Информация и алфавит

- Сообщение есть последовательность знаков алфавита. При их передаче возникает проблема распознавания знака: каким образом прочитать сообщение, т.е. по полученным сигналам установить исходную последовательность знаков первичного алфавита.

Появление конкретного знака (буквы) в конкретном месте сообщения – событие случайное. Следовательно узнавание знака требует получения некоторой порции информации.

Предположим, что появление всех знаков (букв) алфавита в сообщении равновероятно.

$n_e = 27$ – для английского алфавита

$n_r = 34$ – для русского алфавита

- Из формулы Хартли:

$Ie_0 = \log_2 27 = 4{,}755$ бит

$Ir_0 = \log_2 34 = 5{,}087$ бит

Получается, что со знаком русского алфавита, в среднем связано больше информации.

Однако, относительная частота, т.е. вероятность появления различных букв в тексте (или сообщении) различна.

Таблица средних частот букв для русского алфавита (следующий слайд).

$$I = -\sum_{i=1}^{n} p_i \times \log_2 p_i - \text{формула Шеннона (1948 г.)}$$

Таблица для средних частот букв русского алфавита

Буква	**пробел**	**о**	**е, ё**	**а**	**и**	**т**	**н**	**с**
Относительная частота	0,175	0,090	0,072	0,062	0,062	0,053	0,053	0,045
Буква	**р**	**в**	**л**	**к**	**м**	**д**	**п**	**у**
Относительная частота	0,040	0,038	0,035	0,028	0,026	0,025	0,023	0,021
Буква	**я**	**ы**	**з**	**ь, ъ**	**б**	**г**	**ч**	**й**
Относительная частота	0,018	0,016	0,016	0,014	0,014	0,013	0,012	0,010
Буква	**х**	**ж**	**ю**	**ш**	**ц**	**щ**	**э**	**ф**
Относительная частота	0,009	0,007	0,006	0,006	0,004	0,003	0,003	0,002

Вероятность появления p и значения $-\log_{10} p$ для букв англнйского языка

Символ	Вероятность p	$-\log_{10} p$	Символ	Вероятность p	$-\log_{10} p$
Промежуток	0,2	0,699	*L*	0,029	1,54
E	0,105	0,979	*C*	0,023	1,64
T	0,072	1,143	*FU*	0,0225	1,65
O	0,0654	1,184	*M*	0,021	1,68
A	0,063	1,2	*P*	0,0175	1,76
N	0,059	1,23	*YW*	0,012	1,92
I	0,055	1,26	*G*	0,011	1,96
R	0,054	1,27	*B*	0,0105	1,98
S	0,052	1,28	*V*	0,008	2,1
H	0,047	1,33	*K*	0,003	2,52
D	0,035	1,46	*X*	0,002	2,7
			JQZ	0,001	3,0

Сообщения, в которых вероятность появления каждого отдельного знака не меняется со временем, называют шенноновскими, а порождающий его отправитель – шенноновским источником.

Если сообщение является шенноновским, то набор знаков и связанная с каждым знаком информация известны заранее.

В этом случае интерпретация сообщения, представляющего собой последовательность сигналов, сводится к задаче распознавания знака, т.е. выявлению какой именно знак находится в данном месте сообщения.

- Применение формулы Шеннона к алфавиту русского языка даёт:

$Ir_1 = 4{,}36$ бит
$Ie_1 = 4{,}04$ бит
$If_1 = 3{,}96$ бит
$Id_1 = 4{,}10$ бит
$Is_1 = 3{,}98$ бит

Следующим приближением при оценке значения информации, приходящейся на знак алфавита, должен быть учёт корреляций, т.е. связей между буквами в словах («пр..», «шц», «фъ»).

$Ie_2 = 3{,}32$ бит
$Ie_3 = 3{,}10$ бит
$Ie_5 \approx 2{,}1$ бит
$Ie_8 \approx 1{,}9$ бит
$Ir_2 \approx 3{,}52$ бит
$Ir_3 \approx 3{,}01$ бит

Шеннон ввёл величину, которую назвал относительной избыточностью языка:

$$R = 1 - \frac{I_\infty}{I_0}$$

Понятие о кодировании. Коды. Кодирование символьной информации

Теория кодирования информации является одним из разделов теоретической информатики. К основным задачам, решаемым в данном разделе, необходимо отнести следующие:

- разработка принципов наиболее экономичного кодирования информации;
- согласование параметров передаваемой информации с особенностями канала связи;
- разработка приемов, обеспечивающих надежность передачи информации по каналам связи, т.е. отсутствие потерь информации.

