + x + 1$
$s_5(v) = x^5*s(v) \text{ (mod } g(x)) = x^2 + 1$
$s_6(v) = x^6*s(v) \text{ (mod } g(x)) = 1 \Rightarrow$ вес равен 1

Определим полином ошибки

$e(x) = x^{7-6}*s_6(v) \text{ (mod } (x^7+1)) = x$

Построим правильное кодовое слово

$c(v) = c'(v) + e(x) = x^2 + x^4 + x^5 + x^6$

Построим порождающую матрицу

$$G(7,4) = \begin{Vmatrix} g(x) \\ x*g(x) \\ x^2*g(x) \\ x^3*g(x) \end{Vmatrix}$$

$$G(7,4) = \begin{Vmatrix} x^3+x+1 \\ x^4+x^2+x \\ x^5+x^3+x^2 \\ x^6+x^4+x^3 \end{Vmatrix}$$

$$G(7,4) = \begin{Vmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{Vmatrix}$$

$h(x) = (x^7+1)/g(x) = x^4 + x^2 + x + 1$

Запишем вспомогательную матрицу $H'(7,4)$

$$H'(7,4) = \begin{Vmatrix} h(x) \\ x * h(x) \\ x^2 * h(x) \end{Vmatrix}$$

$$H'(7,4) = \begin{Vmatrix} x^4 + x^2 + x + 1 \\ x^5 + x^3 + x^2 + x \\ x^6 + x^4 + x^3 + x^2 \end{Vmatrix}$$

$$H'(7,4) = \begin{Vmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{Vmatrix}$$

Построим проверочную матрицу

$$H(7,4) = \begin{Vmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{Vmatrix}$$

Пусть требуется передать слово 0101. Составим кодовое слово кода:

$x = x_4 * G(7,4) = 1110010$

Проверим правильность с помощью проверочной матрицы

x*HT = (000)

Тема 3 Защита информации

Перестановочные шифры

1.Простой столбцевой перестановочный шифр

В данном виде шифра текст пишется на горизонтально разграфленном листе бумаги фиксированной ширины, а шифротекст считывается по вертикали. Дешифрирование заключается в записи шифротекста вертикально на листе разграфленной бумаги фиксированной ширины и затем считывании открытого текста горизонтально.

Открытый текст: **Математические модели процессов информации и управления**

М	А	Т	Е	М	А
Т	И	Ч	Е	С	К
И	Е		М	О	Д
Е	Л	И		П	Р
О	Ц	Е	С	С	О
В		И	Н	Ф	О
Р	М	А	Ц	И	И
	И		У	П	Р
А	В	Л	Е	Н	И
Я					

Зашифрованный текст: МТИЕОВР АЯАИЕЛЦ МИВТЧ ИЕИА Л ЕЕМ СНЦУЕМСОПСФИПНАКДРООИРИ

2.Перестановочный шифр с ключевым словом

Буквы открытого текста записываются в клетки прямоугольной таблицы по ее строчкам. Буквы ключевого слова пишутся над столбцами и указывают порядок этих столбцов (по возрастанию номеров букв в алфавите). Чтобы получить зашифрованный текст, надо выписывать буквы по столбцам с учетом их нумерации:

Открытый текст: Прикладная математика Ключ: Шифр

Ш	и	ф	р
4	1	3	2
П	р	и	к
л	а	д	н
а	я	м	а
т	е	м	а
т	и	к	а

Криптограмма: Раяеикнаааидммкплатт

Ключевое слово - последовательность столбцов - известно адресату, который легко сможет расшифровать сообщение.

Подстановочные шифры

1. Шифр Цезаря

В шифре Цезаря каждая буква замещается на букву, находящуюся **k** символами правее по модулю равному количеству букв в алфавите. (Согласно Светонию у Цезаря k=3 и n=50)

$C_k(j)=(j+k)(\mathrm{mod}\ n)$, n - количество букв в алфавите

Очевидно, что обратной подстановкой является

$$C_k^{-1}(j)=C_{n-k}=(j+n-k)(\mathrm{mod}\ n)$$

2. Модулярный шифр.

