Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра автоматизированных систем управления (АСУ)

А.Я. Суханов

Информатика

Учебное методическое пособие по практическим занятиям,
самостоятельной и индивидуальной работе студентов направления
09.03.03 Прикладная информатика в экономике

Томск 2020

Суханов А.Я.

Информатика: Учебное методическое пособие по практическим занятиям, самостоятельной и индивидуальной работе студентов – 122 с.

Учебное методическое пособие содержит программу и задания для проведения и выполнения практических занятий, а так же все необходимые формы документов для выполнения практических заданий.

Введение

Информатика это теоретическая и прикладная научная дисциплина изучающая процессы сбора, обработки, хранения и передачи информации. Мы живем в век, когда информационные процессы играют основополагающую роль в жизни человеческого общества. Трудно себе представить современную жизнь без устройств обрабатывающих, предоставляющих и хранящих в том или ином виде разнообразную информацию, требующуюся человеку как для выполнения его рабочих обязанностей, так и для улучшения его качества жизни.

Данное пособие предназначено для выполнения студентами бакалаврами практических и лабораторных работ по информатике, изучения теоретической информатики, теории кодирования и способов представления информации, а также освоения практических навыков работы с программными продуктами, использующимися для реализации информационных процессов, в частности редактирования текстов и обработки табличных данных.

1. Практическое занятие: Кодирование, представления числовой информации

1.1 Равномерное и неравномерное кодирование

Результат одного отдельного альтернативного выбора может быть представлен как 0 или 1. Тогда выбору всякого сообщения (события, символа т.п.) в массиве сообщений соответствует некоторая последовательность двоичных знаков 0 или 1, то есть двоичное слово. Это двоичное слово называют кодировкой, а множество кодировок источника источника сообщений – кодом сообщений.

Кодирование – замена информационного слова на кодовое.

Пример.

Информационное слово	Кодовое слово
000	0000
001	0011
010	0101
011	0110
100	1001
101	1010
110	1100
111	1111

Если количество символов представляет собой степень двойки ($n = 2^N$) и все знаки равновероятны $P = (1/2)^N$, то все двоичные слова имеют длину L=N=ld(n). Такие коды называют равномерными кодами.

Более оптимальным, с точки зрения объема передаваемой информации, является неравномерное кодирование, когда разным сообщениям в массиве сообщений назначают кодировку разной длины. Причем, часто происходящим событиям желательно назначать кодировку меньшей длины и наоборот, т.е. учитывать их вероятность.

Кодирование словами постоянной длины

Буква	a	b	c	d	e	f	g
Кодирование	000	001	010	011	100	101	110

ld(7)=2,807 и L=3.

1.1.1 Кодирование Шеннона Фано

Проведем кодирование, разбивая исходное множество знаков на равновероятные подмножества, то есть так, чтобы при каждом разбиении суммы вероятностей для знаков одного подмножества и для другого подмножества были одинаковы. Для этого сначала

расположим знаки в порядке уменьшения их вероятностей. Затем будем делить таблицу так, чтобы сумма вероятностей символов в одной части таблицы, была примерно равна сумме вероятностей в другой таблице, назначим одной части таблицы начальный символ 1, другой символ 0, затем продолжим делить получившиеся группы символов, так же назначая им следующие коды 1 и 0.

Итоговый результат приведен ниже, данный код получил название код Шеннона-Фано.

Символ	Вероятность, P_i	Кодировка	Длина, L_i	Вероятность*Длина, P_i*L_i
a	0,25	00	2	0,5
e	0,25	01	2	0,5
f	0,125	100	3	0,375
c	0,125	101	3	0,375
b	0,125	110	3	0,375
d	0,0625	1110	4	0,25
g	0,0625	1111	4	0,25

В общем случае алгоритм построения оптимального кода Шеннона-Фано выглядит следующим образом:

1. сообщения, входящие в ансамбль, располагаются в столбец по мере убывания вероятностей;

2. выбирается основание кода K (в нашем случае K=2); 3. все сообщения ансамбля разбиваются на K групп с

суммарными вероятностями внутри каждой группы как можно близкими друг к другу.