В зависимости от целей кодирования различают следующие его виды:

1. **Кодирование по образцу** – используется всякий раз при вводе информации в компьютер для её внутреннего представления;
2. **Криптографическое кодирование** – используется при необходимости защиты информации от несанкционированного доступа;
3. **Эффективное (оптимальное кодирование)** – используется для устранения избыточности информации, т.е. для снижения её объема;
4. **Помехозащитное (помехоустойчивое)** кодирование – используется для обеспечения заданной достоверности в случае, когда на сигнал накладывается помеха (например, при передаче информации по каналам связи)

В целом кодирование представляет собой отображение:

$C: A^* \rightarrow B^*$

- Функция С должна быть достаточно простой
- Отображение С должно быть обратимым.
- При шифровании наоборот С должно быть достаточно сложным

Определения:

- ***Код*** – (1) правило, описывающее соответствие знаков или их сочетаний одного алфавита знакам или их сочетаниям другого алфавита (С); - (2) знаки вторичного алфавита, используемые для представления знаков или их сочетаний первичного алфавита.
- ***Кодирование*** – перевод информации, представленной посредством первичного алфавита, в последовательность кодов (кодер).
- ***Декодирование*** – операция, ***обратная*** кодированию, т.е. восстановление информации в первичном алфавите по полученной последовательности кодов (декодер).

Операции кодирования и декодирования называются обратимыми, если их последовательное применение обеспечивает возврат к исходной информации без каких-либо ее потерь.

Наиболее удобным способом задания кода является табличный способ. Например: пусть имеется алфавит с конечным числом букв: A={a1, a2, a3, a4}. Тогда код для А:

a1	C(a1)
a2	C(a2)
a3	C(a3)
a4	C(a4)

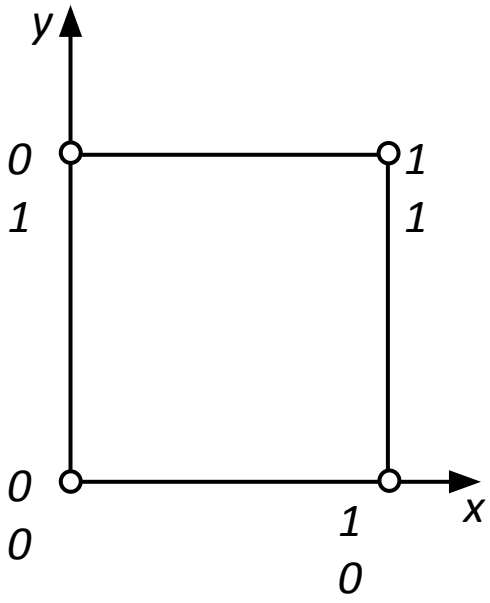
Другим примером кодирования может служить двоичное представление чисел:

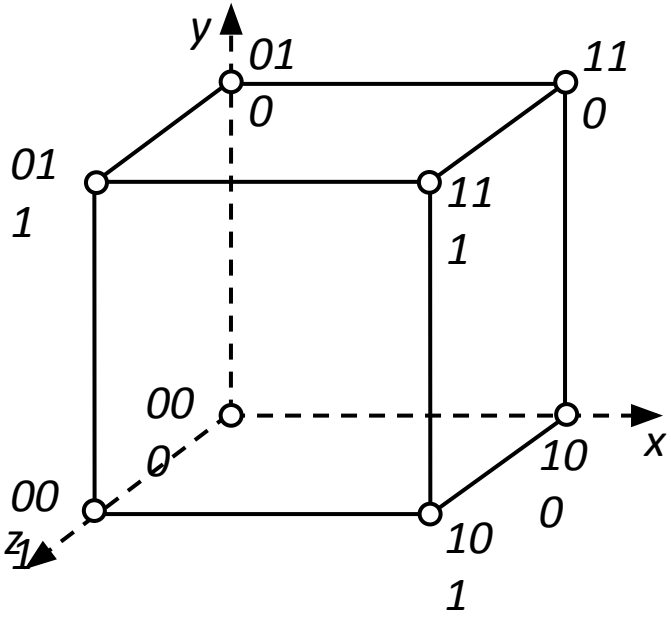
Десятичное число	***Равномерный код***	***Неравномерный код***
0	000	0
1	001	1
2	010	10
3	011	11
4	100	100
5	101	101
6	110	110
7	111	111

