Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Институт повышения квалификации
и переподготовки

Кафедра «Информатика»

В. И. Мисюткин

ЭЛЕМЕНТЫ ТЕОРИИ ИНФОРМАЦИИ

ПОСОБИЕ
по одноименной дисциплине
для слушателей специальности 1-40 01 73
«Программное обеспечение информационных систем»
заочной формы обучения

Гомель 2015

УДК 519.72(075.8)
ББК 32.811я73
М65

Рекомендовано кафедрой «Информатика» ГГТУ им. П. О. Сухого
(*протокол № 12 от 27.03.2015 г.*)

Рецензент: зав. каф. «Промышленная электроника» ГГТУ им. П. О. Сухого
канд. техн. наук, доц. *Ю. В. Крышнев*

Мисюткин, В. И.

М65 Элементы теории информации : пособие по одноим. дисциплине для слушателей специальности 1-40 01 73 «Программное обеспечение информационных систем» заоч. формы обучения / В. И. Мисюткин. – Гомель : ГГТУ им. П. О Сухого, 2015. – 87 с. Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: http://elib.gstu.by. – Загл. с титул. экрана.

Изложены основные вопросы, связанные с информацией: ее хранение, измерение, преобразование и передача. Рассмотрены варианты передачи информации по каналам связи при отсутствии и наличии помех, а также методы исправления ошибок, возникших при передаче сообщения. Изложены способы защиты информации от кражи и преднамеренных искажений методами криптографии.

Для слушателей специальности 1-40 01 73 «Программное обеспечение информационных систем» заочной формы обучения ИПКиП.

УДК 519.72(075.8)
ББК 32.8116я73

Содержание

Тема1. Основные понятия и определения теории информации

1.1. Предмет теории информации

Теория информации (ТИ) является одной из составных частей **кибернетики** – науки об общих законах получения, хранения, передачи и переработки информации.

Она представляет собой математическую теорию, посвященную измерению информации и ее потока, "размеров канала" связи и т. п., применительно к радио, телеграфии, телевидению и к другим средствам связи. Кроме того, ТИ изучает методы построения кодов, посредством которых осуществляется передача информации.

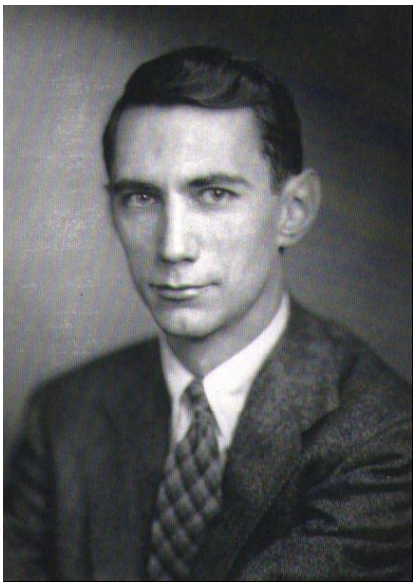
ТИ как самостоятельная дисциплина возникла в ходе решения следующей задачи: *необходимо обеспечить надежную и эффективную передачу информации от источника к приемнику при наличии помех.* При этом:

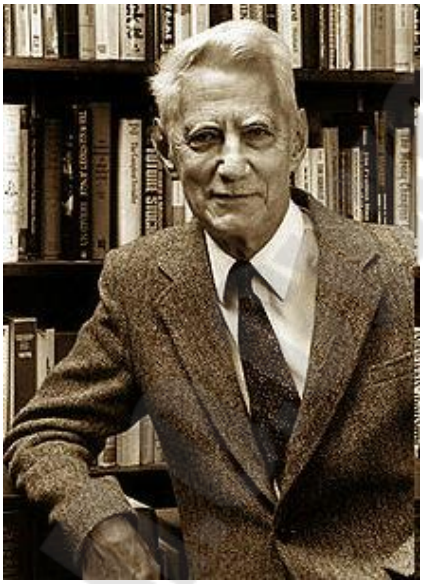
- "надежную" — означает, что в процессе передачи не должно происходить потерь информации;
- "эффективную" — означает, что передача должна осуществляться наиболее быстрым способом.

Решение этой задачи ведется по двум направлениям:

- *техническое* — связано с практической разработкой линий связи и технических устройств, обеспечивающих быструю и надежную связь; обеспечением защиты от помех или уменьшения их воздействия. В основе таких разработок лежат законы, определяющие способы кодирования информации и условия ее надежной передачи;
- м*атематическое* — связано с разработкой новых методов и алгоритмов кодирования и защиты информации от помех и искажений. Оно основывается на теории случайных событий и связано с возможностью количественного измерения информации.

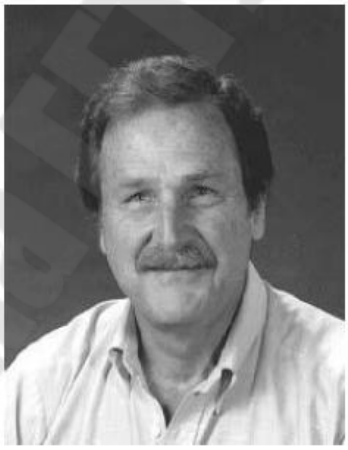
ТИ как наука существует с середины XX века с момента появления основополагающей работы Клода Шеннона (см. фото) "Математическая теория связи" (1948г.).

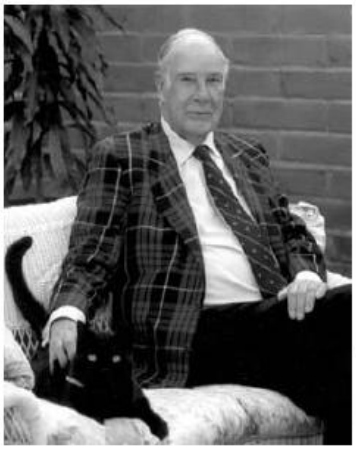




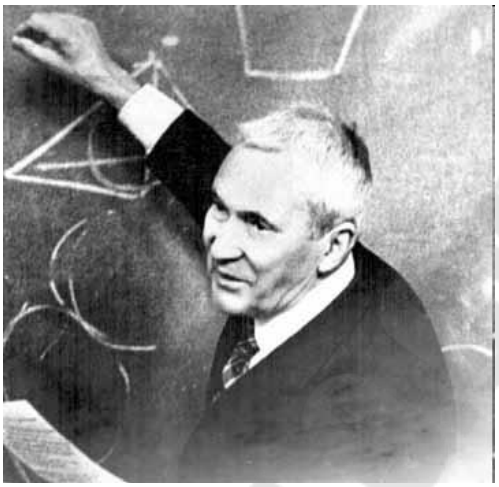
Клод Шеннон Элвуд

Значительный вклад в развитие ТИ внесли такие ученые как Р. Хартли, Давид Хаффмен, Ричард Хэмминг, а так же российские – В.А. Котельников, А.Н. Колмогоров и другие (см. фото).





Ральф Хартли, Давид Хаффмен и Ричард Хэмминг (слева направо)



Котельников Владимир Александрович (слева) и Колмого́ров Андре́й Никола́евич

ТИ тесно связана с такими разделами математики как теория вероятностей и математическая статистика, а также прикладная алгебра, которые представляют для нее математический фундамент.

ТИ делится на фундаментальную и прикладную.

Фундаментальная ТИ занимается исследованиями в областях:

- анализа сигналов как средства передачи сообщений и оценкой переносимого "количества информации";
- анализа информационных характеристик источников сообщений и каналов связи и обоснованием принципиальной возможности кодирования и декодирования сообщений, обеспечивающих предельно допустимую скорость передачи по каналу связи, как при отсутствии, так и при наличии помех.

Прикладная ТИ основывается на практических результатах, полученных при рассмотрении фундаментальных законов. Она занята разработкой конкретных методов и средств кодирования сообщений, а также изучением любых процессов, связанных с получением, передачей, хранением, обработкой и использованием информации.

Примеры практической применимости теории информации можно найти в информатике, технике, психологии, биологии, физике, педагогике, лингвистике и других областях.

1.2. Основные определения

Информация (от лат. Informatio) — сведения, разъяснения, изложение.

В зависимости от области знания существуют различные подходы к определению понятия информации. Под информацией понимают:

- в быту — это сведения об окружающем мире и протекающих в нем процессах, воспринимаемые человеком или специальными устройствами.
- в технике — это сообщения, передаваемые в форме знаков или сигналов.

В настоящее время нет общепринятого и однозначного понимания термина "информация". Понятие информации имеет много определений, от наиболее широкого (*информация* есть формализованное отражение реального мира) до практического (это - сведения и данные, являющиеся объектом хранения, передачи, преобразования, восприятия и управления).

Данные - это совокупность фактов, результатов наблюдений, измерений о каких-либо объектах, явлениях или процессах материального мира, представленных в формализованном виде: количественном или качественном.

Таким образом, информация – это отражение внешнего мира с помощью знаков и сигналов. Информация всегда связана с *материальным носителем*.

*Материальным носит*елем называют материальный объект или среду, которые служат для представления или передачи информации.

Материальными носителями информации могут быть: бумага, воздух, электромагнитное поле и пр. Хранение информации связано с некоторой характеристикой носителя, которая не меняется с течением времени (например: буквы на бумаге, намагниченность участка диска). Передача информации - с характеристикой, которая изменяется с течением времени (например: амплитуда колебаний звуковой волны, электромагнитные колебания).

Хранение информации связано с фиксацией состояния носителя, а распространение - с процессом, который протекает в носителе. Передача информации может осуществляться не всяким процессом. Стационарный процесс, т.е. процесс с неизменными в течение времени характеристиками (ровное горение электролампы), информацию не переносит. Для передачи необходим нестационарный процесс, т.е. процесс, характеристики которого могут изменяться во времени или пространстве. Если лампу включать и выключать, то чередованием вспышек и пауз можно представить и передать информацию (по та-

кому принципу построена азбука Морзе).

Сигнал – это физический процесс, несущий сообщение. Одиночный сигнал содержит очень мало информации, поэтому для передачи информации используется ряд следующих друг за другом сигналов. От источника к приемнику информация передается в виде *сообщений*.

Сообщение - это последовательность сигналов. Сообщение служит переносчиком информации, а информация является содержанием сообщения.

Соответствие между сообщением и содержащейся в нем информацией называется *правилом интерпретации сообщения*. Это соответствие может быть:

- однозначным - сообщение имеет лишь одно правило интерпретации (например, в азбуке Морзе символ "." – всегда обозначает букву "е").
- неоднозначным - сообщение может содержать различную информацию для разных приемников (например, передача в 1936 г. по радио фразы "Над всей Испанией безоблачное небо", которое для непосвященных людей имело смысл прогноза погоды, а для знакомых с правилом интерпретации - сигналом к началу военных действий; или трафарет с буквой М (для одних – это метро, для других – ресторан Макдональдс, а для третьих, извините, может и мужской туалет).

Источник информации - это субъект или объект, порождающий информацию и представляющий ее в виде сообщения.

Приемник информации - это субъект или объект, принимающий сообщение и способный правильно его интерпретировать.

Источники и приемники информации могут быть одушевленными или неодушевленными. Объект считается источником информации, если он не только ее порождает, но и создает сообщение. Например, если человек что-то придумал, но держит это в своей голове, то он еще не является источником информации; однако он им становится, как только свою идею изложит на бумаге или выскажет словами.

Факт приема сообщения то же еще не означает получение информации. Информация может считаться полученной только в том случае, если приемнику известно правило интерпретации сообщения. Например, слыша речь на незнакомом языке, человек оказывается приемником сообщения, но не приемником информации.

Совокупность технических средств используемых для передачи сообщений от источника к приемнику информации называется *системой связи*.

К ним относятся телеграф, телефон, радио и телевидение, компьютерные телекоммуникации и пр.

Общая схема передачи информации в системе связи представлена на рисунке 1.

Рисунок 1 - Схема передачи информации в системе связи

Преобразование сообщения в сигнал, удобный для передачи по данному каналу связи, называют *кодированием*. Операцию восстановления сообщения по принятому сигналу называют *декодированием*.

Под *линией связи* понимают любую физическую среду (воздух, металл, магнитную ленту и т. п.), обеспечивающую поступление сигналов от передающего устройства к приемному. Сигналы на выходе линии связи могут отличаться от переданных вследствие затухания, искажения или воздействия *помех*. Помехами называют любые мешающие передаче возмущения, как внешние (атмосферные или промышленные помехи), так и внутренние (их источник – сама аппаратура связи), вызывающие отклонения принятых сигналов от переданных. Эффект воздействия помех на различные элементы системы стараются учесть изменением характеристик линии связи. Поэтому источник помех условно относят к линии связи.

Из смеси сигнала и помехи приемное устройство выделяет сигнал и посредством декодера восстанавливает сообщение, которое в общем случае может отличаться от посланного. Меру соответствия принятого сообщения исходному называют *верностью передачи*. Обеспечение заданной верности передачи – важнейшая цель системы связи.

Принятое сообщение с выхода системы связи поступает к абоненту-получателю, которому была адресована исходная информация.

Совокупность средств, предназначенных для передачи сообще-

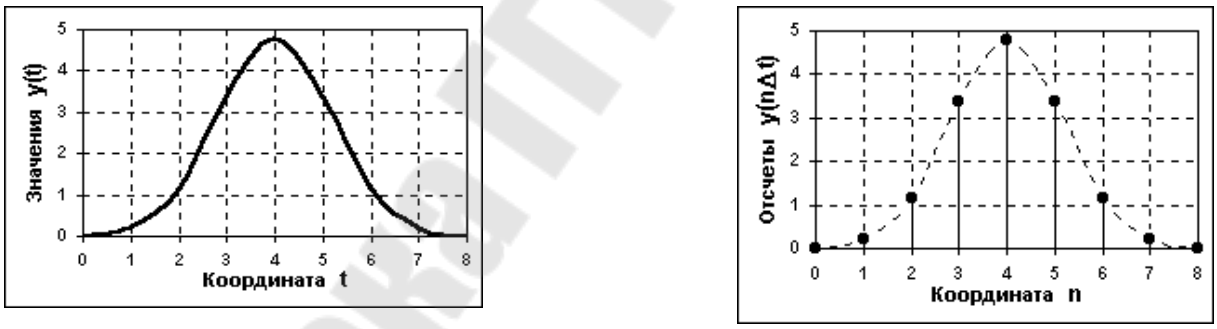
ний, называют *каналом связи*.

1.3. Формы представления информации

Существует два типа сигналов: непрерывные и дискретные. Сигнал называется *непрерывным* (*аналоговым*), если он определен для любого значения аргумента. Источниками аналоговых сигналов, как правило, являются физические процессы и явления, непрерывные в динамике своего развития во времени, в пространстве или по любой другой независимой переменной, при этом регистрируемый сигнал подобен ("аналогичен" – откуда термин - аналоговый) порождающему его процессу. Примеры непрерывных сигналов: речь, музыка, изображение, показание ртутного термометра и пр.

Сигнал называется *дискретным*, если множество его значений является конечным (счетным) и описывается дискретной последовательностью отсчетов. Примеры дискретных сигналов: показания цифровых измерительных приборов, книги, электронные табло и пр.

Обозначим через y(t) - значение параметра сигнала, а через t - время. Тогда непрерывный и аналоговый сигналы можно представить в виде рисунка 2.



а) б)

Рисунок - 2 а) вид непрерывного сигнала; б) вид дискретного сигнала

Принципиальным отличием непрерывных сигналов от дискретных является то, что дискретные сигналы можно обозначить, т.е. задать каждому значению сигнала знак (жест, рисунок, букву), который будет отличать одно значение сигнала от другого.

Знак - это элемент некоторого конечного множества отличных друг от друга сущностей. Вся совокупность знаков, используемых для представления дискретной информации, называется набором знаков.

Упорядоченный набор знаков называется *алфавитом*.

Порядок между знаками устанавливают отношения "больше - меньше": для двух знаков α и β принимается, что $\alpha<\beta$, если порядковый номер у α в алфавите меньше, чем у β.

Примером алфавита может служить совокупность арабских цифр 0,1...9 - с ее помощью можно записать любое целое число в системах счисления от двоичной до десятичной. Если же к этому алфавиту добавить знаки "+", "-", "." и ",", то сформируется набор знаков, который позволит записать любое вещественное число. Но этот набор уже нельзя считать алфавитом, поскольку в нем не определен порядок следования знаков.