4. всем сообщениям первой группы в качестве первого символа присваивается 0, сообщениям второй группы – символ 1, а сообщениям K-й группы – символ (K–1); тем самым обеспечивается равная вероятность появления всех символов 0, 1,…, K на первой позиции в кодовых словах;

5. каждая из групп делится на K подгрупп с примерно равной суммарной вероятностью в каждой подгруппе. Всем сообщениям первых подгрупп в качестве второго символа присваивается 0, всем сообщениям вторых подгрупп – 1, а сообщениям K-ых подгрупп – символ (K–1).

6. процесс продолжается до тех пор, пока в каждой подгруппе не окажется по одному сообщению.

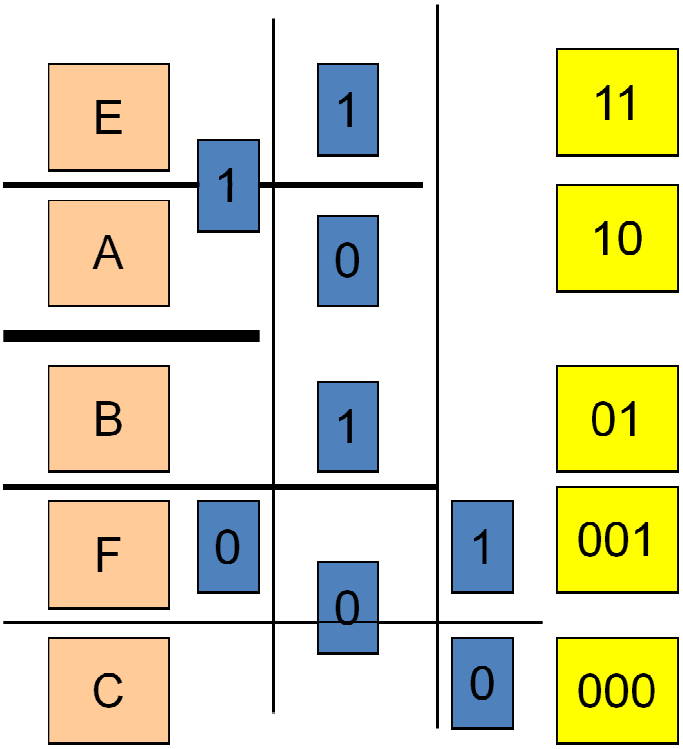
Приведем еще один пример для получения кода Шеннона-Фано. Пусть дана таблица.

Символ	Вероятность, Pi
A	0,2
E	0,4
F	0,125
C	0,125
B	0,15

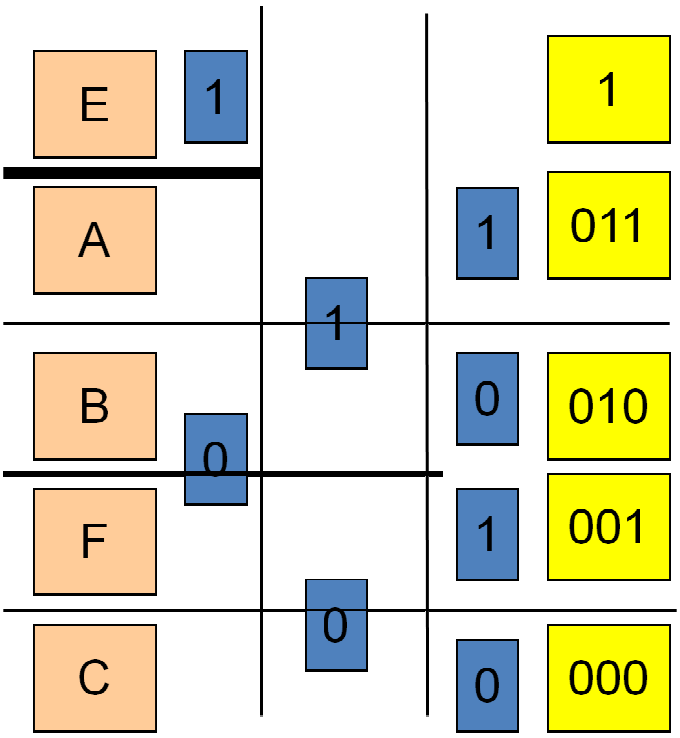
Упорядочим ее по убыванию вероятностей.