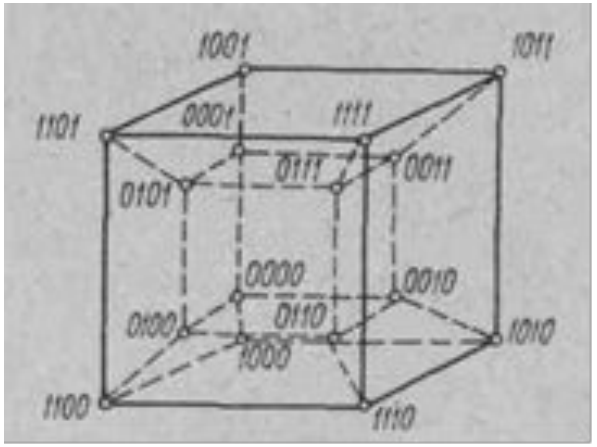
Представление кода в виде геометрической модели:

Представление кода в виде геометрической модели возможно благодаря тому, что кодовые комбинации n-значного кода могут рассматриваться как определенные точки n-мерного пространства

Представление кодов в виде геометрической модели производят для наглядности изображения и облегчения анализа их свойств и даже используют при построении корректирующих кодов.



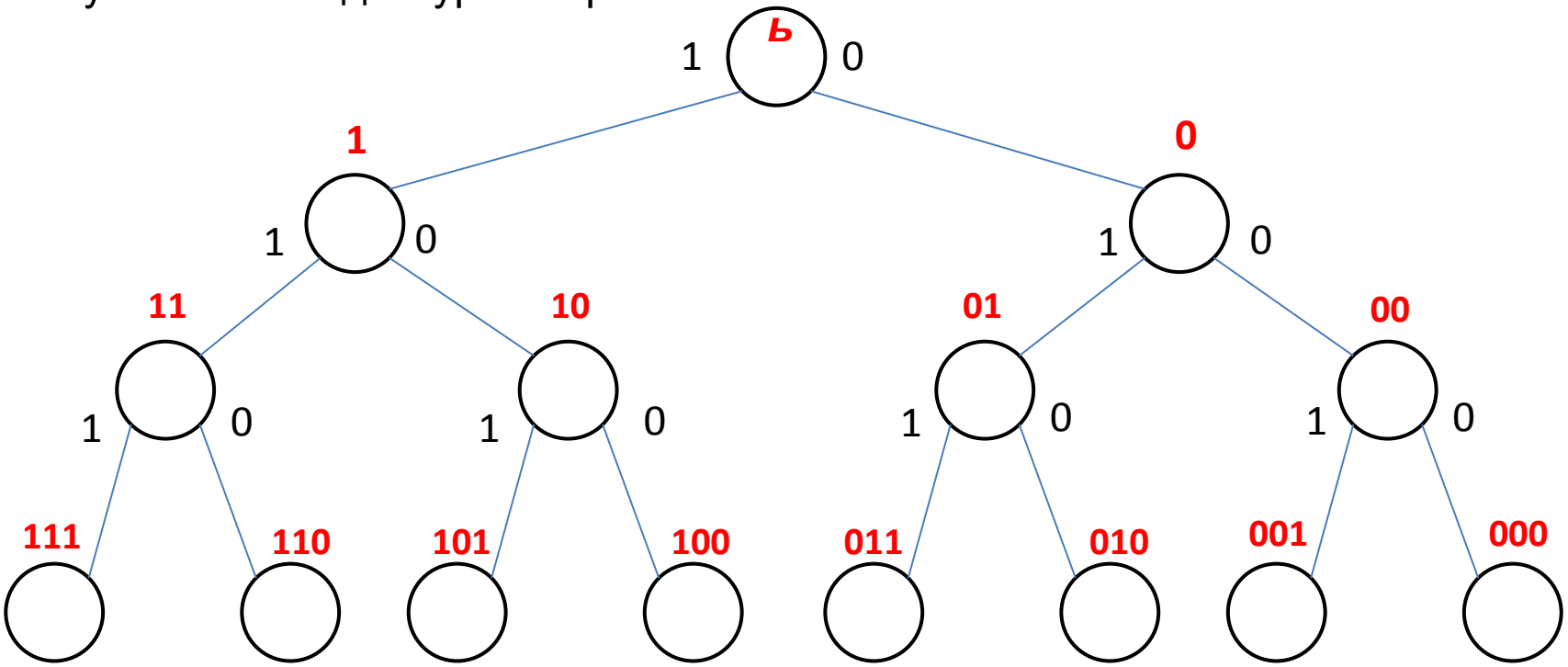




Кодовые деревья

Наглядным способом описания кодов являются так называемые *кодовые деревья.* Представление кода в виде кодового дерева — давно известный и широко распространенный в теории релейно контактных схем способ.