При передаче сообщения параметр сигнала должен меняться. Минимальное количество различных его значений равно двум и, следовательно, алфавит содержит минимум два знака — такой алфавит называется *двоичным*. Верхней границы числа знаков в алфавите не существует.

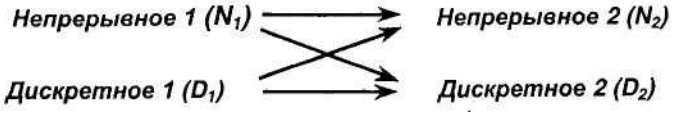
Знаки, используемые для обозначения фонем человеческого языка, называются *буквами*, а их совокупность - *алфавитом языка*.

Сами по себе знак или буква не несут никакого смыслового содержания. Если знаку приписано содержание, то он будет называться *символом*. Например, массу в физике обозначают буквой m, поэтому символ "m" является в формулах символом физической величины "масса". Другим примером символов могут служить пиктограммы, обозначающие в компьютерных программах объекты или действия:  .

Понятия "знак", "буква" и "символ" нельзя считать тождественными.

1.4. Преобразование сообщений

Поскольку имеются два типа сигналов, с помощью которых передаются сообщения, между ними возможны четыре варианта преобразований:



На практике применяются все четыре вида преобразований.

1. Преобразование типа $N_1 \rightarrow N_2$. Примерами устройств, осуществляющих такие преобразования, являются микрофон, магнитофон и видеомагнитофон, телекамера, радио- и телевизионный приемник. Преобразование $N_1 \rightarrow N_2$ всегда сопровождается частичной потерей информации. Потери связаны с внешними помехами или помехами, которые производит само информационное техническое устройство. Помехи примешиваются к основному сигналу и искажают его. Поскольку параметр сигнала может иметь любые значения невозможно однозначно определить был ли сигнал искажен или изначально имел такую величину. В ряде устройств искажение происходит в силу особенностей преобразования в них сообщения, например: в черно-белом телевидении, теряется цвет изображения; телефон пропускает звук в более узком частотном интервале, чем интервал воздействия человеческого голоса.

Передающее устройство преобразует непрерывные сообщения в сигналы, удобные для прохождения по линии связи (хранения). При этом один или несколько параметров выбранного носителя изменяют в соответствии с передаваемой информацией. Такой процесс называют *модуляцией*. Он осуществляется *модулятором*. Обратное преобразование осуществляется *демодулятором*.

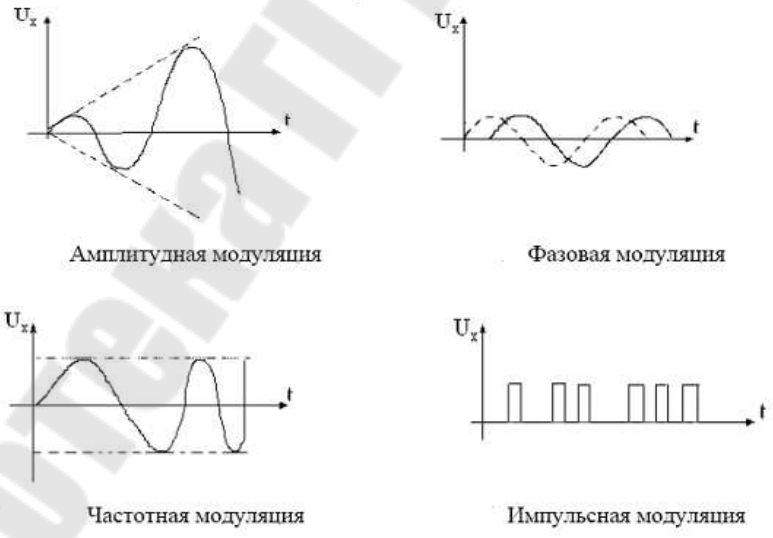


Рисунок 3 - Виды модуляции сигнала

2. Преобразование типа N→D. Перевод N→D означает замену описывающей его непрерывной функции y(t) на некотором отрезке времени $[t_1, t_2]$ конечным множеством $\{y_i, t_i\}$ (i = 0...n, где n- количество точек разбиения временного интервала). Такое преобразование назы-

вается *дискретизацией* непрерывного сигнала и производится с помощью двух операций: *дискретизацией по времени(по ост абсцисс) и квантованию по величине сигнала (по оси ординат)*.

Операция дискретизации осуществляет преобразование аналоговых сигналов, непрерывных по аргументу, в функции мгновенных значений сигналов по дискретному аргументу. Дискретизация обычно производится с постоянным шагом по аргументу (равномерная дискретизация), который определяется по формуле:

$$\Delta t = \frac{t_2 - t_1}{n}. \qquad (1.1)$$

Частота, с которой выполняются замеры аналогового сигнала, называется *частотой дискретизации*. В результате дискретизации непрерывный сигнал переводится в последовательность чисел.

Квантование по величине - это отображение вещественных значений параметра сигнала в конечное множество чисел, кратных некоторой постоянной величине - шагу квантования Δy.

Процесс преобразования отсчетов сигнала в числа называется квантованием по уровню, а возникающие при этом потери информации за счет округления – ошибками. При преобразовании аналогового сигнала непосредственно в цифровой сигнал операции дискретизации и квантования совмещаются.

Совместное выполнение обеих операций эквивалентно нанесению масштабной сетки на график y(t), как показано на рисунке 4. В качестве пар значений $\{y_i, t_i\}$ выбираются узлы сетки, расположенные наиболее близко к $y(t_i)$. Полученное таким образом множество узлов оказывается дискретным представлением исходной непрерывной функции. Таким образом, любое сообщение, связанное с ходом y(t), может быть преобразовано в дискретное, то есть представлено посредством некоторого алфавита.

При такой замене очевидно, что чем меньше n, тем меньше число узлов и, соответственно, меньший объем информации потребуется сохранять, но и точность замены y(t) значениями y_i (i=0,…,n) будет меньшей. При дискретизации может происходить потеря части информации, однако существуют условия, определяемые теоремой Котельникова, согласно которым аналоговый сигнал может быть без потерь информации преобразован в дискретный сигнал, и затем абсолютно точно восстановлен по значениям своих дискретных отсчетов.

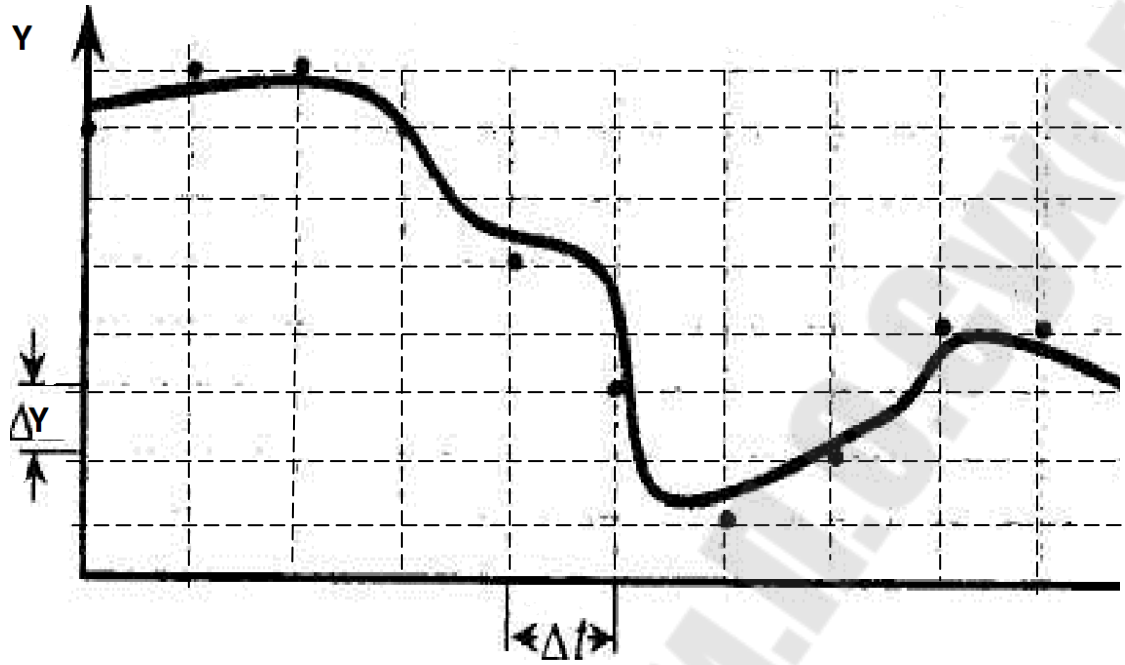


Рисунок 4 – Операции дискретизации и квантования непрерывного сигнала

Теорема отсчетов (В. А. Котельникова): непрерывный сигнал можно полностью отобразить и точно воссоздать по последовательности измерений или отсчетов величины этого сигнала через одинаковые интервалы времени, меньшие или равные половине периода максимальной частоты, имеющейся в сигнале.

Согласно теореме, определяющим является значение верхней границы частоты - ν_m. Смысл теоремы в том, что дискретизация не приведет к потере информации, и по дискретным сигналам можно будет полностью восстановить исходный аналоговый сигнал, если развертка по времени выполнена в соответствии со следующим соотношением:

$$\Delta t \le \frac{1}{2 \cdot \nu_m} \qquad (1.2)$$

Например, для точной передачи речевого сигнала с частотой до ν_m = 4000 Гц при дискретной записи должно производиться не менее 8000 отсчетов в секунду; в телевизионном сигнале ν_m = 4 МГц, следовательно, для его точной передачи потребуется около 8000000 отсчетов в секунду.

Шаг квантования Δy определяется, исходя из следующих соображений. Получатель сообщения — человек или устройство - всегда имеют конечную предельную точность распознавания величины сигнала. Например, человеческий глаз в состоянии различить около 16 миллионов цветов. Это означает, что при квантовании цвета нет смысла делать большее число градаций. Поэтому шаг квантования выбирается, исходя из чувствительности приемного устройства.

Указанные соображения по выбору шага дискретизации по времени и квантования по величине сигнала лежат в основе оцифровки звука и изображения. Примеры таких устройств: сканер, модем, устройства для цифровой записи звука и изображения, лазерный проигрыватель. Термины "цифровая запись", "цифровой сигнал" следует понимать как дискретное представление с применением двоичного цифрового алфавита.

Примером использования этой теоремы являются лазерные компакт-диски, звуковая информация на которых хранится в цифровой форме. Чем выше будет частота дискретизации, тем точнее будут воспроизводиться звуки и тем меньше их можно будет записать на один диск. Но ухо обычного человека способно различать звуки с частотой до 20 КГц, поэтому точно записывать звуки с большей частотой бессмысленно. Согласно рассмотренной теореме частоту дискретизации нужно выбрать не меньшей 40 КГц (в промышленном стандарте на компакт-диске используется частота 44,1 КГц).

Таким образом, преобразование $N \rightarrow D$ и обратное $D \rightarrow N$, может осуществляться без потери содержащейся в них информации, а рациональное выполнение операций дискретизации и квантования приводит к снижению затрат на передачу, обработку и хранение информации.

3. Преобразование типа $D \rightarrow N$. В этом случае определяющей является скорость этого преобразования: чем она выше, с тем более высокочастотными гармониками получится непрерывная величина. Но чем большие частоты встречаются в этой величине, тем сложнее с ней работать. В дальнейшем нас это преобразование интересовать не будет, поэтому его подробности здесь не рассматриваются.

4. Преобразование типа $D_1 \rightarrow D_2$ состоит в переходе при представлении сигналов от одного алфавита к другому. Такая операция носит название перекодировка и может осуществляться без потерь. Примеры: запись-считывание с компьютерных носителей информации; шифровка и дешифровка текста; вычисления на калькуляторе.

Таким образом, за исключением $N_1 \rightarrow N_2$ в остальных случаях оказывается возможным преобразование сообщений без потерь информации. При этом непрерывные и дискретные сообщения не являются равноправными. Сохранение информации в преобразованиях $N \rightarrow D$ и $D \rightarrow N$ обеспечивается благодаря участию в них дискретного представления. Другими словами, преобразование сообщений без потерь информации возможно только в том случае, если хотя бы одно из них является дискретным. В этом проявляется несимметричность видов сообщений и преимущество дискретной формы. К другим ее достоинствам следует отнести:

- применимость таких методов кодирования, которые обеспечивают обнаружение и исправление ошибок;
- возможность избежать свойственного аналоговым сигналам эффекта накопления искажений в процессе их передачи и обработки, поскольку квантованный сигнал легко восстановить до первоначального значения всякий раз, когда величина накопленного искажения становится значимой;
- высокую помехоустойчивость;
- простоту, надежность и относительную дешевизну устройств по обработке информации;
- универсальность устройств для работы с дискретной информацией.

Универсальность дискретных сообщений проявляется в том, что сообщения, составленные в различных алфавитах, с помощью обратимого кодирования можно привести к единому алфавиту. Это позволяет выделить некоторый алфавит в качестве *базового* и представлять в нем любую дискретную информацию. Устройство, работающее с информацией в базовом алфавите, является универсальным, так как оно может быть использовано для переработки любой иной исходной дискретной информации. Базовым алфавитом является *двоичный алфавит*, а универсальным устройством - *компьютер*.

Дискретная форма представления информации по отношению к непрерывной является приоритетной в решении глобальной задачи автоматизации обработки информации. В дальнейшем будем исследовать только дискретную информацию, а для ее представления использовать некоторый алфавит.

Тема 2. Количественная оценка информации

2.1. Понятие энтропии, как меры неопределенности

Факт получения информации от источника всегда связан с уменьшением неопределенности о его состоянии. Степень неопределенности зависит от числа возможных состояний источника. Прежде чем мы установим количественную меру неопределенности для информации, рассмотрим два примера, поясняющих, почему это нужно сделать.

Пример 2.1. Двое играют в шахматы. Один – белыми фигурами, другой – черными. Неопределенность состоит в том, что неизвестно кто начнет игру. В данном случае возможны две ситуации, возникающие с равной вероятностью 1/2: одним из игроков будет выбрана либо белая фигура, либо черная из двух, предложенных "в темную" судьей. Неопределенность обеих ситуаций одинакова, а значит, и количество информации, связанное с каждой ситуацией, тоже будет одинаковым. Роль источника информации может сыграть судья матча, который объявляет, кто будет играть белыми фигурами.

Пример 2.2. На 32 карточках написано по одной букве русского алфавита. Вынимают одну карточку. Вынутой может оказаться любая карточка с вероятностью 1/32. Интуитивно, неопределенность того, какая буква окажется написанной на карточке, большая по сравнению с предыдущим примером, так как в этом случае больше различных ситуаций, а именно, 32. Но насколько она большая, пока мы ответить не можем.

Из приведенных примеров видно, что на практике важно иметь возможность произвести численную оценку неопределенности разных состояний источника информации. Для этого обратимся к теории вероятностей, подразумевая, что исход опыта эквивалентен выбранному состоянию источника информации. Тогда количество возможных исходов опыта есть не что иное, как количество всевозможных состояний N дискретного источника информации, которые он может принять в каждый момент времени случайным образом. Его различные состояния $\boldsymbol{u_i}$ реализуются случайным образом, а весь ансамбль состояний $\boldsymbol{U}$ характеризуется суммой вероятностей их появления:

$$\sum_{i=1}^{N} p(u_i) = 1 \qquad (2.1)$$

Попытаемся выбрать меру для оценки неопределенности со-

стояния источника.

В качестве такой меры нужно использовать непрерывную функцию, зависящую от числа состояний источника N (при равных вероятностях всех состояний) – $f(N)$. Она должна следующим трем требованиям:

1. $f(1)=0$, так как при $N=1$ исход опыта не является случайным и неопределенность отсутствует;
2. условию монотонного возрастания при увеличении числа возможных состояний источника N;
3. условию *аддитивности*, которое формулируется следующим образом: *если два независимых источника с числом равновероятных состояний N и M, соответственно, рассматривать как один источник, одновременно реализующий пары состояний $n_i m_j$, то неопределенность объединенного источника должна равняться сумме неопределенности исходных источников.*

$$f(NM) = f(N)+f(M). \qquad (2.2)$$

Соотношение (2.2) выполняется, если в качестве меры неопределенности источника с равновероятными состояниями принять функцию:

$$H(U)=\log N. \qquad (2.3)$$

Для этой функции выполняются все три выдвинутые нами требования:

1. log(1)=0;
2. функция монотонно возрастает с возрастанием N;
3. выполняется условие аддитивности, так как

$$\log(N \cdot M) = \log(N) + \log(M).$$

В принципе, основание логарифма может быть любым, так как оно определяет только масштаб или единицу измерения неопределенности. Самым удобным основанием оказывается число 2. Это хорошо согласуется с применяемой в компьютерах двоичной системой счисления.