Символ	Вероятность, Pi
E	0,4
A	0,2
B	0,15
F	0,125
C	0,125

Приведем пример, где мы разделили сначала таблицу на символы групп E,A и группы (B, F, C), так как вероятности этих групп 0.6 и 0.4.



Либо разбив сначала на группы 0.4 (E), 0.6 (A, B, F, C) получим следующий результат.



Заметим, что построение кода можно представить в виде бинарного дерева, где каждой ветви назначается знак 1 или 0, например, правой 1, а левой 0.

Рассмотрим пример, где возьмем не вероятности, а частоту встречаемости символа (но это, практически, одно и то же, если поделить на общее количество встреченных символов).

A (частота встречаемости 50)

B (частота встречаемости 39)

C (частота встречаемости 18)

D (частота встречаемости 49)

E (частота встречаемости 35)

F (частота встречаемости 24)

Упорядочим:

A (частота встречаемости 50)

D (частота встречаемости 49)

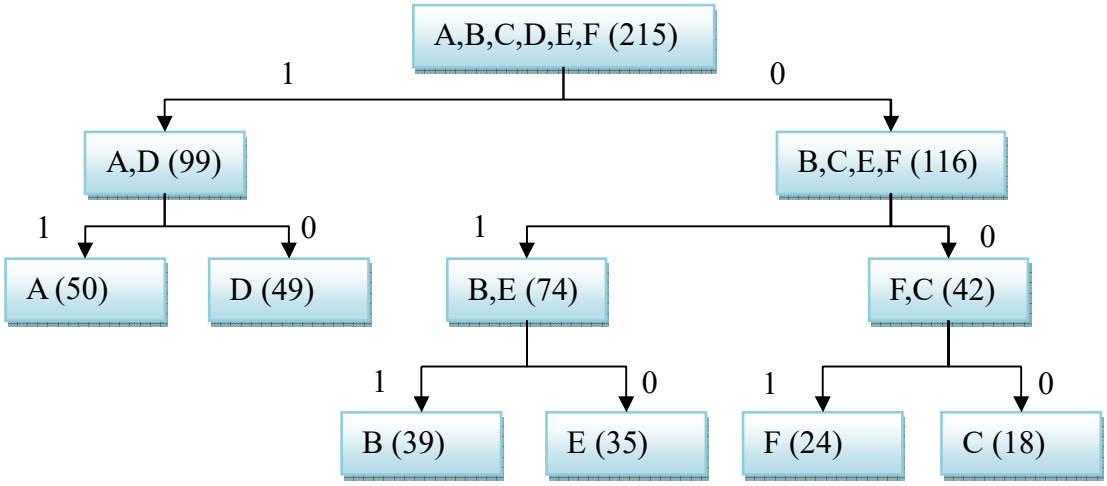
B (частота встречаемости 39)

E (частота встречаемости 35)

F (частота встречаемости 24)

C (частота встречаемости 18)

Получаем следующее дерево:



Проходим по ветвям этого дерева, записывая код для каждой буквы.

1.1.2 Кодирование Хаффмана

Еще один код, относящийся к неравномерному кодированию, это код Хафмана, он используется в процедурах сжатия без потерь.

Классический алгоритм Хаффмана на входе получает таблицу частот встречаемости символов в сообщении. Далее на основании этой таблицы строится дерево кодирования Хаффмана (Н-дерево).

1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.

2. Выбираются два свободных узла дерева с наименьшими весами.

Создается их родитель с весом, равным их суммарному весу.