В общем виде кодовое дерево может быть представлено как граф, состоящий из узлов и ветвей, соединяющих узлы, расположенные на разных уровнях. Истоком графа является корень. Каждый уровень содержит m^n узлов, где **n** — номер уровня, а **m** — значность кода. Для равномерного двоичного кода число узлов на каждом уровне равно 2ⁿ.



Кодовое дерево двоичного

Если число узлов на каждом уровне кодового дерева равно m^n, то дерево называют *полным.* Величина $D = m^k$, где k указывает порядок наивысшего уровня кодового дерева, называется ***объемом дерева***. Объем дерева характеризует число кодовых комбинаций, которое может быть построено при помощи данного дерева.

При помощи кодовых деревьев наглядно представляются коды, обладающие *свойством префикса,* или *префиксные* коды, т. е. коды, которые могут быть получены путем последовательного вычеркивания последнего знака кодовой комбинации (ни одна из комбинаций такого кода не может быть префиксом комбинации того же кода). Например, префиксами кодовой комбинации 11101101 будут: 1, 11, 111,1110, 111011, 1110110, 11101101.

То есть для однозначного декодирования кодовой комбинации 11101101 ни один из передаваемых кодов не может быть представлен комбинациями 1, 11, 111, 1110, 111011, 1110110, 11101101.

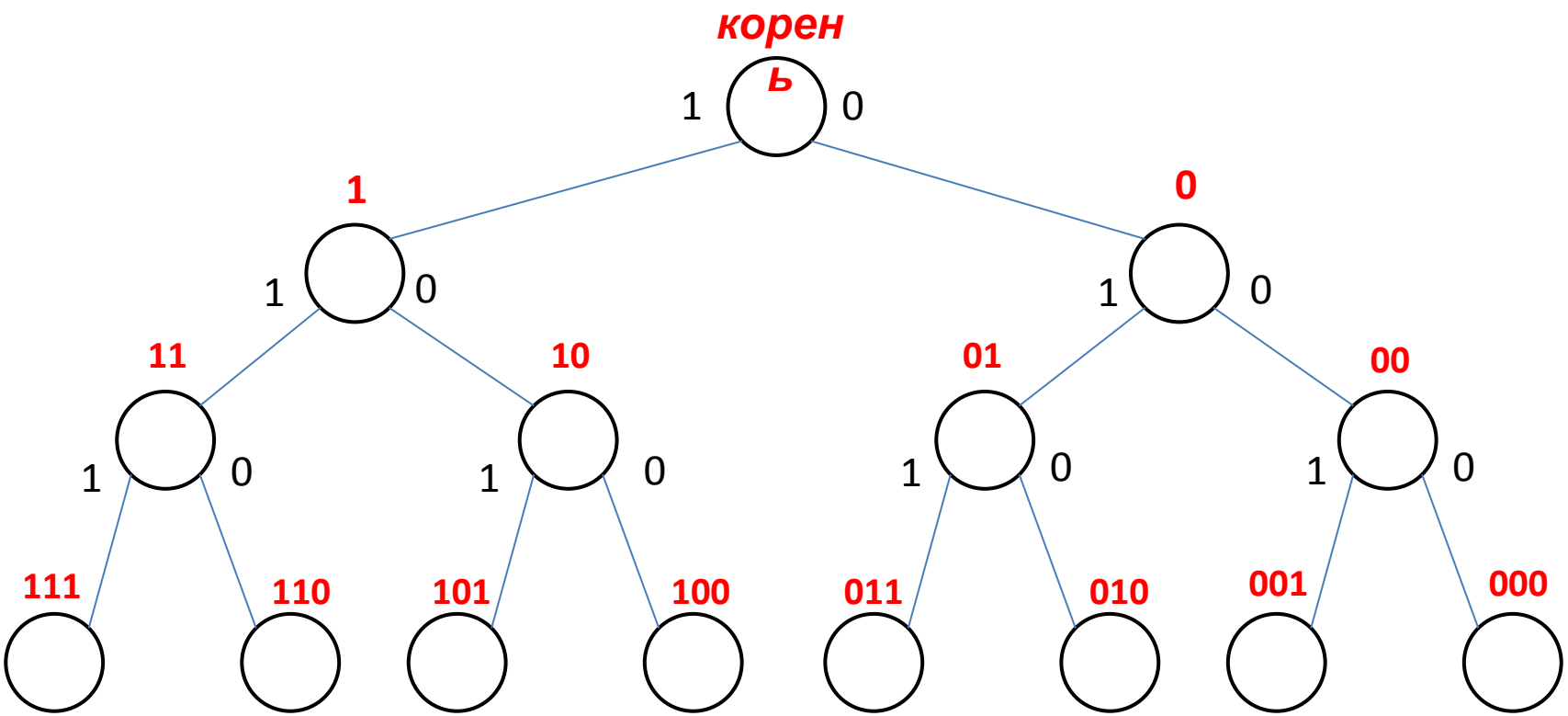
Префиксом данной кодовой комбинации A_i является любая последовательность, составленная из ее начальной части, включая саму комбинацию A_i