Единица измерения неопределенности при двух возможных равновероятных состояниях источника называется ***бит*** (от английского **binary digit** "двоичный разряд" или "двоичная единица"). Еди-

ницы при других основаниях логарифма: трит (основание логарифма равно 3), нат (натуральный логарифм).

Таким образом, за функцию, описывающую меру неопределенности источника, имеющего N равновероятных исходов, принимается функция:

$$H=f(N)=\log_2 N \qquad (2.4)$$

Эта величина получила название ***энтропии***.

У источника с N равновероятными состояниями неопределенность, вносимая одним состоянием, составляет

$$H = \frac{1}{N}\log_2 N = -\frac{1}{N}\log_2 \frac{1}{N} = -p \cdot \log_2 p \qquad (2.5)$$

где $p = \frac{1}{N}$ - вероятность любого из отдельных исходов.

Если источник сообщений имеет N состояний A_1, A_2,....,A_N, вероятности которых неодинаковы и равны $p(A_1)$, $p(A_2)$, ..., $p(A_N)$, то:

$$H(U) = -\sum_{i=1}^{N} p(A_i) \cdot \log_2 p(A_i) \qquad (2.6)$$

Энтропия является мерой неопределенности дискретного источника информации. Эта формула была предложена К. Шенноном.

Пример 2.3. Имеются два ящика, в каждом из которых лежит по 12 шаров. В первом -3 белых, 3 черных и 6 красных; во втором - каждого цвета по 4. Опыты состоят в вытаскивании по одному шару из каждого ящика. Что можно сказать относительно неопределенностей исходов этих опытов?

Решение. Согласно (2.6) находим энтропии обоих опытов:

$$H_A = -\frac{3}{12}\cdot\log_2\frac{3}{12} - \frac{3}{12}\cdot\log_2\frac{3}{12} - \frac{6}{12}\cdot\log_2\frac{6}{12} = 1{,}50 \quad \textit{бит},$$

$$H_B = -\frac{4}{12}\cdot\log_2\frac{4}{12} - \frac{4}{12}\cdot\log_2\frac{4}{12} - \frac{4}{12}\cdot\log_2\frac{4}{12} = 1{,}58 \quad \textit{бит}.$$

Поскольку $H_B > H_A$, неопределенность исхода в опыте B выше и, следовательно, предсказать его можно с меньшей долей уверенности, чем исход опыта A.

Некоторые свойства энтропии

1. Энтропия является вещественной и неотрицательной величиной, т.к. для любого значения p_i (i=1,…,*N*), которое изменяется в пределах от 0 до 1, $log\ p_i$ отрицателен (при 0 – он не существует) и, следовательно, $- p_i \cdot log\ p_i$ – положительно.
2. Энтропия – величина ограниченная. Для слагаемых $- p_i \cdot log\ p_i$ в диапазоне $0 < p_i < 1$ ограниченность очевидна, а предел для $- p_i \cdot log\ p_i$ при $p_i \rightarrow 0$, по правилу Лопиталя, равен 0.
3. Энтропия равна нулю, если вероятность одного из состояний источника равна 1.
4. Энтропия максимальна, когда все состояния источника равновероятны, что доказывается использованием метода неопределенных множителей Лагранжа.
5. Энтропия источника u с двумя состояниями u_1 и u_2 изменяется от 0 до 1, достигая максимума при равенстве их вероятностей:

$$p(u_1)=p)=p(u_2)=1- p=0,5$$

Связь между энтропией и информацией

Если есть два источника информации α и β, которые зависимы друг от друга, то получение информации от предшествующего источника α может уменьшить неопределенность состояния последующего источника β. Разность между $H(\alpha)$ и $H_\alpha(\beta)$ показывает, какие новые сведения относительно состояния β мы получаем, получив информацию от источника α. Эта величина называется *информацией, содержащейся в источнике α относительно источника β.* Таким образом, получение информации об источнике снижает энтропию этого источника, причем именно на ту величину, которую составляет объем полученной информации.

Это позволяет сделать следующий вывод: *поскольку единицей измерения энтропии является бит, то в этих же единицах может быть измерено и количество информации.* Информацию принято обозначать буквой ***I.***

Бит – это очень маленькая единица информации, поэтому часто используется величина в 8 раз большая – *байт* (*byte)*, состоящая из 8 бит (двух тетрад = 2×4). Байт обозначают обычно заглавной русской буквой "Б" или латинской "В". Существуют и более крупные единицы измерения, являющиеся производными от бита и байта и образуемые при помощи приставок: кило (К/К), мега (М), гига(Г/G), тера (Т), пета

(П/Р) и др. Но для битов и байтов они означают не степени 10, а степени двойки: кило – 2^{10} =1024 ≈ 10^3 , мега – $2^{20} \approx 10^6$, гига – $2^{30} \approx 10^9$ и т.п. Например, 1 КБ=8 Кбит = 1024Б = 8192 бит.

Свойства информации:

1. $I(\alpha,\beta) \geq 0$, причем $I(\alpha,\beta) = 0$ тогда и только тогда, когда источники информации α и β независимы.
2. $I(\alpha,\beta) = I(\beta,\alpha)$, т.е. количество информации, содержащейся в α относительно β, равно количеству информации, содержащейся в β относительно α.

Среднее значение количества информации, которое содержит источник по аналогии с (2.6) можно определить по формуле:

$$I = -\sum_{i=1}^{N} p(A_i) \cdot \log_2 p(A_i) \qquad (2.7)$$

т.е. оно равно среднему значению количества информации, содержащейся во всех его состояниях.

Это знаменитая формула К.Шеннона, с работы которого ("Математическая теория связи", 1948г.), принято начинать отсчет возраста информатики, как самостоятельной науки.

Пример 2.4. Какое количество информации требуется, чтобы узнать исход броска монеты?

В данном случае число возможных исходов N = 2 ("решка" или"орел"), и события равновероятны, т.е. $p_1=p_2$ = 0,5, следовательно из (2.7):

$$I = -0{,}5 \cdot \log_2 0{,}5 - 0{,}5 \cdot \log_2 0{,}5 = 1 \text{ бит},$$

т. е. достаточно задать один вопрос (например: "Выпала решка?"), чтобы узнать, каков был исход данного опыта . Такое совпадение между количеством информации и числом вопросов с бинарными ответами ("Да"/"Нет") неслучайно.

Количество информации численно равно, числу вопросов с равновероятными бинарными вариантами ответов, которые необходимо задать, чтобы полностью снять неопределенность задачи.

Рассмотрим случай, когда все N исходов равновероятны, т.е. все

$$p(A_i) = \frac{1}{N}.$$

Тогда из (2.7) получаем

$$I = -\sum_{i=1}^{N} \frac{1}{N} \cdot \log_2 \frac{1}{N} = \log_2 N \qquad (2.8)$$

Эта формула была выведена в 1928 г. американским инженером Р. Хартли и носит его имя. Ее смысл в том, что, если некоторое множество содержит N равновероятностных элементов, то для однозначной идентификации одного элемента среди прочих требуется количество информации, равное $log_2 N$.

Частным случаем применения формулы (2.8) является ситуация, когда $N=2^k$. Подставляя это значение в (2.8), получим:

$$I = k \text{ бит.} \qquad (2.9)$$

Вывод: *к* равно количеству вопросов с бинарными равновероятными ответами, которые определяют количество информации, необходимое для снятия неопределенности состояния подобного источника.

Пример 2.5. Имеется алфавит A, из m букв которого, составляется сообщение длиной n. Тогда, количество возможных вариантов разных сообщений:

$$N = m^n.$$

Таким образом, если букв в алфавите две, например: «В» и «Х», а длина сообщения 3 буквы — то есть: m=2, n=3 - можно составить $N = m^n = 2^3 = 8$ разных сообщений:«ВВВ», «ВВХ», «ВХВ», «ВХХ», «ХВВ», «ХВХ», «ХХВ», «ХХХ» — других вариантов нет.

Пример 2.6. Случайным образом вынимается карта из колоды в 32 карты. Какое количество информации требуется, чтобы узнать, что это за карта? Как построить угадывание?

Решение

Для данной ситуации $N = 2^5$, а $к = 5$ и, следовательно, $I = 5$ бит. Таким образом, за 5 заданных вопросов с бинарными ответами можно угадать карту. Рассмотрим один из возможных вариантов угадывания (представлен на рисунке 2.1), считая, что в колоде 4 масти (черви, бубны, кресты и пики); каждая масть состоит из 8 карт (7,8,9,10, валет, дама, король и туз, а "мальчики" – это валет или король).

Рисунок 2.1. Схема угадывания карты

2.3. Информация и алфавит

При передаче сообщения возникает проблема *распознавания знака*, то есть по полученным сигналам нужно установить исходную последовательность знаков первичного алфавита.

В устной речи это достигается использованием звуков. В письменности это достигается различным начертанием букв и дальнейшим анализом написанного. Узнавание знака требует получения некоторой порции информации. Можно связать эту информацию с са-

мим знаком и считать, что знак несет в себе некоторое количество информации. Оценим это количество с различной степенью приближения.

Нулевое приближение. Пусть появление всех знаков алфавита в сообщении *равновероятно.* Тогда для английского алфавита N_e = 27 (с учетом пробела); для русского алфавита N_r = 34. Из формулы Хартли (2.8) находим:

$$I_0^{(e)} = \log_2 27 = 4{,}755 \text{ бит},$$
$$I_0^{(r)} = \log_2 34 = 5{,}087 \text{ бит}.$$

То есть в нулевом приближении с одним знаком русского алфавита в среднем связано больше информации, чем со знаком английского. Значит, с точки зрения техники, сообщения, состоящие из равного количества символов английского и русского языков, будут иметь разную длину и, соответственно, разное время передачи. Большими они окажутся у сообщений на русском языке.

Первое приближение. Учтем, что вероятность появления различных букв в тексте различна. Считая, что е=ё, ь=ъ (так принято в телеграфном кодировании), получим для русского языка алфавит из 32 знаков (включая пробел) со следующими вероятностями их появления в русских текстах (таблица 2.1):

Таблица 2.1

буква	p_i	буква	p_i	буква	p_i	буква	p_i	буква	p_i
а	0,064	з	0,015	о	0,096	х	0,009	э	0,003
б	0,015	и	0,064	п	0,024	ц	0,004	ю	0,007
в	0,039	й	0,010	р	0,041	ч	0,013	я	0,019
г	0,014	к	0,029	с	0,047	ш	0,006	про-бел	0,124
д	0,026	л	0,036	т	0,056	щ	0,003		
е,ё	0,074	м	0,026	у	0,021	ъ,ь	0,015		
ж	0,008	н	0,056	ф	0,020	ы	0,016		

Если считать, что p_i - вероятность появления знака номер i данного алфавита, то среднее количество информации, приходящейся на один знак, определяется по формуле Шеннона (2.7).

Сообщения, в которых вероятность появления каждого отдельного знака не меняется со временем, называются *шенноновскими сообщениями,* а порождающий их отправитель - *шенноновским источником.*

Если сообщение является шенноновским, то набор знаков (алфавит) и связанная с каждым знаком информация, известны заранее. В этом случае интерпретация сообщения, сводится к задаче *распознавания знака*.

Теория информации строится именно для шенноновских сообщений, поэтому в дальнейшем будем рассматривать только такие сообщения.

Применение формулы (2.7) дает следующие значения средней информации на один знак алфавита:

- для русского языка $I_1^{(r)}$= 4,36 бит;
- для английского $I_1^{(e)}$= 4,04 бит;
- для французского $I_1^{(f)}$= 3,96 бит;
- для немецкого $I_1^{(d)}$= 4,10 бит;
- для испанского $I_1^{(s)}$= 3,98 бит.

Второе приближение. Значение средней информации на букву может быть уменьшено с учетом связей (*корреляции*) между буквами в словах. В словах буквы появляются не в любых сочетаниях. Это понижает неопределенность угадывания следующей буквы, например, в русском языке нет слов, в которых встречается сочетание *щц* или *фъ*. И наоборот, после распространенного сочетания *пр* всегда следует гласная буква, а их в русском языке 10 и, следовательно, вероятность угадывания следующей буквы 1/10, а не 1/33.

Учет в английских словах двухбуквенных сочетаний понижает среднюю информацию на знак до значения $I_2^{(e)}$= 3,32 бит, учет трехбуквенных - до $I_3^{(e)}$= 3,10 бит. Для русского языка дают; $I_2^{(r)}$= 3,52 бит; $I_3^{(r)}$= 3,01 бит.

Последовательность I_0, I_1, I_2 является убывающей в любом языке. Шеннон ввел величину, которую назвал *относительной избыточностью языка:*

$$R = 1 - \frac{I_\infty}{I_0} \qquad (2.10)$$

Где I_∞ — *предельная наименьшая* информация на знак в данном языке, I_0 —*наибольшая информация,* которая может содержаться в знаке данного алфавита. Избыточность показывает, какую долю лишней информации содержат тексты данного языка или структура самого языка, когда текст может быть восстановлен в буквенном виде.

Исследования Шеннона для английского языка дали значение $I_\infty \approx 1{,}4 \div 1{,}5$ бит, что по отношению к I_0= 4,755 бит создает избыточ-

ность около 0,68. Для русского избыточность составляет ≈60 - 70%. Это означает, что возможно почти трехкратное сокращение текстов без ущерба для их содержательной стороны. Например, телеграфные тексты делаются короче за счет отбрасывания союзов и предлогов; в них же используются однозначно интерпретируемые сокращения "ЗПТ" и "ТЧК" вместо полных слов. Такое "экономичное" представление слов снижает разборчивость языка и уменьшает возможность понимания речи при наличии шума. Избыточность языка позволяет легко восстановить текст, даже если он содержит большое число ошибок или неполон (примерами являются кроссворды, игра "Поле чудес"). В этом смысле избыточность есть определенная страховка и гарантия разборчивости.

Тема 3. Кодирование символьной информации

3.1. Постановка задачи кодирования. Первая теорема Шеннона

Источник представляет информацию в форме дискретного сообщения, используя для этого алфавит, который называется *первичным.* Сообщение попадает в устройство - *кодер*, преобразующее и представляющее его в другом алфавите, который называется *вторичным*.

Код - это правило, описывающее соответствие знаков или их сочетаний первичного алфавита знакам или их сочетаниям вторичного алфавита.

Кодирование - перевод информации, представленной сообщением в первичном алфавите, в последовательность кодов вторичного алфавита.

Декодирование - восстановление информации в первичном алфавите по полученной последовательности кодов.

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

Пример обратимого кодирования - представление знаков в телеграфном коде и их восстановление после передачи, необратимого - перевод с одного естественного языка на другой. Будем рассматривать только обратимое кодирование.

Пусть первичный алфавит A состоит из N знаков со средней информацией на знак $I^{(A)}$, а вторичный алфавит B - из M знаков со средней информацией на знак $I^{(B)}$. Если исходное сообщение, представленное в первичном алфавите, содержит n знаков, а закодированное сообщение - m знаков, то условие обратимости кодирования запишется в виде:

$$n \cdot I^{(A)} \le m \cdot I^{(B)} \text{ или } \mathrm{I(A)} \le \frac{\mathrm{m}}{\mathrm{n}} \cdot \mathrm{I(B)}$$

Это значит, что операция обратимого кодирования может увеличить количество информации в сообщении, но не может его уменьшить.

Отношение $\frac{m}{n}$ характеризует среднее число знаков вторичного

алфавита, которое приходится использовать для кодирования одного знака первичного алфавита. Оно называется *длиной кода.* Обозначим его *K(A,B).* Тогда

$$K(A,B) \geq \frac{I^{(A)}}{I^{(B)}} \qquad (3.1)$$

Обычно $N>M$ и $I^{(A)}>I^{(B)}$, откуда $K(A,B)>1$, т.е. один знак первичного алфавита представляется несколькими знаками вторичного. Способов построения кодов, при фиксированных алфавитах А и В существует множество, поэтому возникает проблема выбора *оптимального кода,* который при передаче информации позволит затратить на передачу сообщения меньше времени.