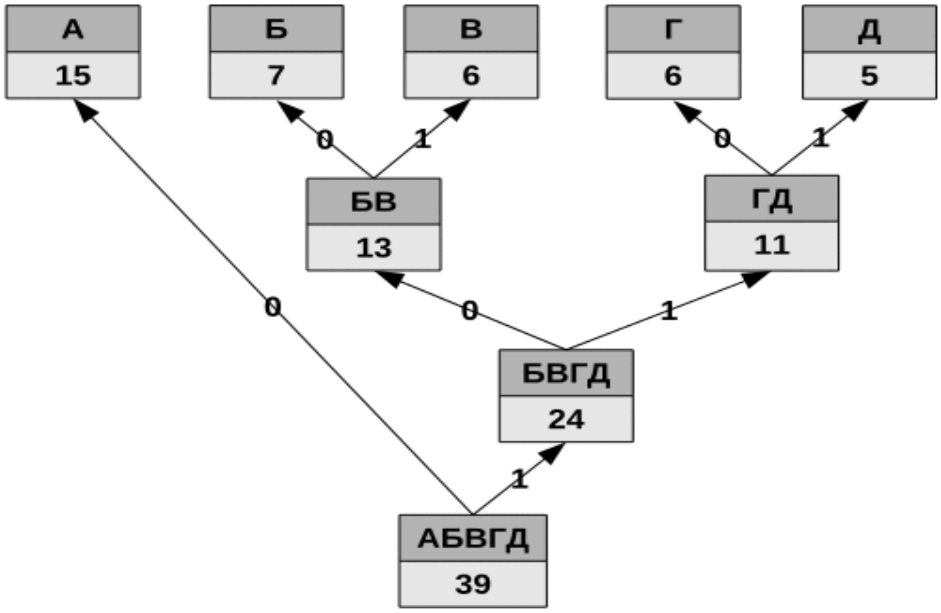
4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.

5. Одной дуге, выходящей из родителя, ставится в соответствие бит 1, другой — бит 0. Битовые значения ветвей, исходящих от корня, не зависят от весов потомков.

Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

Проще это описать следующим образом: берем два наименее встречающиеся символа и объединяем вместе, приписывая им общую вероятность появления, одному символу при этом ставим в соответствие 1, другом ноль, теперь это один общий символ, повторяем все с этим новым символом и остальными.

Пример дерева для кодирования приведен ниже:



Итого:

А	Б	В	Г	Д
0	100	101	110	111

Здесь мы начинаем с Г и Д, далее объединяем Б и В. Потом объединяем (Г,Д) и (Б,В).

При неравномерном кодировании вводят среднюю длину кодировки, которая определяется по формуле

$$L = \sum_{i=1}^{n} P_i L_i$$

В общем же случае связь между средней длиной кодового слова L и энтропией H источника сообщений дает следующая теорема кодирования Шеннона:

имеет место неравенство L>=H, причем L = H тогда, когда набор знаков можно разбить на точно равновероятные подмножества;

всякий источник сообщений можно закодировать так, что разность L – H будет как угодно мала.

Разность L–H называют избыточностью кода (мера бесполезно совершаемых альтернативных выборов).

Символ	Вероятность	Кодировка	Длина	В×Д
A	0,7	0	1	0,7
B	0,2	10	2	0,4
C	0,1	11	2	0,2

Средняя длина слова: $L = 0,7+0,4+0,2=1,3$.

Среднее количество информации, содержащееся в знаке (энтропия):

$H = 0,7\times\text{ld}(1/0,7)+0,2\times\text{ld}(1/0,2)+0,1\times\text{ld}(1/0,1) = 0,7\times0,515+0,2\times2,322+$

$+0,1\times3,322 = =1,1571$.

Избыточность $L - H = 1,3 - 1,1571 = 0,1429$.

Следует *не просто кодировать каждый знак в отдельности*, а рассматривать вместо этого двоичные кодирования для *nk* групп по *k* знаков.

Пары	Вероятность	Кодировка	Длина	В×Д
AA	0,49	0	1	0,49
AB	0,14	100	3	0,42
BA	0,14	101	3	0,42
AC	0,07	1100	4	0,28
CA	0,07	1101	4	0,28
BB	0,04	1110	4	0,16
BC	0,02	11110	5	0,10
CB	0,02	111110	6	0,12
CC	0,01	111111	6	0,06
Средняя длина кодовой группы из 2-х символов равна				2,33

Средняя длина кода одного знака равна 2,33/2=1,165 – уже ближе к энтропии. Избыточность равна $L - H = 1,165 - 1,1571 \approx 0,008$.