Та часть кодовой комбинации, которая дополняет префикс до самой кодовой комбинации, называется *суффиксом,* т. е. каждую кодовую комбинацию можно разбить на префикс и соответствующие суффиксы.

Имея кодовое дерево, легко определить, обладает ли данный код свойством префикса. Если ни один узел кодового дерева не является вершиной данного кода, то он обладает свойством префикса. Для заданного кода кодовое дерево всегда одно и то же.

Узлы, которые не соединяются с последующими уровнями, называются *оконечными.* Комбинации кода, соответствующие оконечным узлам, являются комбинациями кода, обладающего свойством префикса. Если добавление к комбинациям заданного кода некоторой новой комбинации ведет к нарушению свойства префикса, то такой код называется *полным.*

Свойство префикса широко используют при построении неравномерных кодов с минимальной длиной кодовых слов, в частности кодовых деревьев по методу Хаффмена



Кодовое дерево двоичного кода

Математическая постановка задачи кодирования

Пусть первичный алфавит **A** содержит **N** знаков со средней информацией на знак, определенной с учетом вероятностей их появления, $\mathbf{I}^{(A)}$. Вторичный алфавит **B** – **M** знаков со средней информационной емкостью $\mathbf{I}^{(B)}$. Пусть также исходное сообщение, представленное в первичном алфавите, содержит **n** знаков, а закодированное сообщение – **m** знаков. Если исходное сообщение содержит $\mathbf{I}_{st}^{(A)}$ информации, а закодированное – $\mathbf{I}_{fin}^{(B)}$, то условие обратимости кодирования, т.е. неисчезновения информации при кодировании, очевидно, может быть записано следующим образом:

$$\mathbf{I}_{st}^{(A)} \leq \mathbf{I}_{fin}^{(B)}$$

$$K\ \ (A,B) \geq \frac{I^{\ (A)}}{I^{\ (B)}} \qquad (1)$$

$$K^{min}(A,B) = \frac{I^{\ (A)}}{I^{\ (B)}} \qquad (2)$$

Первая теорема Шеннона о передаче информации, которая называется также основной теоремой о кодировании при отсутствии помех, формулируется следующим образом:

При отсутствии помех передачи всегда возможен такой вариант кодирования сообщения, при котором среднее число знаков кода, приходящихся на один знак кодируемого алфавита, будет сколь угодно близко к отношению средних информаций на знак первичного и вторичного алфавитов.

$$K^{min}(A,B) = \frac{I_1{}^{(A)}}{\log_2 M} \qquad (3)$$

$$Q(A,B) = \frac{K(A,B) - K^{min}(A,B)}{K^{min}(A,B)} = \frac{K(A,B)}{K^{min}(A,B)} - 1 = \frac{K(A,B) * I^{(B)}}{I^{(A)}} - 1 \qquad (4)$$

Данная величина показывает, насколько операция кодирования увеличила длину исходного сообщения.

Используя понятие избыточности кода, можно дать более короткую формулировку теоремы:

При отсутствии помех передачи всегда возможен такой вариант кодирования сообщения, при котором избыточность кода будет сколь угодно близкой к нулю.

Определение количества переданной информации при двоичном кодировании сводится к простому подсчету числа импульсов (единиц) и пауз (нулей).