Как следует из (3.1), минимально возможным значением средней длины кода будет:

$$K^{\min}(A,B) = \frac{I^{(A)}}{I^{(B)}} \qquad (3.2)$$

Выражение (3.2) устанавливает нижний предел длины кода, но из него неясно как реально приблизить $K(A,B)$ к $K^{min}(A,B)$. По этой причине, для теории кодирования и теории связи важнейшее значение имеют две теоремы, доказанные Шенноном. Первая затрагивает ситуацию с кодированием при отсутствии помех, искажающих сообщение. Вторая теорема относится к реальным линиям связи с помехами.

Первая теорема Шеннона (*основная теорема о кодировании при отсутствии помех*): При отсутствии помех всегда возможен такой вариант кодирования сообщения, при котором среднее число знаков кода, приходящихся на один знак первичного алфавита, будет сколь угодно близко к отношению средних количеств информации на знак первичного и вторичного алфавитов.

В ситуации, рассмотренной К. Шенноном, при кодировании сообщения в первичном алфавите учитывается различная вероятность появления знаков, но их зависимости не отслеживаются. Источники подобных сообщений называются *источниками без памяти.* Если при этом обеспечена равная вероятность появления знаков вторичного алфавита, то, как следует из (3.2), для минимальной средней длины кода оказывается справедливым соотношение:

$$K^{\min}(A,B)=\frac{I_1^{(A)}}{\log_2 M} \qquad (3.3)$$

Превышение $K(A,B)$ над $K^{min}(A,B)$ называется *относительной избыточность кода*, которая определяется по формуле:

$$Q(A,B)=\frac{K(A,B)-K^{\min}(A,B)}{K^{\min}(A,B)}=\frac{K(A,B)}{K^{\min}(A,B)}-1=\frac{K(A,B)\cdot I^{(B)}}{I^{(A)}}-1 \qquad (3.4)$$

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

$Q(A,B)\to 0$ при $K(A,B)\to K^{min}(A,B)$. Оптимизация кода состоит в нахождении таких схем кодирования, которые обеспечили бы приближение средней длины кода к значению $K^{min}(A, B)$.

Используя понятие избыточности кода, можно построить иную формулировку теоремы Шеннона:

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

Наиболее важной для практики оказывается ситуация, когда $M=2$, т.е. когда во вторичном алфавите используется два типа сигналов. Такое кодирование называется *двоичным.* Технически это наиболее просто реализуемый вариант, например, существование напряжения в проводе - *импульс* (1) или отсутствие – *пауза* (0). Удобство двоичных кодов и в том, что при равной длительности и вероятности каждый элементарный сигнал несет в себе 1 бит информации ($log_2 2 = 1$).

Применение формулы (3.4) для двоичных сообщений источника без памяти при кодировании знаками равной вероятности дает формулу:

$$Q(A,2)=\frac{K(A,2)}{I_1^{(A)}}-1 \qquad (3.5)$$

При декодировании двоичных сообщений возникает проблема

выделения из потока сигналов (последовательности импульсов и пауз) кодовых слов (групп элементарных сигналов), соответствующих отдельным знакам первичного алфавита. При этом приемное устройство фиксирует *интенсивность* и *длительность* сигналов и может сравнивать принятую последовательность сигналов с эталонной *(таблицей кодов).*

В этом случае возможны следующие особенности вторичного алфавита, используемого при кодировании:

- элементарные сигналы (0 и 1) могут иметь одинаковые длительности ($\tau_o = \tau_1$) или разные ($\tau_o \neq \tau_1$);
- длина кода может быть одинаковой для всех знаков первичного алфавита (*равномерный* код) или коды разных знаков первичного алфавита могут иметь различную длину (*неравномерный* код);
- коды могут строиться для отдельного знака первичного алфавита *(алфавитное кодирование)* или для их комбинаций *(кодирование блоков, слов).*

3.2. Способы построения двоичных кодов

3.2.1. Алфавитное неравномерное двоичное кодирование сигналами равной длительности. Префиксные коды

В случае неравномерного кодирования необходимо построить такую схему кодирования, при которой суммарная длительность кодов при передаче данного сообщения была бы наименьшей. Длительность сообщения будет меньше, если знакам первичного алфавита, которые встречаются в сообщении чаще, присвоить меньшие по длине коды, а тем, которые реже - более длинные.

Параллельно должна решаться проблема *различимости кодов.* Если бы код был равномерным, приемное устройство при декодировании просто отсчитывало бы заданное число элементарных сигналов (например, 5) и интерпретировало их в соответствии с кодовой таблицей. При использовании неравномерного кодирования возможны два подхода к обеспечению различимости кодов:

1. используются комбинации элементарных сигналов, которые интерпретируются декодером как разделители;
2. применяются *префиксные коды*.

Неравномерный код с разделителем

Пусть разделителем отдельных кодов букв будет - *00* (признак конца знака), а разделителем слов - *000* (пробел). Тогда код строится по правилам:

1) коды букв не должны содержать двух и более нулей подряд в середине;
2) код буквы (кроме пробела) всегда должен начинаться с 1;
3) разделителю слов *000* всегда предшествует признак конца знака *00*, при этом реализуется последовательность *00000*.

В соответствии с правилами, построим кодовую таблицу 3.1 для букв русского алфавита, основываясь на приведенных ранее в таблице 2.1. вероятностях появления отдельных букв. Код каждой следующей буквы получается добавлением единицы к коду предыдущей (по правилам двоичного сложения), а в ситуации, когда нарушается первое правило создания неравномерных кодов (не должно быть 2-х нулей в середине), запрещенный код пропускается. В таблице 3.1. коды букв записаны с разделителем 00 в конце.

Таблица 3.1.

Буква	**Код**	$\mathbf{p_i*10^3}$	$\mathbf{k_i}$	**Буква**	**Код**	$\mathbf{p_i*10^3}$	$\mathbf{k_i}$
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	10111100	3	8
п	1010000	23	7	э	11010000	3	8
у	1010100	21	7	ф	11010100	2	8

Теперь можно найти среднюю длину кода *K(r,2)* для данного способа кодирования:

$$K(r,2) = \sum_{j=1}^{32} p_j k_j = 4.964$$

Для русского языка $I^{(r}{}_1{}^{)}$ = 4,356 бит, избыточность кода по (3.5)

$$Q(r,2) = 4{,}964/4{,}356\text{-}1 = 0{,}14.$$

Это означает, что при данном способе кодирования будет передаваться на 14% больше информации, чем содержит исходное сообщение. Аналогичные вычисления для английского языка дают значение

$$K(r,2) = 4{,}716,\ I^{(e)}{}_1 = 4{,}036 \text{ бит},\ Q(e,2) = 0{,}168.$$

Код называется *префиксным*, если он удовлетворяет *условию Фано*:

Неравномерный код может быть однозначно декодирован, если никакой из кодов не совпадает с началом какого-либо иного, более длинного кода.

Например, если имеется код *110*, то уже не могут использоваться коды *1, 11, 1101, 110101* и пр. При использовании префиксного кодирования не нужно передавать разделители знаков, что делает сообщение более коротким.

Префиксный код Шеннона – Фано

Пусть имеется алфавит A, состоящий из знаков $a_1, a_2, a_3, \ldots$ и известны вероятности появления его знаков в сообщениях $p_1, p_2, p_3, \ldots$ Алгоритм создания неравномерного префиксного Шеннона – Фано запишется следующим образом.

1. Расположить знаки алфавита в порядке убывания вероятностей их появления в текстах.
2. Разделить знаки на две группы так, чтобы суммы вероятностей в каждой из них были бы приблизительно равными.
3. Первой группе присвоить знак кода "0", а второй группе - знак кода "1".
4. Продолжить деление каждой из групп на подгруппы по этой же схеме (п. 2÷3).

Рассмотрим применение этого алгоритма на следующем примере.

Пример 3.1. Первичный алфавит *A,* состоит из шести знаков a_1 ...a_6 с вероятностями появления в сообщении 0,3; 0,2; 0,2; 0,15; 0,1; 0,05, соответственно. Требуется создать для этого алфавита префиксный код Фано.

Решение

1. Расположим эти знаки в порядке убывания вероятностей.
2. Разделим знаки на две группы так, чтобы суммы вероятностей в каждой из них были бы приблизительно равными.
3. Первой группе (a_1 и a_2 с суммой вероятностей 0,5) присвоим первый знак кода "0". Второй группе (сумма также равна 0,5) - первый знак кода "1".
4. Продолжим деление каждой из групп на подгруппы по этой же схеме. В результате получаем следующую таблицу

Знак	p_i	Разряды кода				Код
		1	2	3	4	
a_1	0,30	0	0			00
a_2	0,20	0	1			01
a_3	0,20	1	0			10
a_4	0,15	1	1	0		110
a_5	0,10	1	1	1	0	1110
a_6	0,05	1	1	1	1	1111

Коды удовлетворяют условию Фано и, следовательно, полученный код является префиксным.

Средняя длина кода равна:

$K(A,2)$=0,3×2+0,2×2+0,2×2+0,15×3+0,1×4+0,05×4=2,45 символов.

Количество информации на один знак алфавита будет равно:

$$I^{(A)}_1 = 2{,}409 \text{ бит.}$$

Избыточность кода $Q(A,2)$=0.017, т.е. около 1,7%.

Данный код не оптимальный, т.к. вероятности появления 0 и 1 неодинаковы (примерно 0,35 и 0,65, соответственно).

Применение алгоритма кодирования Шеннона - Фано к русскому алфавиту дает избыточность кода 0,0147.

Префиксный код Хаффмена

Алгоритм создания неравномерного префиксного кода Хаффмена для тех же условий запишется следующим образом.

1. Расположить эти знаки в порядке убывания вероятностей.
2. Создать новый алфавит A_1, объединив два знака с наименьшими вероятностями их появления в текстах и заменив их одним знаком $a^{(1)}$ с вероятностью, равной сумме вероятностей объединяемых знаков.
3. Остальные знаки включить в новый алфавит без изменения.
4. Снова расположить знаки нового алфавита в порядке убывания вероятностей.
5. Продолжить создание новых алфавитов согласно п. 2 ÷ 4, пока в последнем не останется два знака.
6. Провести кодирование в обратном направлении. Для этого, двум знакам последнего алфавита присвоить коды: 0 - верхнему знаку и 1 – нижнему (как в методе Шеннона-Фано).
7. В предпоследнем алфавите, состоящем из трех знаков, одиночному знаку задать код знака последнего алфавита имеющего такую же вероятность; коды знаков, которые объединялись в один, сделать двузначными: их первой цифрой кода сделать код "родителя" – знака, имеющего объединенную вероятность, а второй цифрой: 0 для верхнего и 1 – для нижнего.
8. Повторять п.7 до тех пор, пока не будут определены коды всех символов первичного алфавита.

Рассмотрим работу алгоритма построения кодов Хаффмена для условий примера 3.1.

1. Расположим знаки алфавита в порядке убывания вероятностей.
2. Создадим новый алфавит $A^{(1),}$ объединив два знака с наименьшими вероятностями (a_5 и a_6) и заменив их одним знаком $a_5^{(1)}$ с вероятностью, равной сумме вероятностей a_5 и $a_{6.}$
3. Остальные знаки включим в новый алфавит без изменения. Общее число знаков в новом алфавите будет на 1 меньше, чем в исходном.
4. Упорядочим знаки в новом алфавите по убыванию вероятностей и продолжим создавать новые алфавиты согласно п. 2 ÷ 3, пока в последнем не останется два знака (таблица 3.2).

Таблица 3.2.

№ 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				

5. Проведем кодирование в обратном направлении. Двум знакам последнего алфавита присвоим коды 0 и 1 (верхний знак - 0, а нижний - 1). В примере знак $a_1^{(4)}$ алфавита $A^{(4)}$ имеющий вероятность 0,6 получит код 0, а $a_2^{(4)}$ с вероятностью 0,4 - код 1. В алфавите $A^{(3)}$ знак $a_1^{(3)}$ получает от $a_2^{(4)}$ его вероятность 0,4 и код 1; коды знаков $a_2^{(3)}$ и $a_3^{(3)}$, происходящие от знака $a_1^{(4)}$ с вероятностью 0,6 будут уже двузначным: их первой цифрой станет код их "родителя" (0), а вторая цифра - как мы условились - у верхнего 0, у нижнего - 1; таким образом, $a_2^{(3)}$ будет иметь код 00, а $a_3^{(3)}$ - код 01 и т.д. Процедура кодирования представлена в таблице 3.3.

Таблица 3.3.

№ 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								

Коды удовлетворяют условию Фано и, следовательно, не требуют разделителя. Средняя длина кода:

$K(A,2) = 0{,}3\times2 + 0{,}2\times2 + 0{,}2\times2 + 0{,}15\times3 + 0{,}1\times4 + 0{,}05\times4 = 2{,}45$.

Избыточность $Q(A,2) = 0{,}017$, однако, вероятности 0 и 1 сблизились (0,47 и 0,53, соответственно). Не удивляйтесь, что характеристики кода Хаффмена совпали с характеристиками кода Шеннона – Фано!

Первичный алфавит очень короткий, и полученные коды во многом похожи.

Более высокая эффективность кодов Хаффмена, по сравнению с кодами Шеннона – Фано, становится очевидной, если сравнить избыточности кодов для естественного языка. Применение описанного алгоритма для букв русского алфавита порождает коды, представленные в таблице 3.4.

Таблица 3.4.

Буква	**Код**	**p_i*10^3**	**$к_i$**	**Буква**	**Код**	**p_i*10^3**	**$к_i$**
пробел	000	174	3	я	001101	18	6
о	111	90	3	ы	010110	16	6
е	0100	72	4	з	010111	16	6
а	0110	62	4	ь,ъ	100001	14	6
и	0111	62	4	б	101100	14	6
т	1001	53	4	г	101101	13	6
н	1010	53	4	ч	110011	12	6
с	1101	45	4	й	0011001	10	7
р	00101	40	5	х	1000000	9	7
в	00111	38	5	ж	1000001	7	7
л	01010	35	5	ю	1100101	6	7
к	10001	28	5	ш	00110000	6	8
м	10111	26	5	ц	11001000	4	8
д	11000	25	5	щ	11001001	3	8
п	001000	23	6	э	001100010	3	9
у	001001	21	6	ф	001100011	2	9

Средняя длина кода оказывается равной $K(r,2)$ = 4,395; избыточность кода Q(r,2) = 0,0090, т.е. не превышает 1 %, что меньше избыточности кода Шеннона-Фано.

Код Хаффмена является *самым экономичным* из всех возможных, т.е. *ни для какого метода алфавитного кодирования длина кода не может оказаться меньше, чем код Хаффмена*.

Код Хаффмена широко применяется в алгоритмах, используемых программами-архиваторами, программами резервного копирования файлов и дисков, в системах сжатия информации в модемах и факсах.

3.2.2. Равномерное алфавитное двоичное кодирование. Байтовый код

Примером использования равномерного алфавитного кодирования является представление символьной информации в компьютере. Чтобы определить длину кода, необходимую для кодирования знака, нужно установить количество знаков в первичном алфавите. Компьютерный алфавит должен включать:

- 26x2 = 52 буквы латинского алфавита (с учетом прописных и строчных);
- 33x2 = 66 букв русского алфавита (кириллица);
- цифры 0...9 - всего 10;
- знаки математических операций, знаки препинания, спецсимволы ≈ 20.

Получаем, что общее число символов $N \approx 148$. Теперь можно оценить длину кодовой цепочки: $K(c,2) \geq \log_2 148 \geq 7{,}21$. Длина кода выражается целым числом, поэтому $K(c,2) = 8$. Такой способ кодирования принят в компьютерных системах: любому символу ставится в соответствие код из 8 двоичных разрядов (8 бит). Эта последовательность сохраняется и обрабатывается как единое целое (т.е. отсутствует доступ к отдельному биту) - по этой причине разрядность устройств компьютера, предназначенных для хранения или обработки информации, кратна 8. Совокупность восьми связанных бит получила название *байт,* а представление символов таким образом - *байтовым кодированием.*

Один байт соответствует количеству информации в одном знаке алфавита при их равновероятном распределении. Именно байт принят в качестве единицы измерения количества информации в международной системе единиц СИ.