В этом шифре задаются следующие параметры: n – количество букв в алфавите, два числа a и b, причем a и в должны быть больше или равны нулю, но меньше n и число a должно быть взаимно простым с n. Взаимная простота a и b требуется для того, чтобы каждой букве соответствовал уникальный шифр. Результатом шифрования будет

$$\mathbf{C_{a,b}(j)=(a*j+b)\mathrm{mod}\ n}$$

Пример: Пусть имеется алфавит: а б в г д е ж з. Тогда n=8. В качестве параметров возьмем a=3 и b=5. Зашифруем буквы

$C_{3,5}$(б)=(3*1+5)mod 8=0→а

$C_{3,5}$(г)=(3*3+5)mod 8=6→ж

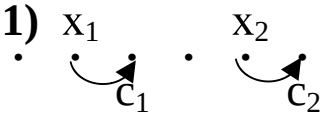
Обобщение подстановочных шифров

1) Для модулярного шифра.

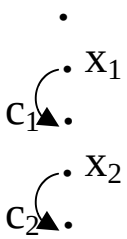
Вместо выражения a*j+b можно использовать более сложный полином, даже нелинейного вида.

2) Обобщение для шифра Цезаря.

Биграмный шифр. Для этого шифра сначала из букв алфавита строится произвольный квадрат. Ключом будет являться размер алфавита и расположение букв в квадрате. Буквы из открытого текста выбираются парами. Их кодировка зависит от их взаимного расположения в ключевом квадрате.

1) 

Код – буква соседняя справа, если буквы в одной строке.

2) 

Код – буква соседняя снизу.

3) 

c_2 x_2 c_1 x_1

3.Шифр Полибия

Наиболее древней из известных является система греческого историка Полибия. Суть шифра состоит в следующем: рассмотрим прямоугольник, часто называемый доской Полибия.

	А	Б	В	Г	Д	Е
А	А	Б	В	Г	Д	Е
Б	Ж	З	И	Й	К	Л
В	М	Н	О	П	Р	С
Г	Т	У	Ф	Х	Ц	Ч
Д	Ш	Щ	Ъ	Ы	Ь	Э
Е	Ю	Я	.	,	-	

Каждая буква может быть представлена парой букв, указывающих строку и столбец, в которых расположена данная буква. Так представления букв В, Г, П, У будут АВ, АГ, ВГ, ГБ соответственно, а сообщение ПРИКЛАДНАЯ МАТЕМАТИКА зашифруется как

ВГВДБВБДБЕАААДВБААЕБЕЕВАААГАААЕВАААГАБВБДА АЕЕ

4.Шифр Вернама (одноразовый блокнот)

В классическом понимании одноразовый блокнот является большой неповторяющейся последовательностью символов ключа, распределенных случайным образом. Первоначально это была одноразовая лента для телетайпов. Отправитель использовал каждый символ ключа для шифрования только одного символа открытого текста. Шифрование представляет собой сложение по модулю **n** (мощность алфавита) символа открытого текста и символа ключа из одноразового блокнота.

5.Шифр Вижинера

Для шифрования строится таблица Вижинера – это матрица n?n, где n – объем алфавита символов.

Первая строка в таблице – все символы алфавита

Вторая строка – все символы со сдвигом на одну букву

Третья строка – все символы со сдвигом на две буквы и т.д.

Затем задается ключ – это слово или набор неповторяющихся букв.

Шифрование:

1) Записываем буквы ключа под шифруемым текстом, повторяя ключ нужное количество раз.

2) Выписываем из таблицы Вижинера первую строку и те строки, которые соответствуют ключу по первым буквам.

3) Буква шифра стоит на пересечении столбца, соответствующего букве открытого текста и строки, соответствующей нужной букве ключевого слова.