Как видно при объединении в группы, избыточно стала уменьшаться и среднее число бит на символ стало ближе к энтропии.

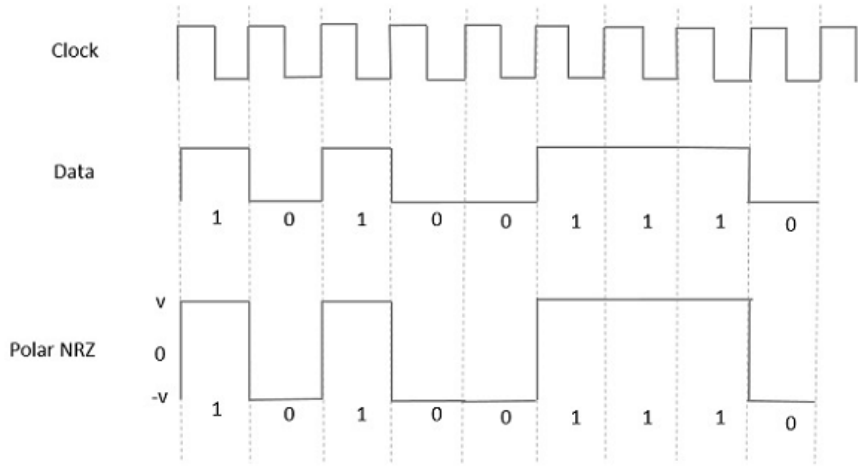
1.2 Кодирование в условиях помех

При передаче данных по каналам связи, происходит их искажение за счет наличия различных шумов, воздействующих на передающийся сигнал. Часто возникает так называемая аддитивная помеха, которая изменяет уровень сигнала, что приводит к его неправильной интерпретации в процессе демодуляции.

Используют разные способы физического кодирования, потенциальное кодирование, импульсное кодирование, выделяют методы манипуляции и модуляции, которые используют ту или иную физическую характеристику сигнала, изменяющуюся за время такта (фазу, частоту и амплитуду). Понятие манипуляция относится к цифровому сигналу, а модуляция к аналоговому.

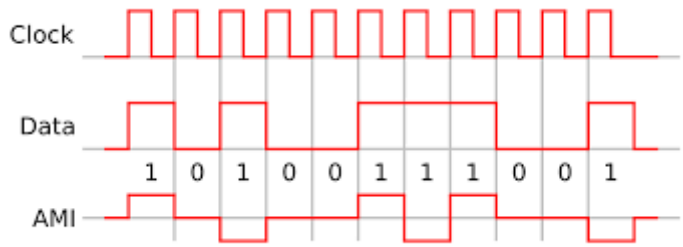
Приведем пример потенциального кода и на его примере рассмотрим основные два вида ошибок, которые возникают при передаче. Первый тип это стирание или вставка, второй тип это замена.

Потенциальный код предполагает передачу в течение такта положительного уровня для передачи 1 и нулевого уровня или отрицательного уровня для передачи 0. Рассмотрим код 101001110.



Предположим, что мы передаем данные на удаленное расстояние, тогда мы будем вынуждены прокладывать еще дополнительный тактовый канал (Clock), тратя на это провода или дополнительные ресурсы, тем более не факт что на больших расстояниях у нас не произойдет рассинхронизация. Допустим, мы попытаемся передавать информацию по одному каналу связи, используя тактовые генераторы на приемнике и передатчике, при этом пусть мы передаем много единиц или много нулей по каналу связи. Тогда на приемнике мы получаем один и тот же уровень сигнала без каких либо различий, при этом при рассинохранизации тактового генератора мы получим или лишнюю единицу, или не дополучим и пропустим ее, это и будет ошибкой вставки или стирания. Здесь мы видим, что наш код плохо самосинхронизован, потому что на приемнике не видно прихода нового символа, не происходит смена физического состояния, которое мы могли бы отличить. Потому все использующиеся физические коды самосинронизированы, например, биполярный импульсный код, код AMI, код MLT-3, манчестерский код, в них всегда есть смена состояния в такте, которая может быть различена приемником. Здесь представлен код AMI, который синхронизован по единицам, каждая новая 1 кодируется новой полярностью. Более того, так еще можно и отслеживать ошибки, если, например, за положительной полярностью следует опять положительная, а не отрицательная. Для улучшения синхронизации по нулям используются специальные коды, вводящие избыточность и запрещающие использование кодов, где, например, присутствует много

нулей подряд, например, b8zs, hdb3. Использующие коды где запрещено использование множества подряд нулей или единиц, например, 4b5b.