M=2 $K^{min}(A,2) = I_1{}^{(A)}$

При отсутствии помех средняя длина двоичного кода может быть сколь угодно близкой к средней информации, приходящейся на знак первичного алфавита.

Применение формулы (4) для двоичных сообщений источника без памяти при кодировании знаками равной вероятности даёт:

$$Q(A,2) = \frac{K(A,2)}{I_1{}^{(A)}} - 1 \qquad (5)$$

При декодировании двоичных сообщений возникает проблема выделения из потока сигналов (последовательности импульсов и пауз) отдельных кодов, соответствующим отдельным знакам первичного алфавита. При этом приемное устройство фиксирует интенсивность и длительность сигналов.

Возможны следующие особенности вторичного алфавита:

- Элементарные сигналы (0 и 1) могут иметь **одинаковые** или **разные** длительности.
- Их количество в коде (**длина кодовой цепочки**), который ставится в соответствие знаку первичного алфавита, также может быть **одинаковым** (в этом случае код называется **равномерным**) или **разным** (**неравномерный код**).
- Коды могут строиться для каждого знака исходного алфавита (**алфавитное кодирование**) или для их комбинаций (**кодирование блоков, слов**).

Длительности элементарных сигналов	Кодировка первичных символов (слов)	Ситуация
одинаковые	равномерная	(1)
одинаковые	неравномерная	(2)
разные	равномерная	(3)
разные	неравномерная	(4)

В случае использования неравномерного кодирования или сигналов разной длительности (ситуации (2), (3) и (4)) для отделения кода одного знака от другого между ними необходимо передавать специальный сигнал – **временной разделитель** (**признак конца знака**) или применять такие коды, которые оказываются уникальными, т.е. несовпадающими с частями других кодов. При равномерном кодировании одинаковыми по длительности сигналами (ситуация (1)) передачи специального разделителя не требуется, поскольку отделение одного кода от другого производится по **общей длительности**, которая для всех кодов оказывается одинаковой (или одинаковому числу бит при хранении).

Алфавитное неравномерное двоичное кодирование сигналами равной длительности

Для передачи информации в среднем приходящейся на знак первичного алфавита необходимо время $K(A,2)*\tau$

построить такую систему кодирования, чтобы суммарная длительность кодов при передаче (или суммарное число кодов при хранении) данного сообщения была бы наименьшей

0010001000011101010111000011 0

- Использовать специальную комбинацию элементарных сигналов, которая интерпретируется декодером как ***разделитель знаков***
- *Применение* ***префиксных кодов***

Неравномерный код с разделителем

00 – признак конца знака

000 – признак конца слова

- код признака конца знака может быть включен в код буквы, поскольку не существует отдельно (т.е. кода всех букв будут заканчиваться 00);
- коды букв не должны содержать двух и более нулей подряд в середине (иначе они будут восприниматься как конец знака);
- код буквы (кроме пробела) всегда должен начинаться с 1;
- разделителю слов (000) всегда предшествует признак конца знака; при этом реализуется последовательность 00000 (т.е. если в конце кода встречается комбинация …000 или …0000, они не воспринимаются как разделитель слов); следовательно, коды букв могут оканчиваться на 0 или 00 (до признака конца знака).

Буква	Код	P_i*10^3	k_i	Буква	Код	P_i*10^3	k_i
пробел	000	174	3	**я**	1011000	18	7
о	100	90	3	**ы**	1011100	16	7
е	1000	72	4	**з**	1101000	16	7
а	1100	62	4	**ь, ъ**	1101100	14	7
и	10000	62	5	**б**	1110000	14	7
т	10100	53	5	**г**	1110100	13	7
н	11000	53	5	**ч**	1111000	12	7
с	11100	45	5	**й**	1111100	10	7
р	101000	40	6	**х**	10101000	9	8
в	101100	38	6	**ж**	10101100	7	8
л	110000	35	6	**ю**	10110000	6	8
к	110100	28	6	**ш**	10110100	6	8
м	111000	26	6	**ц**	10111000	4	8
д	111100	25	6	**щ**	101111100	3	8
п	1010000	23	7	**э**	11010000	3	8
у	1010100	21	7	**ф**	11010100	2	8

$$K^{(2)} = \sum_{i=1}^{32} p_i \cdot k_i = 4{,}964$$

Поскольку для русского языка, $\mathbf{I_1^{(r)}=4{,}356}$ бит, избыточность данного кода, согласно, составляет:

$\mathbf{Q^{(r)} = 4{,}356/4{,}964 - 1 \approx 0{,}122}$

Это означает, что при данном способе кодирования будет передаваться приблизительно на 12% больше информации, чем содержит исходное сообщение. Аналогичные вычисления для английского языка дают значение **K(e, 2) = 4,716**, что при $\mathbf{I_1^{(e)} = 4{,}036}$ бит приводят к избыточности кода **Q(e,2)= 0,168**.