Использование 8-битных цепочек позволяет закодировать $2^8=256$ символов, что превышает оцененное выше N и дает возможность употребить оставшуюся часть кодовой таблицы для представления дополнительных символов.

Но недостаточно только условиться об определенной длине кода. Для совместимости технических устройств и обеспечения возможности обмена информацией между многими потребителями требуется еще и согласование кодов. Подобное согласование осуществляется использованием *стандартов кодовых таблиц.*

В персональных компьютерах и телекоммуникационных сис-

темах применяется международный байтовый код ***ASCII*** (American Standard Code for Information - американский стандартный код обмена информацией). Стандартный набор символов ASCII использует только 7 битов для каждого символа. Он регламентирует коды первой половины кодовой таблицы (номера кодов от 0 до 127, первый бит всех кодов 0). В эту часть попадают коды прописных и строчных английских букв, цифры, знаки препинания и математических операций, а также некоторые управляющие коды (номера от 0 до 31), вырабатываемые при использовании клавиатуры.

Добавление 8-го разряда позволяет увеличить количество кодов таблицы ASCII до 255. Коды от 128 до 255 представляют собой расширение таблицы ASCII. Эти коды используются для кодирования символов национальных алфавитов, а также символов псевдографики, которые можно использовать, например, для оформления в тексте различных рамок и текстовых таблиц. Для этой части также имеются стандарты, например, для символов русского языка это КОИ-8, КОИ-7, Windows 1251 и др. В таблице 3.5 приведены *ASCII*- коды символов.

Таблица 3.5. ASCII - коды символов.

Основная таблица ASCII

	00	10	20	30	40	50	60	70
0		►		0	@	P	`	p
1	☺	◄	!	1	A	Q	a	q
2	☻	↕	"	2	B	R	b	r
3	♥	‼	#	3	C	S	c	s
4	♦	¶	$	4	D	T	d	t
5	♣	§	%	5	E	U	e	u
6	♠	▬	&	6	F	V	f	v
7	•	↨	'	7	G	W	g	w
8	◘	↑	(	8	H	X	h	x
9	○	↓	)	9	I	Y	i	y
A	◙	→	*	:	J	Z	j	z
B	♂	←	+	;	K	[	k	{
C	♀	∟	,	<	L	\	l	|
D	♪	↔	-	=	M	]	m	}
E	♫	▲	.	>	N	^	n	~
F	☼	▼	/	?	O	_	o	⌂

Расширенная таблица ASCII (cp866)

	80	90	A0	B0	C0	D0	E0	F0
0	А	Р	а	░	└	╨	р	≡
1	Б	С	б	▒	┴	╤	с	±
2	В	Т	в	▓	┬	╥	т	≥
3	Г	У	г	│	├	╙	у	≤
4	Д	Ф	д	┤	─	╘	ф	⌠
5	Е	Х	е	╡	┼	╒	х	⌡
6	Ж	Ц	ж	╢	╞	╓	ц	÷
7	З	Ч	з	╖	╟	╫	ч	≈
8	И	Ш	и	╕	╚	╪	ш	°
9	Й	Щ	й	╣	╔	┘	щ	∙
A	К	Ъ	к	║	╩	┌	ъ	·
B	Л	Ы	л	╗	╦	█	ы	√
C	М	Ь	м	╝	╠	▄	ь	ⁿ
D	Н	Э	н	╜	═	▌	э	²
E	О	Ю	о	╛	╬	▐	ю	■
F	П	Я	п	┐	╧	▀	я	

В таблице коды букв и цифр соответствуют их порядку следования в алфавите.

В операционной системе Windows имеется возможность вводить символы при помощи сочетания клавиш <Alt<+<X>, где X — десятичный код символа. Такой способ ввода символов, зачастую является единственно возможным, либо просто наилучшим.
Для ввода нужно удерживать нажатой клавишу <Al>t и, не отпуская ее, набрать десятичное число на цифровой клавиатуре.
Например, 1 с нажатым <Alt> введет смайлик ☺. 3 с нажатым <Alt> введет сердечко ♥ и т.д.

В настоящее время широко используется еще один международный стандарт кодировки - *Unicode.* В нем использовано 16 - битное кодирование, т.е. для каждого символа отводится 2 байта. Такая длина кода обеспечивает включения в первичный алфавит 65536 знаков. (Есть уже и Unicode с 32-битным кодированием). Это позволяет создать и использовать единую для всех распространенных алфавитов кодовую таблицу, но в этой системе все текстовые документы автоматически становятся вдвое длиннее.

Алфавитное кодирование с неравной длительностью элементарных сигналов

В качестве примера алфавитного кодирования с неравной длительностью сигнала рассмотрим *телеграфный код Морзе.* В нем каждой букве или цифре сопоставляется последовательность импульсов (точек и тире), разделяемых паузами. Под знаками кода Морзе следует понимать: "." – *"короткий импульс + короткая пауза",* "-" – *"длинный импульс + короткая пауза",* "0" – *"длинная пауза",* т.е. код оказывается *троичным.*

Свой код Морзе разработал в 1838 г., т.е. задолго до проведения исследований относительной частоты появления различных букв в текстах. Им был правильно выбран принцип кодирования - буквы, которые в текстах встречаются чаще, должны иметь более короткие коды, чтобы сократить общее время передачи. Относительные частоты букв английского алфавита он оценил простым подсчетом литер в ячейках типографской наборной машины. Самая распространенная английская буква "e" получила код *"точка".* При составлении кодов Морзе для букв русского алфавита учет относительной частоты букв не производился, что повысило его избыточность. Произведем оценку этой избыточности, используя таблицу 3.6. В ней, признак конца буквы (0) в кодах не отображается, но учтен в величине k_i - длине кода буквы *i.*

Таблица 3.6.

Буква	Код	p_i*10^3	$к_i$	Буква	Код	p_i*10^3	$к_i$
пробел	00	174	2	я	. - . -	18	5
о	- - -	90	4	ы	- . - -	16	5
е	.	72	2	з	- - .	16	4
а	. -	62	3	ь, ъ	- . . -	14	5
и	..	62	3	б	- . . .	14	5
т	-	53	2	г	- - .	13	4
н	- .	53	3	ч	- - - .	12	5
с	. . .	45	4	й	. - - -	10	5
р	. - .	40	4	х	. . . .	9	5
в	. - -	38	4	ж	. . . -	7	5
л	. - . .	35	5	ю	. . - -	6	5
к	- . -	28	4	ш	- - - -	6	5
м	- -	26	2	ц	- . - .	4	5
д	- . .	25	4	щ	- - . -	3	5
п	. - -	23	4	э	. . - . .	3	6
у	. . -	21	4	ф	. . - .	2	5

Среднее значение длины кода $K(r,3)$ = 3,361. Средняя информация на знак $I^{(2)} = \log_2 3 = 1{,}585$ бит.

Для русского алфавита $I_1^{(r)} = 4{,}356$ бит, поэтому избыточность равна:

$$Q(r,3) = 3{,}361 * 1{,}585/4{,}356 - 1 \approx 0{,}223,$$

то есть чуть больше 22%. Для английского языка она ≈ 19%.

Код Морзе широко применяется, когда источником и приемником сигналов является человек, и на первый план выдвигается удобство восприятия сигналов человеком.

3.2.3. Блочное двоичное кодирование

Как мы убедились, наилучший результат кодирования знаков получается с применением метода Хаффмена. Для русского алфавита избыточность кода оказалась в этом случае менее 1 %. Однако возможны варианты кодирования, при которых кодовый знак относится сразу к нескольким буквам или слову (блоку) первичного алфавита или языка. Кодирование блоков понижает избыточность. Докажем это следующим примером.

Пусть имеется словарь некоторого языка, содержащий n=16000 слов. Поставим в соответствие каждому слову равномерный

двоичный код. Длина кода может быть найдена из соотношения $K(A,2) \geq \log_2 16000 \geq 13{,}97 = 14$. Следовательно, каждое слово кодируется комбинацией из 14 нулей и единиц - получатся своего рода *двоичные иероглифы*. Например, пусть слову "мама" соответствует код *10101011100110,* слову "мыла" - *00000000000001,* а слову "раму" -*00100000000010;* тогда последовательность:

10101011100110 00000000000001 00100000000010

будет означать "мама мыла раму".

Средняя длина русского слова $K^{(r)}$= 6,3 буквы (5,3 буквы + пробел), тогда средняя информация на знак первичного алфавита оказывается равной $I^{(A)} = K(A,2)/K^{(r)} = 14/6{,}3 = 2{,}222$ бит, что значительно меньше, чем 5 бит, которые необходимы при равномерном алфавитном кодировании. Для английского языка такой метод кодирования дает 2,545 бит на знак. Таким образом, кодирование слов оказывается более выгодным, чем алфавитное.

Еще более эффективным окажется кодирование, если сначала установить относительную частоту появления различных слов в текстах и затем использовать код Хаффмена. Подобные исследования провел в свое время Шеннон: по относительным частотам 8727 наиболее употребительных в английском языке слов он установил, что средняя информация на знак первичного алфавита оказывается равной 2,15 бит.

Вместо слов можно кодировать сочетания букв - блоки. Блоки можно считать словами равной длины смыслового содержания. Удлиняя блоки и применяя код Хаффмена теоретически можно добиться того, что средняя информация на знак кода будет сколь угодно приближаться к I_∞. Блочное кодирование обеспечивает построение более оптимального кода, чем алфавитное. При использовании блоков большей длины (трехбуквенных и более) избыточность стремится к нулю в соответствии с первой теоремой Шеннона.

Пример. Пусть первичный алфавит состоит из двух знаков a и b с вероятностями 0,75 и 0,25. Нужно сравнить избыточность кода Хаффмена при алфавитном и блочном двухбуквенном кодировании.

При алфавитном кодировании:

Знак	$\mathbf{p_i}$	**Код**
a	0,75	0
b	0,25	1

$I(A) = 0{,}811$, $K(A,2) = 1$, $Q(A,2) = 0{,}233$

При блочном двухбуквенном кодировании этих знаков (считая, что $p_{ij} = p_i \cdot p_j$):

Знак	p_{ij}	**Код**
aa	0,562	0
ab	0,188	11
ba	0,188	100
bb	0,062	101

I(A) = 1,623 бит (или в пересчете на 1 знак - 0,811),
K(A,2) = 1,688симв. (или в пересчете на знак - 0,844),
Q*(A,2)* = 0,040.

Тема 4. Представление и обработка чисел в компьютере

4.1. Системы счисления

Система счисления – это совокупность приемов записи чисел с помощью цифр. Системы счисления бывают: *унарные, непозиционные* и *позиционные*.

Унарная - это система счисления, в которой для записи чисел используется только один знак - I ("палочка"). Следующее число получается из предыдущего добавлением новой I. Их количество равно самому числу. В унарной системе число представляется наиболее простым способом и операции с ним простые. Эта система определяет значение целого числа количеством содержащихся в нем единиц.

Примером непозиционной системы счисления является римская. В ней цифра, записанная в разных позициях числа имеет одно и то же значение, например: в числе XXVX – знак X везде обозначает десять.

В позиционных системах счисления вес каждой цифры зависит от ее положения в числе - позиции (разряда), например: в числе 777,7 первая семерка означает семь сотен, вторая – семь десятков, третья – семь единиц, а четвертая – семь десятых, то есть число можно записать в следующем виде:

$$777{,}7=7\times10^{2}+7\times10^{1}+7\times10^{0}+7\times10^{-1}.$$

Здесь оно представлено в виде степеней основания умноженных на коэффициенты, причем степени основания увеличиваются влево от запятой с шагом 1, начиная с нуля, а вправо от запятой они уменьшаются с шагом -1. Коэффициенты при степенях – это цифры данной системы счисления. Само же число представляет собой набор из этих коэффициентов, переписанных слева направо. Разложение такого вида справедливо для любой позиционной системы счисления.

Количество цифр, используемых в системе счисления для записи чисел, называется основанием системы – q. В вычислительной технике и информационных технологиях используются в основном двоичная, восьмеричная, десятичная и шестнадцатеричная системы. В них используются следующие цифры:

- двоичная (q=2): 0 и 1;
- восьмеричная (q=8): 0,1,2,3,4,5,6,7;
- десятичная (q=10): 0,1,…,9;
- шестнадцатеричная (q=16): 0,…,9,A, B, C, D, E, F.

Запись первых двух десятков десятичных чисел в этих системах

представим в виде таблицы 4.1.

Таблица 4.1.

Десятичная	**Двоичная**	**Восьмеричная**	**Шестнадцатеричная**
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

Для перевода чисел из одной позиционной системы счисления в другую используют следующие правила:

1. ***Общее правило.*** Число нужно записать в виде многочлена, слагаемые которого представляют собой степени основания, умноженные на некоторый коэффициент <q:

$$Z = a_{n-2} \cdot q^{n-2} + a_{n-1} \cdot q^{n-1} + \ldots + a_2 \cdot q^2 + a_1 \cdot q^1 + a_0 \cdot q^0 +$$
$$a_{-1} \cdot q^{-1} + a_{-2} \cdot q^{-2} + \ldots + a_m \cdot q^m,$$

где n – количество цифр в целой части числа, m – количество цифр в дробной части числа.

В новой системе счисления число запишется в виде последовательности из этих коэффициентов, перечисленных слева направо с учетом запятой, разделяющей целую и дробную части.

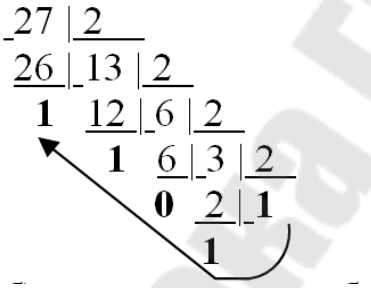
Примеры

$$35{,}6_8 = 3 \cdot 8^1 + 5 \cdot 8^0 + 6 \cdot 8^{-1} = 29{,}75$$

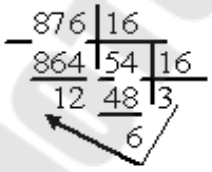
$$78_{10} = 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 +$$
$$+ 1 \cdot 2^1 + 0 \cdot 2^0 \rightarrow 1001110_2$$

2. ***Правило перевода целых чисел.*** Число нужно многократно делить нацело на основание новой системы, записывая остатки от деления. Деление нужно продолжать до тех пор, пока частное не станет меньше делителя. Число в новой системе записать в виде остатков деления, начиная с последнего частного (записывать в обратном порядке).

Примеры



$27_{10} = 11011_2$



$876_{10}=36C_{16}$.

3. ***Правило перевода дробей.*** Число нужно многократно умножать на основание новой системы, записывая целые части получающихся произведений. Умножать нужно до тех пор, пока не получится ноль или не будет достигнута требуемая точность. Дробь в новой

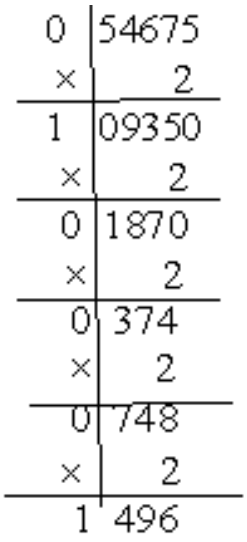
системе записать в виде целых частей произведений, начиная с первого.

Предельная абсолютная погрешность такого перевода равна

$$\Delta = \frac{q}{2}^{-(k+1)},$$

где k - количество знаков после запятой, полученное в процессе перевода.

Пример



$0{,}54675_{10}=0{,}10001_2$.

4. ***Правило перевода смешанных дробей.*** По правилам 2 и 3 перевести отдельно целую и дробную части и записать их рядом через запятую.
5. ***Правило перевода чисел из восьмеричной и шестнадцатеричную системы в двоичную.*** Заменить каждую цифру восьмеричного (шестнадцатеричного) числа соответствующей двоичной триадой (тетрадой).

Пример

а) перевести шестнадцатеричное число FA4,5BC в двоичную систему счисления;

б) перевести восьмеричное число 756,24 в в двоичную систему счисления:

Решение

а) 1111 1010 0100,0101 1011 1100;
F A 4 5 B C

б) 111101110,01011.
7 5 6 2 4

6. ***Правило перевода чисел из двоичной системы в восьмеричную (шестнадцатеричную).*** Разбить исходное число на триады (тетрады) влево и вправо от запятой. Неполные триады (тетрады) дополнить нулями, соответственно, слева или справа. Заменить каждую триаду (тетраду) соответствующей восьмеричной (шестнадцатеричной) цифрой.