Шифр Вижинера можно применять несколько раз. Такой шифр называется *составным шифром Вижинера.*

Другой разновидностью шифра Виженера, имеющей легкозапоминаемый квадрат подстановок, является шифр Бофорта(Beaufort, в некоторой литературе читается как Бьюфорт), названный в честь адмирала сэра Френсиса Бофорта - создателя шкалы для определения скорости ветра. Его строками являются строки квадрата Виженера, записанные в обратном порядке, а первая и последняя строки поменяны местами:

	А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я
А	А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б
Б	Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В
В	В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г
Г	Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д
Д	Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е
Е	Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж
Ж	Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З
З	З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И
И	И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й
Й	Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К
К	К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л
Л	Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М
М	М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н
Н	Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О
О	О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П
П	П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р

Р	Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С
С	С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т
Т	Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У
У	У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф
Ф	Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х
Х	Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч Ц
Ц	Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш Ч
Ч	Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ Ш
Ш	Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ Щ
Щ	Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы Ъ
Ъ	Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь Ы
Ы	Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э Ь
Ь	Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю Э
Э	Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я Ю
Ю	Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А Я
Я	Я Ю Э Ь Ы Ъ Щ Ш Ч Ц Х Ф У Т С Р П О Н М Л К Й И З Ж Е Д Г В Б А

Таблица Бофорта для русского алфавита

Формулой преобразования будет:

$$Bof_d(m_i)=(k_{i \bmod d}-m_i)(\bmod n)$$

Шифрование открытым ключом

В алгоритмах этого типа для шифрования и расшифровки информации используются пара ключей - открытый и закрытый, каждый из которых не может быть получен из другого. Открытый ключ рассылается всем абонентам, закрытый держится в тайне. Для того, чтобы отправить сообщение абоненту, нужно при шифровании использовать его открытый ключ, получатель же расшифровывает сообщение при помощи своего закрытого секретного ключа.

Самым популярным сегодня является алгоритм RSA, предложенный американскими учеными Райвестом, Шамиром и Адельманом в 1977 году.

Алгоритм шифрования данных с открытым ключом RSA (Copyright (c) Roman Lonely 2:5015/52.38)

Понятия:

Простое число - делится только на 1 и на само себя.

Взаимно простые- не имеют ни одного общего делителя, кроме 1.

Результат операции i mod j - остаток от целочисленного деления i на j.

Чтобы использовать алгоритм RSA, надо сначала сгенерировать открытый и секретные ключи выполнив следующие шаги:

1) Выберем два очень больших простых числа p и q.

2) Определим n, как результат умножения p на q (n= p*q).

3) Выберем большое случайное число, которое назовем d. Это число должно быть взаимно простым с результатом умножения (p-1)*(q-1).

4) Определим такое число e, для которого является истинным следующее соотношение (e*d) mod ((p-1)*(q-1))=1.

5) Назовем открытым ключом числа e и n, а секретным ключом - числа d и n.

==========

Теперь, чтобы зашифровать данные по известному ключу {e,n}, необходимо сделать следующее:

Разбить шифруемый текст на блоки, каждый из которых может быть представлен в виде числа M(i)=0,1,2..., n-1(т.е. только до n-1).

Зашифровать текст, рассматриваемый как последовательность чисел M(i) по формуле C(i)=(M(i)^e)mod n. Здесь ^ операция возведения в степень.

Чтобы расшифровать эти данные, используя секретный ключ {d,n}, необходимо выполнить следующие вычисления:
M(i) = (C(i)^d) mod n.

В результате будет получено множество чисел M(i), которые представляют собой исходный текст.

==========

Рассмотрим следующий пример:

Зашифруем и расшифруем сообщение "CAB" по алгоритму RSA. Для простоты будем использовать маленькие числа(на практике - нужно брать намного большие).

1) Выберем p=3 and q=11.

2)Определим n= 3*11=33.

3) Найдем (p-1)*(q-1)=20. Следовательно, d будет равно, например, 3: (d=3).

4) Выберем число e по следующей формуле: (e*3) mod 20=1. Значит, e будет равно, например, 7: e=7.