За счет чего происходят искажения вида замена? За счет различных физических эффектов: затухание сигнала в канале связи, поглощение или рассеяние, возможно, что эти эффекты сильнее на каких-то длинах волн, что приводит к сглаживанию сигнала, и, например, можно запутать единицу с нулем. Так же возможны помехи, которые приводят к тому, что на приемнике появляется усиленный или ослабленный сигнал, или вовсе изменивший свое состояние.

Для того, чтобы бороться с помехами типа замена используется избыточное кодирование, дополняющее информационные биты или символы дополнительными проверочными, здесь выделяют так называемое блочное кодирование и сверточное кодирование.

Прежде чем рассматривать данные методы, рассмотрим, что такое расстояние Хэмминга. Расстояние Хэмминга между двумя словами есть число разрядов, в которых эти слова различаются.

Примеры:

1. Расстояние Хэмминга d(000, 011) есть 2.
2. Расстояние Хэмминга d(10101, 11110) равно 3

В процессе декодирования происходит исправления ошибок, если они произошли. Рассмотрим пример:

Пусть есть множество кодовых слов {00000,01101,10110,11011}

Если полученное слово 10000, то декодируем в «ближайшее» слово 00000.

Если полученное слово 11000 – то только обнаружение ошибки, так как есть два варианта с минимальными до данного слова расстояниями хэмминга: 11000 – в 00000 или 11000 – в 11011.

Коды, в которых возможно автоматическое исправление ошибок, называются самокорректирующимися. Для построения самокорректирующегося кода, рассчитанного на исправление одиночных ошибок, одного контрольного разряда недостаточно.

Количество контрольных разрядов k должно быть выбрано так, чтобы удовлетворялось неравенство:

$$2^k \geq k + m + 1$$

где m количество разрядов кодируемого слова.

Минимальные значения k при заданных значениях m

Диапазон m	k_{min}
1	2
2-4	3
5-11	4
12-26	5
27-57	6

1.3 Кодирование числовых данных

Известно, что число с дробной частью в позиционной системе счисления можно представить следующим образом:

$$p = \sum_{i=-m}^{n} a_i p^i$$

где a_i – цифра числа в позиции, p – основание системы счисления.

Таким образом, позиция цифры определяет вес цифры в числе, определяемый основанием системы счисления в степени номера позиции. Отрицательные номера определяют дробную часть числа, положительные и нулевая степень целую часть числа.

Один из способов перевода числа из какой-либо системы в десятичную заключается в разложение на соответствующие степени и выполнения операций в десятичной системе для получения числа в десятичной системе, так же можно перевести из десятичной системы в любую другую, реализовав разложение, начиная с большей степени, так чтобы в итоге получить исходной число.

Например, переведем из десятеричной системы счисления число 25_{10} в двоичную систему счисления. Первая степень, которая входит в данное число это 16, остается 9, потом входит 8, и затем 1. Степени, которые входят, помечаем как 1, те которые не входят в число, помечаем как 0.

2^4	2^3	2^2	2^1	2^0
16	8	4	2	1

1	1	0	0	1

Можем проверить, сложив соответствующие степени, где стоит 1 и получим 25.

Обратно из двоичной перевести так же просто, допустим число 10111_2.

$1*2^4+0*2^3+1*2^2+1*2^1+1*2^0=23_{10.}$

Так же можно переводить из одной системы счисления в десятичную операцией делением, делим на основание системы счисления, получаем остаток от деления и записываем его в качестве младшей цифры числа, затем результат от целого деления продолжаем делить на основание системы счисления, пока не получим 0.