Пример

Перевести двоичное число 111011100011,11011101:

a) в восьмеричную;
b) шестнадцатеричную систему.

Решение

a) /111/011/100/011, /110/111/010/;
b) 1110/1110/0011, /1101/1101/.

Ответы

a) $7343{,}672_8$;
b) $EE3{,}DD_{16}$.

Примечание. Деление на триады и тетрады показано с помощью символа "/".

4.2. Арифметические действия в позиционных системах счисления

Правила двоичной арифметики. Сложение и умножение в двоичной системе осуществляются в двоичной системе в соответствии со следующими правилами:

+	0	1
0	0	1
1	1	10

×	0	1
0	0	0
1	0	1

Если при сложении возникает переполнение разряда, то единица переносится в соседний левый разряд, а в сам разряд записывается нуль.

Операция умножения сводится к многократному поразрядному сдвигу с дальнейшим выполнением сложения.

Примеры действий в двоичной системе счисления

а) $10000000100_2 + 111000010_2 = 10111000110_2$

Действия	**Проверка**
10000000100	708
+ 111000010	+450
10111000110	1158

б) $1101_2 \times 11 = 100111_2$

Действия	**Проверка**
1101	13
× 11	× 3
1101	39
+1101	
100111	

в) $10010_2 - 101_2 = 1101_2$

Действия	**Проверка**
10010	17
— 101	- 5
01101	12

Действия в шестнадцатеричной системе производятся также как и в десятичной: сначала шестнадцатеричные цифры разряда переводим в десятичные, выполняем над ними необходимую операцию, а результат переводим шестнадцатеричную систему и записываем, делая, при необходимости, перенос в старший разряд.

Примеры действий в шестнадцатеричной системе счисления

```
  7AC93,F94          5A137, 024          873          155E38 |8
+ 9C7 8F,F89       - 1E495, 387        × 497          - 10   |2ABC7
 117 4 23,F1D        3BCA1,C9D          3B25            55
                                      + 4C0B          - 50
                                        21CC            5E
                                        26C7D5        - 58
                                                        63
                                                      - 60
                                                        38
                                                      - 38
                                                        0
```

Тема 5. Передача информации

5.1. Общая схема передачи информации в линии связи

Существует множество способов передачи информации: почта, телефон, радио, телевидение, компьютерные сети и пр. Однако в них можно выделить общие элементы (рисунок 5.1).

Рис. 5.1. Общая схема передачи информации

Источник информации порождает информацию и для передачи представляет ее в виде сообщения. Для представления информации он использует систему кодирования. *Кодирующее устройство* может являться подсистемой источника или внешним устройством по отношению к источнику информации.

Преобразователь переводит коды в последовательность материальных сигналов, т.е. помещает их на материальный носитель. Преобразователь может быть совмещен с кодирующим устройством (телеграфный аппарат), но может быть и самостоятельным элементом линии связи (например, модем). При преобразовании часть информации может теряться.

Например, полоса пропускания частот при телефонной связи от 300 до 3400 Гц, в то время как частоты, воспринимаемые человеческим ухом, лежат в интервале 16 - 20000 Гц; в черно-белом телевидении при преобразовании теряется цвет изображения.

В связи с этим встает задача выработки такого способа кодирования сообщения, который обеспечивал бы возможно более полное представление исходной информации при преобразовании и был бы согласован со скоростью передачи информации по каналу связи.

После преобразователя сигналы поступают и распространяются по *каналу связи.* Реальный канал связи подвержен внешним воздействиям, а также в нем могут происходить внутренние процессы, в результате которых искажаются передаваемые сигналы и связанное с ними сообщение. Такие воздействия называются *шумами* (помехами).

Внешние помехи – это "наводки" от мощных потребителей электричества или атмосферных явлений; одновременное действие нескольких близкорасположенных однотипных источников и т.п.

Внутренние помехи – к ним относятся физические неоднородности носителя; процессы затухания сигнала в линии связи из-за большой удаленности и др..

После прохождения сообщения по линии связи сигналы с помощью *преобразователя* переводятся в последовательность кодов, которые затем *декодируются* в форму, необходимую приемнику информации. На этапе приема преобразователь также может быть совмещен с декодирующим устройством (радиоприемник или телевизор) или существовать самостоятельно (модем).

Понятие *канал связи* объединяет все элементы от источника до приемника информации. Здесь мы будем рассматривать *дискретные* каналы связи, передача сообщений по которым осуществляется за счет электрических или световых импульсов (пример: компьютерные сети).

Дискретный канал – это канал связи, используемый для передачи дискретных сообщений. Рассмотрим его наиболее важные характеристики.

5.2. Характеристики канала связи

Ширина полосы пропускания

Любой преобразователь, работа которого основана на использовании колебаний, может формировать и пропускать сигналы из ограниченной области частот. Примерами являются: мобильная связь (GSM 900, 1800) и радио и телевизионная связь, где весь частотный спектр разделен на диапазоны (УКВ, ДМВ и т.д.), в пределах которых, каждая станция занимает свой диапазон.

Интервал частот, используемый каналом связи для передачи сигналов, называется *шириной полосы пропускания*.

Пропускная способность канала

С точки зрения теории важна не сама ширина полосы пропускания, а максимальное значение этой частоты из заданной полосы (ν_m).

Если импульсы порождаются тактовым генератором, имеющим частоту ν_m, то длительность элементарного импульса будет равна

$$\tau_0 = \frac{1}{\nu_m} \qquad (5.1)$$
(5.1)

Таким образом, каждые τ_0 секунд можно передавать импульс или паузу, связывая их последовательно в определенные коды. Использование сигналов большей длительности, чем τ_0 (например, $2\cdot\tau_0$) не приведет к потере информации, однако снизит скорость передачи. Использование сигналов более коротких, чем τ_0, может привести к информационным потерям, поскольку в этом случае сигналы будут принимать промежуточные значения между минимальным значением (0) и максимальным (1), что затруднит их интерпретацию.

Если с передачей одного импульса связано количество информации I_{imp}, а передается оно за время τ_0, то отношение I_{imp} к τ_0 будет отражать *среднее количество информации, передаваемое по каналу за единицу времени* - эта величина является характеристикой канала связи и называется ***пропускной способностью канала***:

$$C = \frac{I_{imp}}{\tau_0} \qquad (5.2)$$

Если I_{imp} выражено в битах, а τ_0 - в секундах, то единицей измерения C будет *бит/с*. Эту единицу связисты назвали специальным термином – *бод*.

Величину I_{imp} в формуле (5.2) можно установить из следующих соображений: если первичный алфавит содержит N знаков с вероятностями появления их в сообщении p_i, то по формуле Шеннона (2.7) можно найти среднюю информацию на знак первичного алфавита I_1, для представления которого используется двоичный код длиной $K^{(2)}$. Тогда

$$I_{imp} = \frac{I_1}{K^{(2)}}$$

При подстановке этого выражения в (5.2) получаем:

$$C = \frac{I_1}{K^{(2)} \cdot \tau_0} \qquad (5.3)$$

Пример 5.1. Первичный алфавит состоит из трех знаков с вероятностями $p_1 = 0{,}2$; $p_2 = 0{,}7$; $p_3 = 0{,}1$. Для передачи по каналу без помех используются равномерный двоичный код. Частота тактового генератора 500 Гц. Какова пропускная способность канала?

Число знаков первичного алфавита $N = 3$. Из формулы (2.10) получаем:

$$I_1 = - 0{,}2 \cdot \log_2 0{,}2 - 0{,}7 \cdot \log_2 0{,}7 - 0{,}1 \cdot \log_2 0{,}1 = 1{,}16 \text{ бит},$$

$$K^{(2)} \geq \log_2 N = 2.$$

Следовательно, из формулы (5.3) получаем:

$$C = \frac{I_1}{K^{(2)} \cdot \tau_0} = \frac{\nu \cdot I_1}{K^{(2)}} = \frac{500 \cdot 1{,}16}{2} = 290 \; бит/с$$

Скорость передачи информации

Пусть по каналу связи за время t передано количество информации I. Введем величину, характеризующую быстроту передачи информации - *скорость передачи информации J:*

$$J = \frac{I}{t} \qquad (5.4)$$

Размерностью J, как и C, является также бит/с. Максимальной скорости передачи информации по данной линии связи можно достигнуть, ***когда она равна максимальной пропускной способности канала,*** то есть, когда $J_{max} = C$; в остальных случаях $J \leq C$. Таким образом, **максимальная скорость передачи информации по каналу связи равна его пропускной способности.**

5.3. Влияние шумов на пропускную способность канала

Отличие реального канала от идеального в том, что шумы приводят к *снижению пропускной способности канала.*

Рассмотрим использование двух элементарных сигналов равной длительности: 1(импульс) и 0 (пауза). Каждый из них на входе канала связи несет $I_{imp} = 1$ бит информации. После прохождения сигнала по идеальному каналу и на выходе импульс (1) интерпретируется как импульс, а пауза (0) - как пауза. Связанная с ними информация не меняется также и в количественном отношении.

В реальном канале из-за шумов при передаче может произойти

искажение сигнала, в результате чего вместо 1 на выходе будет получен 0, а вместо 0 - 1. Пусть вероятности таких искажений для 0 и 1 одинаковы и равны *p*. Тогда вероятность того, что исходный сигнал придет без искажений равна 1-*p*. Следовательно, при распознавании конечного сигнала возникает *неопределенность,* которая, как следует из формулы (2.6), может быть охарактеризована средней энтропией:

$$H=-p\cdot log_2 p-(1-p)\cdot log_2(1-p)$$

Эта неопределенность приведет к *уменьшению* количества информации, содержащейся в сигнале, на величину *H,* то есть оно станет равным:

$$I_{imp}'=I_{imp}-H$$

Поскольку длительность импульса τ_0 определяется частотой ν_m и не зависит от наличия шумов, пропускная способность реального канала C_R оказывается меньше, чем у аналогичного идеального *C:*

$$C_R=\frac{I_{imp}'}{\tau_0}=\frac{I_{imp}+p\cdot\log_2 p+(1-p)\cdot\log_2(1-p)}{\tau_0}\le C \qquad (5.5)$$

На графике показана зависимость $C_R(p)$.

Как следует из формулы (5.5), при p = 0,5 $I_{imp}{}' = 0$ и, следовательно, $C_R = 0$. Это связано с тем, что в данном случае на приемном конце линии связи с одинаковой вероятностью получаются 0 и 1 независимо от того, каков был сигнал на входе, так что передача информации по такой линии оказывается вообще невозможной. Максимального значения пропускная способность достигает при p близких к нулю, что соответствует отсутствию помех, а также при p близких к единице, т.е. таких помехах, которые каждый входной сигнал 1 переводят в 0 на выходе, а каждый 0 на входе - в 1 на выходе. Помехи такого рода не мешают распознаванию того, какой сигнал был послан и пропускная способность от этого не страдает. В остальных случаях $C < C_R$.

Пример 5.2. Каково отношение C_R/C, если средняя частота появления ошибки при передаче сообщения по каналу связи с шумом составляет 1 ошибочный знак на 100 переданных, и при этом используется двоичное кодирование?

Вероятность появления ошибки передачи p = 1/100; поскольку код двоичный, I_{imp} = 1 бит. Следовательно, из формул (5.3) и (5.5) получаем:

$$\frac{C_R}{C} = 1 + p \cdot \log_2 p + (1-p) \cdot \log_2 (1-p) = 0.9192$$

т.е. пропускная способность канала снизилась приблизительно на 8%.

5.4. Обеспечение надежности передачи и хранения информации

5.4.1. Постановка задачи

На практике все реальные каналы связи подвержены воздействию помех, но Шеннон доказал *теоретическую возможность* передачи сообщения без потерь информации по реальным каналам, если при этом выполняются определенные условия.

Вторая теорема Шеннона: *При передаче информации по каналу с шумом всегда найдется способ кодирования, при котором сообщение будет передаваться со сколь угодно высокой достоверностью, если скорость передачи не превышает пропускной способности канала*.

Смысл теоремы в том, что при передаче по реальным каналам можно закодировать сообщение таким образом, что действие шумов не приведет к потере информации. Это достигается за счет повышения избыточности кода. При этом возрастает время передачи, что следует считать платой за надежность.

Таким образом, нужно использовать такие методы кодирования информации, которые позволили бы контролировать правильность передачи и при обнаружении ошибки исправлять ее.

К решению проблемы возможны два подхода:

- кодирование обеспечивает *только установление факта* искажения информации - в этом случае исправление производится путем ее повторной передачи;
- кодирование позволяет *определить* и *автоматически исправить* ошибку передачи.

Общим условием для второго подхода является использование *только равномерных кодов.* Надежность обеспечивается тем, что с битами, кодирующими сообщение (*информационными битами),* передаются дополнительные биты (*контрольные*), по состоянию которых можно судить о правильности передачи. При *равномерном кодировании* сообщения длина кодовой цепочки на знак k складывается из длины информационной части k_i, и числа контрольных битов k_c. Очевидно, $k \geq k_i$. Подобно введенной ранее в (3.4) величине Q, характеризующей относительную избыточность кода, определим *избыточность сообщения* для реального канала L по отношению к идеальному следующим образом:

$$L = \frac{k}{k_i} = \frac{k_i + k_c}{k_i} = 1 + \frac{k_c}{k_i} \qquad (5.6)$$

Относительная избыточность сообщения - это характеристика, показывающая, во сколько раз требуется удлинить сообщение, чтобы обеспечить его надежную передачу (хранение).

L характеризует эффективность кодирования при передаче информации по реальным каналам. При равных надежностях передачи предпочтение должно быть отдано тому способу кодирования, при котором избыточность окажется наименьшей.

Определим, какой должна быть минимальная избыточность сообщения для того, чтобы при искажении информации ее можно было

бы полностью восстановить. Учитывая, что шумы в канале связи ведут к частичной потере передаваемой информации вследствие возникающей неопределенности величиной H, определяемой по формуле (2.4), определяем, что вместо передачи каждого 1 бит информации следует передавать $1+H$ бит. Тогда нужная минимальная избыточность сообщения составит:

$$L_{\min} = \frac{1+H}{1} = 1 - p \cdot \log_2 p - (1-p) \cdot \log_2 (1-p) \quad (5.7)$$

Пример 5.3. Какое минимальное количество контрольных бит должно передаваться вместе с 16-ю информационными битами для обеспечения восстановимости информации, если вероятность искажения составляет 1%?

Решение

Подставляя p=0,01 в формулу (5.7), находим L_{min}=1,081.

При k_i=16 из формулы (5.6) получаем:

$k=k_i \times L_{min}$=17,29 бит.

С учетом того, что количество контрольных бит выражается целым числом, получаем k=18, откуда k_c=2.

Тогда реальная избыточность, рассчитанная по формуле (5.6) составит L=1,125.

5.4.2. Коды, обнаруживающие ошибку

Задача *обнаружения* ошибки может быть решена довольно легко. Достаточно передавать каждую букву сообщения дважды. Например, слова *"гора"* можно передать *"ггоорраа"*. При получении сообщения *"гготрраа"* можно догадаться, каким было исходное слово. Но возможно искажение, которое делает неоднозначным интерпретацию полученного сообщения, например, *"гпоорраа"*. Цель такого способа кодирования состоит не в исправлении ошибки, а в фиксации факта искажения и повторной передаче части сообщения. Недостаток данного способа обеспечения надежности состоит в том, что избыточность сообщения оказывается очень большой, *L = 2*.