Оптимальное кодирование. Префиксные коды

Оптимальным кодированием называется процедура преобразования символов первичного алфавита m_i в кодовые слова во вторичном алфавите m_1, при которой средняя длина сообщений во вторичном алфавите имеет минимально возможную для данного m_2 длину.

Оптимальными называются коды, представляющие кодируемые понятия кодовыми словами минимальной средней длины.

В сообщениях, составленных из кодовых слов оптимального кода, статистическая избыточность сведена к минимуму, в идеальном случае — к нулю.

Основные свойства оптимальных кодов:

1. минимальная средняя длина кодового слова оптимального кода обеспечивается в том случае, когда избыточность каждого кодового слова сведена к минимуму (в идеальном случае — к нулю);
2. кодовые слова оптимального кода должны строиться из равновероятных и взаимонезависимых символов.

Префиксные коды

Условие Фано: ***Неравномерный код может быть однозначно декодирован, если никакой из кодов не совпадает с началом (префиксом) какого-либо иного более длинного кода.***

Например, если имеется код 110, то уже не могут использоваться коды 1, 11, 1101, 110101 и пр. Если условие Фано выполняется, то при прочтении (расшифровке) закодированного сообщения путем сопоставления со списком кодов всегда можно точно указать, где заканчивается один код и начинается другой.

Пример:

а	л	м	р	у	ы
10	010	00	11	0110	0111

00100010000111010101110000110

- Отрезать от текущего сообщения крайний левый символ, присоединить к рабочему кодовому слову.
- Сравнить рабочее кодовое слово с кодовой таблицей; если совпадения нет, перейти к (1).
- Декодировать рабочее кодовое слово, очистить его.
- Проверить, имеются ли еще знаки в сообщении; если "да", перейти к (1).

Применение данного алгоритма даёт:

Шаг	Рабочее слово	Текущее сообщение	Распознанный знак	Декодированное сообщение
0	пусто	**0**0100010000111010101110000110	–	–
1	0	**0**100010000111010101110000110	нет	–
2	**00**	**1**00010000111010101110000110	**м**	м
3	1	**0**0010000111010101110000110	нет	м
4	**10**	**0**010000111010101110000110	**а**	ма
5	0	**0**100001110101011100000110	нет	ма
6	**00**	**1**00001110101011100000110	**м**	мам
…				

Построение оптимального кода по методу Шеннона — Фано для сообщений сводится к следующей процедуре:

1. множество из *M* сообщений располагают в порядке убывания вероятностей;
2. первоначальный ансамбль кодируемых сигналов разбивают на две группы таким образом, чтобы суммарные вероятности сообщений обеих групп были по возможности равны;
3. первой группе присваивают символ 0, второй группе — символ 1;
4. каждую из подгрупп делят на две группы так, чтобы их суммарные вероятности были по возможности равны;
5. первым подгруппам каждой из групп вновь присваивают О, а вторым — 1, в результате чего получают вторые цифры кода. Затем каждую из четырех подгрупп вновь делят на равные (с точки зрения суммарной вероятности) части и т. д. до тех пор, пока в каждой из подгрупп останется по одной букве.

Префиксный код Шеннона-Фано (1948-1949)

a1	a2	a3	a4	a5	a6
0,3	0,2	0,2	0,15	0,1	0,05

знак	Pi	Разряды кода				Код
		1	2	3	4	
A1	0,3	0	0			00
A2	0,2	0	1			01
A3	0,2	1	0			10
A4	0,15	1	1	0		110
A5	0,1	1	1	1	0	1110
A6	0,05	1	1	1	1	1111

K(A,2) = 0,3*2+ 0,2*2+ 0,2*2 +0,15*3+0,1*4+0,05*4=2,45

$I_1^{(A)}$=2,390 бит

Избыточность кода Q(A,2) = 0,0249, т.е. около 2,5%

Префиксный код Хаффмана

Пример тот же.