5) Представим шифруемое сообщение как последовательность чисел в диапазоне от 0 до 32 (так как n =33).

Буква A =1, B=2, C=3.

Теперь зашифруем сообщение, используя открытый ключ {7,33}

C1 = (3^7) mod 33 = 2187 mod 33 = 9;

C2 = (1^7) mod 33 = 1 mod 33 = 1;

C3 = (2^7) mod 33 = 128 mod 33 = 29;

Теперь расшифруем эти данные, используя закрытый ключ {3,33}.

M1=(9^3) mod 33 =729 mod 33 = 3(C);

M2=(1^3) mod 33 =1 mod 33 = 1(A);

M3=(29^3) mod 33 = 24389 mod 33 = 2(B);

Все, данные расшифрованы.

Задание № 3 Применение методов защиты информации

Цель работы: выработка практических навыков криптографической защиты информации.

Задача 1. Использование симметричных алгоритмов шифрования

Написать программу, которая должна:

- давать возможность выбора или редактирования алфавита;
- задавать любой ключ шифрования;
- позволять вводить открытый текст и шифротекст;
- шифровать открытый текст и выводить результат;
- дешифровать шифротекст и выводить результат;

Отчет должен содержать:

1. Алгоритм работы программы.

2. Исходный текст и результат шифрования.
3. Шифротекст и результат дешифровки.
4. Текст программы.

!!!Задачу 1 можно выполнить без написания программы. В этом случае задачей является шифрование вручную своего имени и фамилии с использованием пяти шифров: биграммного, модулярного, перестановочного с ключевым словом, шифра Вижинера и шифра Полибия.

Варианты к задаче 1:

0. Биграммный шифр.
1. Модулярный шифр.
2. Перестановочный с ключевым словом.
3. Простой перестановочный шифр.
4. Составной шифр Вижинера.
5. Шифр Бофорта.
6. Шифр Вернама («открытый блокнот»).
7. Шифр Вижинера.
8. Шифр Полибия.
9. Шифр Цезаря.

Задача 2. Шифрование открытым ключом.

Для выполнения задания требуется:

- Сгенерировать открытый и закрытый ключи.
- Зашифровать свою фамилию.
- Записать результат.
- Дешифровать текст и проверить правильность шифра.

Литература

1. Цымбал В.П. Теория информации и кодирования. – Киев: Вища школа, 1992.
2. Цымбал В.П. Задачник по теории информации и кодирования. – Киев: Вища школа, 1976.
3. Колесник В.Д., Полтырев Г.Ш. Курс теории информации – М.: Наука, 1982.
4. Кузьмин И.В., Кедрус В.А. Основы теории информации и кодирования. – Киев: Вища школа, 1977.
5. Хэмминг Б. Теория информации и кодирования. – М.: Радио и связь, 1983
6. Тюрин Ю.Н., Макаров А.А. Анализ данных на компьютере. – М.: ИНФРА-М, 1993.
7. Филипчук Е.В., Пахомов С.В. Теория информации и помехоустойчивое кодирование. – М.: МИФИ, 1989.
8. Ризаев И.С. Теория информации и кодирования. – Казань: КГТУ, 2002

Для заметок

УЧЕБНОЕ ИЗДАНИЕ

МАСКАЕВА Ирина Владимировна

«МАТЕМАТИЧЕСКИЕ МОДЕЛИ ПРОЦЕССОВ ИНФОРМАЦИИ И УПРАВЛЕНИЯ»

Задания

к контрольной работе

В авторской редакции

Подписано в печать 10.04.2005 г.(51) Формат 60x84 1/16. Бумага офсетная. Печать офсетная. Усл.печ.л. 2,0. Уч-изд.л. 1,4. Тираж 35 экз.

Учреждение образования
«Гомельский государственный университет
имени Франциска Скорины»,
246019, г. Гомель, ул. Советская, 104

Отпечатано в учреждении образования
«Гомельский государственный университет
имени Франциска Скорины»
246019, г. Гомель, ул. Советская, 104