Для обнаружения ошибки можно использовать другой способ

кодирования. Пусть имеется цепочка информационных бит длиной k_i. Добавим к ним один контрольный бит k_c, значение которого определяется тем, что новая кодовая цепочка из k_i+1 бит должна содержать *четное количество единиц (вместе с битом четности!)* - по этой причине такой контрольный бит называется *битом четности.* Например, для информационного байта

01010100

бит четности будет иметь значение 1, а для байта

11011011

бит четности равен 0. В случае одиночной ошибки передачи число 1 перестает быть четным, что и служит свидетельством сбоя. Например, если получена цепочка 110110111 (контрольный бит выделен подчеркиванием), ясно, что передача произведена с ошибкой, поскольку общее количество единиц равно 7. Этот способ кодирования не позволяет установить, в каком конкретно бите содержится ошибка и не дает возможности ее исправить. Избыточность сообщения при этом, исходя из формулы (5.6) равна:

$$L = \frac{k_i + 1}{k_i} = 1 + \frac{1}{k_i}$$

На первый взгляд кажется, что путем увеличения k_i, можно сколь угодно приближать избыточность к ее минимальному значению (L_{min} = 1). Однако с ростом k_i растет вероятность парной ошибки, которая контрольным битом не отслеживается, при обнаружении ошибки потребуется заново передавать много информации. Поэтому обычно k_i=8 или 16 и L = 1,125 (1,0625).

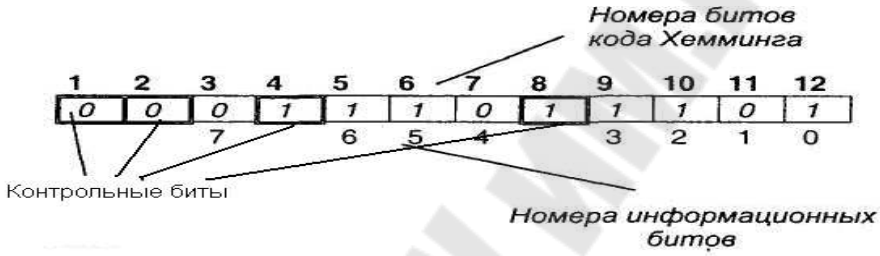
5.4.3. Коды, исправляющие одиночную ошибку

По аналогии с предыдущим пунктом можно было бы предложить простой способ установления ошибки - передавать каждый символ трижды, например, *"гггоооррраааа"* - тогда при получении сообщения *"гггооопрааа"* ясно, что ошибочной оказывается буква "п" и ее следует заменить на "р". Безусловно, при этом предполагается, что вероятность появления парной ошибки невелика. Такой метод кодирования приводит к избыточности сообщения L = 3, что неприемлемо с экономической точки зрения.

Рассмотрим метод кодирования позволяющий определить, в ка-

ком бите находится ошибка. Метод был предложен в 1948 г. Р. Хеммингом. Поэтому коды, построенные таким способом, получили название *коды Хемминга*.

Идея состоит в том, что к информационным битам добавляется не один, а сразу несколько битов четности, причем каждый из них контролирует не все, а только определенные информационные биты. Если пронумеровать все передаваемые биты в цепочке, начиная с единицы *слева направо* (*а информационные биты нумеруются с нуля и наоборот, справа налево*), то контрольными оказываются биты, номера которых равны степеням числа 2, а все остальные являются информационными. Например, для 8-битного информационного кода контрольными окажутся биты с номерами 1, 2, 4 и 8:



Номера контролируемых битов для каждого контрольного приведены в табл. 5.1. В перечень контролируемых битов входят и сами контрольные. Перед передачей цепочки состояние каждого контрольного бита устанавливается таким образом, чтобы *суммарное количество единиц во всех контролируемых им битах, включая его самого, было бы четным.*

Таблица 5.1

Контрольные биты	**Контролируемые биты**											
1	1	3	5	7	9	11	13	15	17	19	21	…
2	2	3	6	7	10	11	14	15	18	19	22	…
4	4	5	6	7	12	13	14	15	20	21	22	…
8	8	9	10	11	12	13	14	15	24	25	26	…
16	16	17	18	19	20	21	22	23	24	25	26	…
32	32	33	34	35	36	37	38	39	40	41	42	…

Нет смысла помнить эту таблицу, гораздо проще понять и запомнить правило, по которому она получена.

Правило: для любого номера контрольного бита *n*, начиная с него, *n* бит подряд являются проверяемыми, затем следует группа из *n* непроверяемых бит; далее происходит чередование групп.

Пример 5.4. Получено машинное слово, закодированное с использованием кода Хемминга: 101111110001. Требуется найти и исправить ошибку передачи.

Решение: Запишем поступившее машинное слово побитно

1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	1	1	1	1	1	0	0	0	1
		7			6	5	4		3	2	1	0

Проверяем состояние контрольных битов (они выделены жирной рамкой):

Бит 1. (Контролирует нечетные биты: 1, 3, 5, 7, 9, 11). **Верно** (так как сумма контролируемых бит =4 - четная). Следовательно, ошибки в нечетных битах нет.

Бит 2. (Контролирует биты: 2, 3, 6, 7, 10, 11). **Неверно** (так сумма контролируемых бит =3 - нечетная). С учетом предыдущей проверки следует, что ошибка возможна во 2, 6 или 10 битах.

Бит4. (Контролирует биты: 4, 5, 6, 7, 12). **Неверно** (так сумма контролируемых бит =5 - нечетная)**.** С учетом предыдущего следует, что ошибка в 6 бите. Хотя ошибочный бит уже определен, сделаем проверку и для последнего контрольного бита.

Бит 8. (Контролирует биты: 8, 9, 10, 11, 12). **Верно** (так как сумма контролируемых бит =2 - четная)**.** Следовательно, то, что ошибка в 6 бите подтверждается. На ее существование указали контрольные коды 2 и 4. Их сумма равна 2+4=6, т.е. номеру бита, содержащего ошибку, что является *общим свойством кодов Хемминга*.

Содержимое шестого информационного бита нужно инвертировать (заменить 1 на 0) и, тем самым, восстановить правильную последовательность.

***Алгоритм* проверки и исправления передаваемой последовательности бит по методу Хемминга:**

1. произвести проверку всех битов четности;
2. если все биты четности верны, то перейти к п.5;
3. вычислить сумму номеров всех неправильных битов четно-

сти;

4. инвертировать содержимое бита, номер которого равен сумме, найденной в п.3;
5. исключить биты четности, принять правильный информационный код.

Избыточность кодов Хемминга для различных длин передаваемых последовательностей приведена в таблице 5.6:

Таблица 5.6.

Число информационных битов	**Число контрольных битов**	**Избыточность, *L***
8	4	1,50
16	5	1,31
32	6	1,06

Из таблицы видно, что выгоднее передавать и хранить более длинные последовательности битов. При этом избыточность не может оказаться меньше L_{min} для выбранного канала связи.

Пример 5.5. Получена цепочка из пяти символьных сообщений, длиной в один байт каждое (в ASCII – кодировке), закодированных кодом Хемминга (столбцы с контрольными битами выделены красным цветом).

	1	2	3	4	5	6	7	8	9	10	11	12
1)	1	0	1	0	1	0	0	1	0	0	0	1
2)	1	0	0	0	1	0	1	1	1	0	1	0
3)	1	0	0	0	1	0	1	1	0	0	1	0
4)	1	0	0	0	1	1	0	1	0	0	0	1
5)	1	1	0	1	1	0	1	0	1	0	1	1

Требуется его декодировать с учетом возможных одиночных ошибок в каждом информационном байте.

Решение

Применим алгоритм Хемминга для поиска одиночных ошибок в каждом из пяти сообщений. Для удобства, результаты подсчета оформим в виде таблицы, в которой знак "-" говорит о том, что контрольный бит указывает на наличие ошибки, а знак "+" – на ее отсутствие.

№ п/п	Контрольные биты				Σ	№ ошибочного бита
	1	2	4	8		
1.	-	-	+	+	3	3
2.	-	+	+	-	9	9
3.	+	+	+	+		
4.	+	-	-	+	6	6
5.	-	-	+	-	11	11

Таким образом, исправленное сообщение будет иметь вид:

1	2	3	4	5	6	7	8	9	10	11	12
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	0	1	0	1	1	0	0	1	0
1	0	0	0	1	0	1	1	0	0	1	0
1	0	0	0	1	0	0	1	0	0	0	1
1	1	0	1	1	0	1	0	1	0	0	1

Формируем цепочки с исправленными информационными битами, убрав контрольные.

№ п/п	Информационные биты								16-тиричный код	Символ
	7	6	5	4	3	2	1	0		
1	0	1	0	0	0	0	0	1	41	A
2	0	1	0	1	0	0	1	0	52	R
3	0	1	0	1	0	0	1	0	52	R
4	0	1	0	0	0	0	0	1	41	A
5	0	1	0	1	1	0	0	1	59	Y

Коды символов определяем согласно таблице кодов ASCII (Таб-

лица 3.5) (ниже приведена ее нужная нам часть – заглавные латинские буквы):

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51

R	S	T	U	V	W	X	Y	Z
52	53	54	55	56	57	58	59	5A

Ответ: получено слово ARRAY

Пример 5.5. Обратная задача: подготовить к отправке информационную восьмибитную цепочку (букву W): 01010111, зашифровав ее кодом Хемминга.

Решение

Сделаем заготовку для отправляемой информационной цепочки с учетом добавляемых четырех контрольных бит, и впишем в нее известные нам информационные.

1	2	3	4	5	6	7	8	9	10	11	12
		0		1	0	1		0	1	1	1
		7		6	5	4		3	2	1	0

Значения контрольных бит определим путем следующих рассуждений.

Бит 1. Его значение задаем равным **1,** чтобы вместе с ним в нечетных битах количество единиц было бы четным (= 4).

Бит 2. Его значение задаем равным **1,** чтобы вместе с ним в битах: 2, 3, 6, 7, 10, 11 количество единиц было бы четным (= 4)

Бит 4. Его значение задаем равным **1,** чтобы вместе с ним в битах: 4, 5, 6, 7, 12 количество единиц было бы четным (=4).

Бит 8. Его значение задаем равным **1,** чтобы вместе с ним в битах: 8, 9, 10,11,12 количество единиц было бы четным (= 4).

Вписываем полученные значения в соответствующие биты. Цепочка к отправке готова.

1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	1	1	0	1	1	0	1	1	1
		7		6	5	4		3	2	1	0

Тема 6. Технологии защиты данных. Криптографическая защита информации

6.1 Основные понятия

Информация, которая передается по каналам связи или хранится на каком-либо носителе, может быть перехвачена злоумышленником или скопирована с носителя. Чтобы исключить потери, связанные с кражей важной информации (например, банковской), ее нужно зашифровать. Методологической основой современных систем обеспечения безопасности информации в компьютерных системах и сетях является *криптография*.

Криптография (с греческого – тайнопись) зародилась как способ скрытой передачи сообщений и представляет собой совокупность методов преобразования данных, направленных на то, чтобы защитить эти данные, сделав их бесполезными для незаконных пользователей.

Для обеспечения безопасности данных необходимо поддерживать три основные функции:

- защиту конфиденциальности передаваемых и хранимых в памяти данных;
- подтверждение целостности и подлинности данных;
- аутентификацию абонентов при входе систему и при установлении соединения.

Для реализации указанных функций используются криптографические технологии шифрования, цифровой подписи и аутентификации.

Основой большинства криптографических средств защиты информации является шифрование данных.

Под шифром понимают совокупность процедур и правил криптографических преобразований, используемых для зашифрования и расшифрования информации по ключу шифрования. Под зашифрованием информации понимается процесс преобразования открытой информации (исходного текста) в зашифрованный текст (шифротекст). Обратный процесс называют расшифрованием (дешифрованием).

Рассмотрим схему криптосистемы шифрования (рисунок 6.1.)

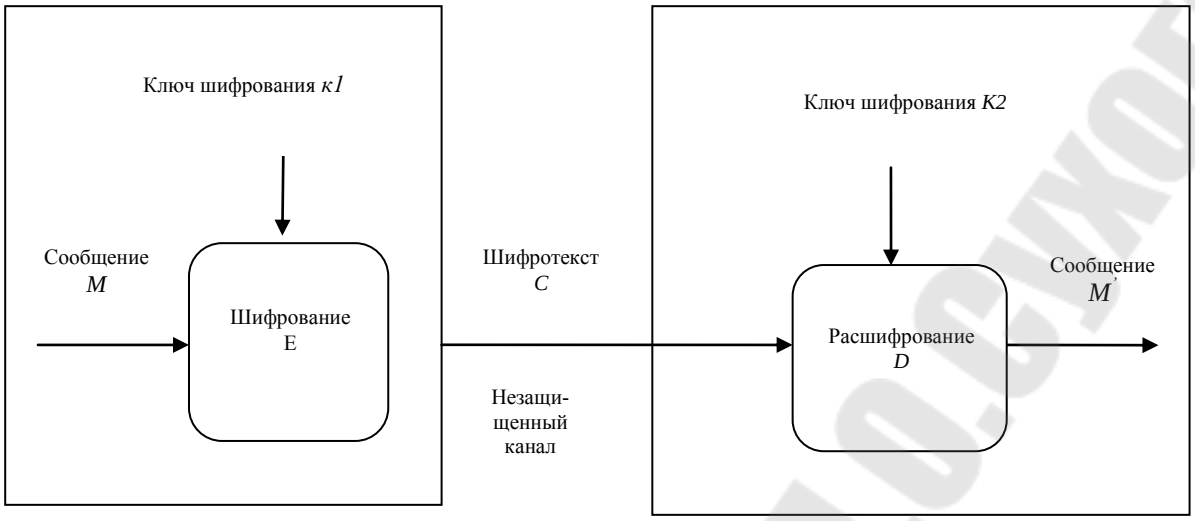


Рисунок 6.1 – Обобщенная схема криптошифрования

Исходный текст передаваемого сообщения (или хранимой информации) *M* зашифровывается с помощью криптографического преобразования E_{k1} с получением в результате *шифротекста C*:

$$C=E_{k1}(M),$$

где*:* *k1* – параметр функции *E*, называемый ключом шифрования.

Шифротекст *C*, называемый еще *криптограммой*, содержит исходную информацию *M* в полном объеме, однако последовательность знаков в нем внешне представляется случайной и не позволяет восстановить исходную информацию без знания ключа шифрования *к1*. Зашифрованная с использованием конкретного ключа информация может быть расшифрована только его владельцем (или владельцами).

Обратное преобразование информации выглядит следующим образом:

$$M'=D_{k2}(C).$$

Функция *D* является обратной к функции *E* и производит расшифрование шифротекста. Она также имеет дополнительный параметр в виде ключа *к2*. Ключ расшифрования *к2* должен однозначно соответствовать ключу *к1*, в этом случае полученное в результате расшифрования сообщение *M′* будет эквивалентно *M*. При отсутствии

верного ключа к получить исходное сообщение $M'=M$ с помощью функции D невозможно.

Преобразование шифрования может быть симметричным или асимметричным относительно преобразования расшифрования. Соответственно, различают два основных класса криптосистем:

- симметричные криптосистемы;
- ассиметричные криптосистемы.

Известно несколько классификаций криптографических алгоритмов. Одна из них подразделяет КА в зависимости от числа ключей, применяемых в конкретном алгоритме:

- бесключевые КА – не используют в вычислениях никаких ключей;
- одноключевые КА – работают с одним ключевым параметром;
- двухключевые КА – на различных стадиях работы в них применяются два ключевых параметра: секретный и открытый ключи.

Более детальная классификация представлена на рисунке 6.2.