Алгоритм:

1. Создадим новый вспомогательный алфавит $\mathbf{A_1}$, объединив два знака с наименьшими вероятностями **(a_5 и a_6)** и заменив их одним знаком (например, $a^{(1)}$);

2. вероятность нового знака будет равна сумме вероятностей тех, что в него вошли, т.е. 0,15;

3. остальные знаки исходного алфавита включим в новый без изменений; общее число знаков в новом алфавите, очевидно, будет на 1 меньше, чем в исходном.

4. Аналогичным образом продолжим создавать новые алфавиты, пока в последнем не останется два знака; ясно, что число таких шагов будет равно **N** – 2, где **N** – число знаков исходного алфавита (в нашем случае **N = 6**, следовательно, необходимо построить 4 вспомогательных алфавита). В промежуточных алфавитах каждый раз будем переупорядочивать знаки по убыванию вероятностей.

Прямой ход:

№ знака	Вероятности				
	Исходный алфавит	Промежуточные алфавиты			
		$A^{(1)}$	$A^{(2)}$	$A^{(3)}$	$A^{(4)}$
1	0,3	0,3	0,3	0,4	0,6
2	0,2	0,2	0,3	0,3	0,4
3	0,2	0,2	0,2	0,3	
4	0,15	0,15	0,2		
5	0,1	0,15			
6	0,05				

Обратный ход:

№ знака	Вероятности									
	Исходный алфавит		Промежуточные алфавиты							
			$A^{(1)}$		$A^{(2)}$		$A^{(3)}$		$A^{(4)}$	
1	0,3	**00**	0,3	**00**	0,3	**00**	0,4	**1**	0,6	**0**
2	0,2	**10**	0,2	**10**	0,3	**01**	0,3	**00**	0,4	**1**
3	0,2	**11**	0,2	**11**	0,2	**10**	0,3	**01**		
4	0,15	**010**	0,15	**010**	0,2	**11**				
5	0,1	**0110**	0,15	**011**						
6	0,05	**0111**								

Средняя длина кода оказывается равной $\mathbf{K^{(2)} = 4{,}395}$; избыточность кода $\mathbf{Q^{(r)} = 0{,}00887}$, т.е. менее 1%.

Таким образом, можно заключить, что существует метод построения оптимального неравномерного алфавитного кода.

Метод Хаффмана и его модификация – метод адаптивного кодирования (динамическое кодирование Хаффмана) – нашли широчайшее применение в программах-архиваторах, программах резервного копирования файлов и дисков, в системах сжатия информации в модемах и факсах

Равномерное алфавитное двоичное кодирование. Байтовый код

В этом случае двоичный код первичного алфавита строится цепочками равной длины, т.е. со всеми знаками связано одинаковое количество информации равное I_0.

Передавать признак конца знака не требуется, поэтому для определения длины кодовой цепочки можно воспользоваться формулой: $K(2) = \log_2 N$.

Приемное устройство просто отсчитывает оговоренное заранее количество элементарных сигналов и интерпретирует цепочку (устанавливает, какому знаку она соответствует).

Правда, при этом недопустимы сбои, например, пропуск одного элементарного сигнала приведет к сдвигу всей кодовой последовательности и неправильной ее интерпретации; решается проблема путем синхронизации передачи или иными способами.

С другой стороны, применение равномерного кода оказывается одним из средств контроля правильности передачи, поскольку факт поступления лишнего элементарного сигнала или, наоборот, поступление неполного кода сразу интерпретируется как ошибка.

Телеграфный код Бодо

Примером равномерного алфавитного кодирования является телеграфный код Бодо, пришедший на смену азбуке Морзе.

Замечания:

1. Исходный алфавит должен содержать не более 32-х символов;
2. $I = \log_2 32 = 5$, т.е. каждый знак содержит 5 бит информации.
3. Условие $N = 32$, очевидно, выполняется для языков, основанных на латинском алфавите ($N = 27 = 26$+"пробел"), однако в русском алфавите 34 буквы (с пробелом) – именно по этой причине пришлось "сжать" алфавит (как в коде Хаффмана) и объединить в один знак "е" и "ё", а также "ь" и "ъ".
4. После такого сжатия $N = 32$, однако, не остается свободных кодов для знаков препинания, поэтому в телеграммах они отсутствуют или заменяются буквенными аббревиатурами; это не является заметным ограничением, поскольку, как указывалось выше, избыточность языка позволяет легко восстановить информационное содержание сообщения.
5. Избыточность кода Бодо для русского языка $Q(r) = 0{,}129$, для английского $Q(e) = 0{,}193$.