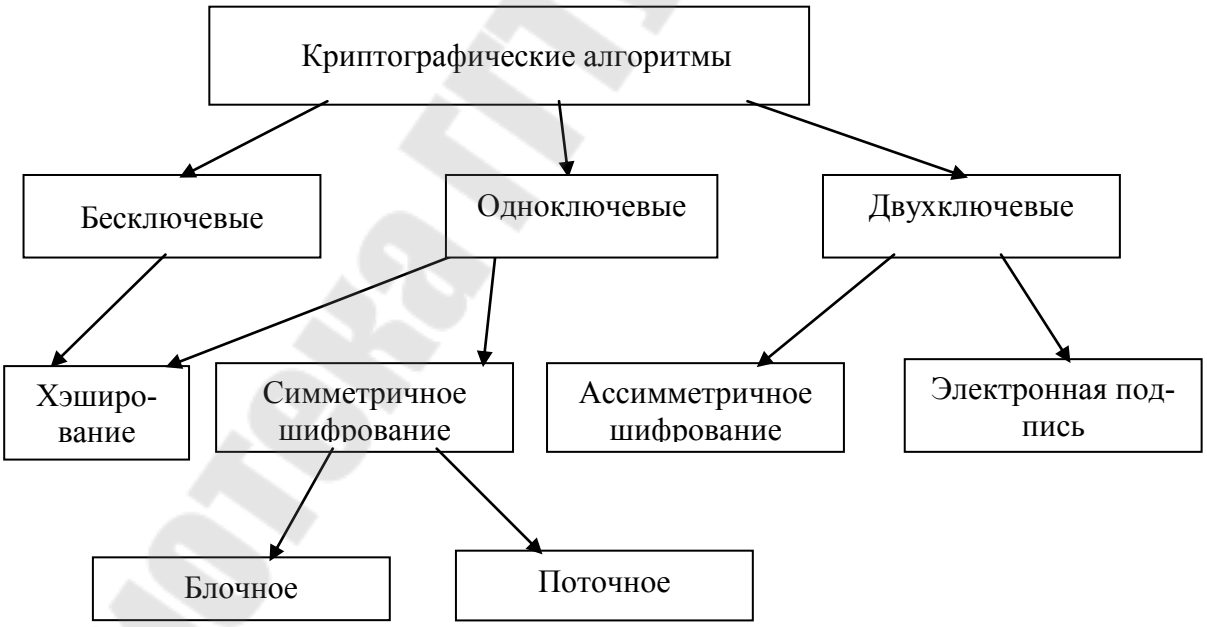


Рисунок 6.2 - Классификация криптографических алгоритмов

Хэширование – это метод криптозащиты, представляющий собой контрольное преобразование информации; из данных неограни-

ченного размера путем выполнения криптографических преобразований вычисляется хэш-значение фиксированной длины, однозначно соответствующее исходным данным. Хэширование может выполняться как с использованием некоторого секретного ключа, так и без него.

Такое криптографическое контрольное суммирование широко используется в различных методах защиты информации, в частности, для подтверждения целостности данных, если применение электронной подписи невозможно или избыточно.

Симметричное шифрование использует один и тот же ключ как для зашифрования, так и для расшифрования информации. Симметричное шифрование подразделяется на два вида: блочное и поточное, хотя это разделение – чисто условное.

Блочное шифрование характеризуется тем, что информация предварительно разбивается на блоки фиксированной длины (например, 64 или 128 бит). При этом блоки могут шифроваться как независимо друг от друга, так и со "сцеплением" – когда результат шифрования текущего блока данных зависит от значения предыдущего блока или результата шифрования предыдущего блока.

Поточное шифрование применяется прежде всего тогда, когда информацию невозможно разбить на блоки. Алгоритмы поточного шифрования шифруют данные побитно или посимвольно.

Асиметричное шифрование характеризуется применением двух типов ключей: открытого – для зашифрования информации, и секретного – для ее расшифрования. Оба ключа связаны между собой достаточно сложным соотношением. Главное в этом соотношении – легкость вычисления открытого ключа из секретного и невозможность (за ограниченное время при ограниченных реальных ресурсах) вычисления секретного ключа из открытого.

Электронная цифровая подпись (ЭЦП) используется для подтверждения целостности и авторства данных. Как и в случае асимметричного шифрования, в данном методе применяются двухключевые алгоритмы с таким же простым вычислением открытого ключа из секретного и практически невозможностью обратного вычисления. Однако назначение ключей ЭЦП совершенно иное. Секретный ключ применяется для вычисления ЭЦП, а открытый ключ необходим для ее проверки. При соблюдении правил безопасного хранения секретного ключа никто, кроме его владельца, не в состоянии вычислить верную ЭЦП какого-либо электронного документа.

6.2. Симметричные криптосистемы шифрования

Такие системы исторически появились первыми. В них используется один и тот же ключ для зашифрования и расшифрования информации. Это означает, что любой, кто имеет доступ к ключу шифрования, может расшифровать сообщение. Поэтому все ключи шифрования в симметричных криптосистемах должны содержаться в секрете. Схема симметричной криптосистемы шифрования выглядит следующим образом (рисунок 6.3).

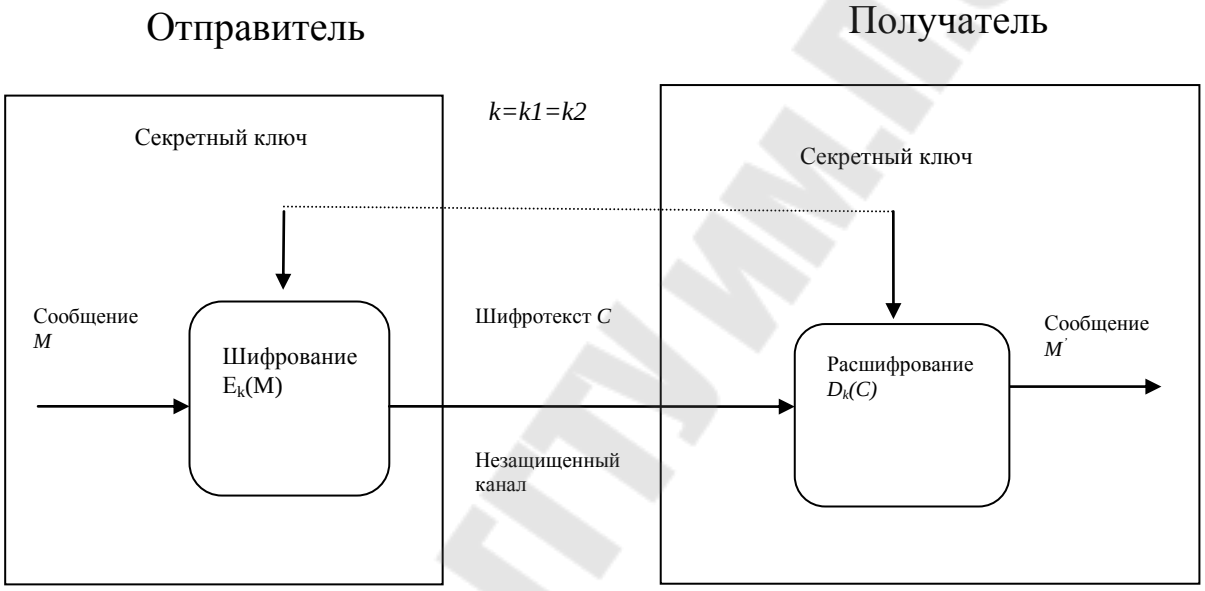


Рисунок 6.3 - Схема симметричной криптосистемы шифрования

Конфиденциальность передачи информации в таких системах зависит от надежности шифра и обеспечения конфиденциальности ключа шифрования. Обычно ключ шифрования представляет собой файл или массив данных и хранится на персональном ключевом носителе, например дискете или флэш-карте; обязательно принятие мер, обеспечивающих недоступность персонального ключевого носителя кому-либо, кроме его владельца.

Подлинность обеспечивается за счет того, что без предварительного расшифрования практически невозможно осуществить смысловую модификацию и подлог криптографически закрытого сообщения. Фальшивое сообщение не может быть правильно зашифровано без знания секретного ключа.

Целостность данных обеспечивается присоединением к передаваемым данным специального кода (имитовставки), вырабатываемого по секретному ключу. *Имитовставка* является разновидностью контрольной суммы, то есть некоторой эталонной характеристикой сообщения, по которой осуществляется проверка целостности последнего. Если выработанное по секретному ключу значение имитовставки для полученного сообщения совпадает с полученным значением имитовставки, то делается вывод, что информация на пути от отправителя к получателю не была модифицирована. Симметричное шифрование идеально подходит в случае шифрования информации "для себя", например, с целью предотвратить несанкционированный доступ к ней в отсутствии владельца.

Обладая высокой скоростью шифрования, одноключевые криптосистемы позволяют решать многие важные задачи информации. Однако автономное использование симметричных криптосистем в компьютерных сетях порождает проблему распределения ключей шифрования между пользователями.

Перед началом обмена зашифрованными данными необходимо обменяться секретными ключами со всеми адресатами. Передача секретного ключа симметричной криптосистемы не может быть осуществлена по общедоступным каналам связи, секретный ключ нужно передавать отправителю и получателю по *защищенному каналу*.

Существуют реализации алгоритмов симметричного шифрования для абонентского шифрования данных – то есть для отправки шифрованной информации абоненту, например, через Интернет. Использование одного ключа для всех абонентов подобной криптографической сети недопустимо по соображениям безопасности, так как в случае компрометации (утери, хищения) ключа под угрозой будет находиться документооборот всех абонентов. В этом случае может быть использована матрица ключей.

Матрица ключей представляет собой таблицу, содержащую ключи парной связи абонентов. Каждый элемент таблицы K_{ij} предназначен для связи абонентов i и j и доступен только им. Соответственно, для всех элементов матрицы ключей соблюдается равенство $K_{ij} = K_{ji}$. Матрица ключей для всех абонентов выглядит следующим образом.

K_{11}	K_{12}	K_{13}	...	K_{1n}

Набор ключей для абонента 1

K_{21}	K_{22}	K_{23}	…	K_{2n}
K_{31}	K_{32}	K_{33}	…	K_{3n}
…	…	…	…	…
K_{n1}	K_{n2}	K_{n3}	…	K_{nn}

Набор ключей для абонента 2

Набор ключей для абонента 3

…

Набор ключей для абонента *n*

Каждая *i*-я строка матрицы представляет собой набор ключей конкретного абонента *i* для связи с остальными *n*-1 абонентами. Наборы ключей распределяются между всеми абонентами криптографической сети и должны распределяться по защищенным каналам связи или из рук в руки.

Характерной особенностью симметричных криптоалгоритмов является то, что в ходе своей работы они производят преобразование блока входной информации фиксированной длины и получают результирующий блок того же объема, но недоступный для прочтения сторонним лицам, не владеющим ключом. Схему работы симметричного блочного шифра можно описать функциями:

$$C=E_K(M) \text{ и } M=D_K(C),$$

где: M – исходный (открытый) блок данных,

C – зашифрованный блок данных.

Блочные шрифты являются той основой, на которой реализованы практически все симметричные криптоалгоритмы. Симметричные криптоалгоритмы позволяют кодировать и декодировать файлы произвольной длины. На сегодняшний день разработано достаточно много стойких блочных шифров.

Криптоалгоритм считается идеально стойким, если для прочтения зашифрованного блока данных необходим перебор всех возможных ключей до тех пор, пока расшифрованное сообщение не окажется осмысленным. В общем случае стойкость блочного ключа зависит только от длины ключа и возрастает экспоненциально с ее ростом.

К. Шеннон предложил для получения стойких блочных шрифтов использовать два принципа: *рассеивание* и *перемешивание*.

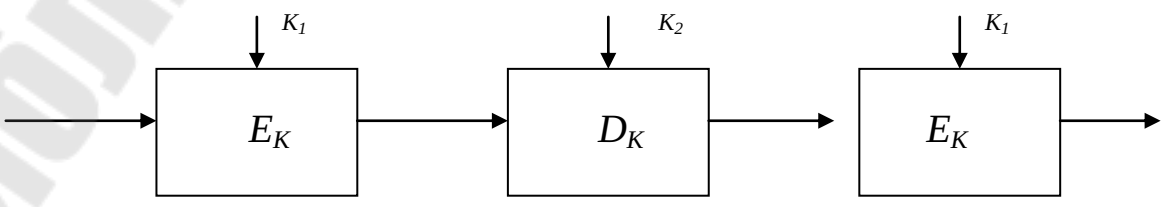
Рассеивание представляет собой распространение влияния од-

ного знака открытого текста на много знаков шифротекста, что позволяет скрыть статистические свойства открытого текста.

Перемешивание предполагает использование таких шифрообразующих преобразований, которые усложняют установление взаимосвязи статистических свойств открытого и шифрованного текстов. Однако шифр должен не только затруднять раскрытие, но и обеспечивать легкость шифрования и расшифрования при известном пользователю ключе.

Дадим краткую характеристику наиболее известным симметричным криптоалгоритмам шифрования.

Алгоритм шифрования данных DES (Data Encryption Standart), опубликованный в 1997 году. В нем используется только один ключ длиной 56 бит. До недавнего времени это алгоритм считался относительно безопасным алгоритмом шифрования. Самым практичным способом его взламывания является метод перебора всех возможных значений ключа, а их всего 256 (более чем 7,2×1016). Но в настоящее время на рынке имеются чипы, позволяющие перебирать до 200 миллионов значений ключа в секунду, а их стоимость составляет всего десятки долларов. Поэтому, если хакеру, имеющему бюджет до 500 долларов, на успешный взлом подобного ключа с помощью персонального компьютера потребуется несколько десятков лет времени, то крупной корпорации, имеющей бюджет до 10 млн. долларов и суперЭВМ на это может потребоваться всего 6 минут. Поэтому DES имеет смысл использовать в качестве фундамента в других алгоритмах с более длинным ключом. К таким алгоритмам относятся алгоритмы, получаемые путем комбинирования блочных алгоритмов. Одним из таких способов комбинирования является многократное шифрование, то есть использование блочного алгоритма несколько раз с разными ключами для шифрования одного и того же блока открытого текста. Например, У. Тачмен предложил шифровать блок открытого текста P три раза с помощью двух ключей K_1 и K_2.



Шифрование

Расшифрование

Процедура такого шифрования выглядит так:

$C=E_{K1}(D_{K2}(E_{K1}(P)))$,

то есть блок открытого текста P сначала шифруется ключом K_1, затем расшифровывается ключом K_2 и окончательно зашифровывается ключом K_1. Процедура расшифровывания выполняется в обратном порядке:

$P=D_{K1}(E_{K2}(D_{K1}(C)))$,

то есть блок шифротекста C сначала расшифровывается ключом K_1, затем зашифровывается ключом K_2 и окончательно расшифровывается ключом K_1.

При трехкратном шифровании можно применить три разных ключа. При этом возрастает общая длина результирующего ключа, но трехключевой вариант имеет еще большую стойкость. Алгоритм 3-DES (Tripple DES – тройной DES) используется в ситуациях, когда надежность алгоритма DES считается недостаточной.

Стандарт шифрования ГОСТ 28147-78. Этот алгоритм криптографического преобразования данных предназначен для аппаратной и программной реализации и представляет собой 64–битовый блочный алгоритм с 256-битовым ключом (число значений ключа - 1,15792×1077). Алгоритм ГОСТ 28147-89 считается очень стойким – в настоящее время для его раскрытия не предложено более эффективных методов, чем метод "грубой силы", то есть простого перебора. Его высокая стойкость достигается в первую очередь за счет большой длины ключа – 256 бит, которая может увеличиваться до 320.

Американский стандарт шифрования AES. В 1997 году Американский институт стандартизации объявил конкурс на новый стандарт

симметричного криптоалгоритма, названного AES (Advanced Encryption Standard). К его разработке были подключены самые крупные центры криптологи со всего мира.

К криптоалгоритмам – кандидатам на новый стандарт AES – были предъявлены следующие требования:

алгоритм должен быть симметричным;

алгоритм должен быть блочным шрифтом;

алгоритм должен иметь длину блока 128 бит и поддерживать три длины ключа: 128, 102 и 256 бит.

В октябре 2000 были подведены итоги конкурса. В результате из 15 алгоритмов - претендентов победителем был объявлен алгоритм Rijndael, разработанный двумя криптографами из Бельгии: Винсентом Риджменом и Джоан Даймен. Этот алгоритм и стал новым стандартом шифрования данных AES. Он стал новым стандартом шифрования данных благодаря целому ряду преимуществ перед другими алгоритмами. Прежде всего, он обеспечивает высокую скорость шифрования на всех платформах: как при программной, так и при аппаратной реализации. Кроме того, требования к ресурсам для его работы минимальны, что важно при его использовании в устройствах, обладающих ограниченными вычислительными возможностями.

Для шифрования данных применяются и другие блочные симметричные криптоалгоритмы: *IDEA, RC2, RC5, Blowfish*.

6.3. Асимметричные системы шифрования

Такие системы были разработаны в 1970-х годах. Их принципиальное отличие от симметричных криптосистем состоит в том, что для шифрования информации и ее последующего расшифрования используются различные ключи:

- *открытый ключ К*: используется для шифрования информации, вычисляется из секретного ключа *k*;
- *секретный ключ k*: используется для расшифрования информации, зашифрованной с помощью парного ему открытого ключа *К*.

Эти ключи различаются таким образом, что с помощью вычислений нельзя вывести секретный ключ *k* из открытого ключа *К*. Поэтому открытый ключ *К* может свободно передаваться по каналам связи.

Асимметричные системы называют еще двухключевыми криптографическими системами или системами с открытым ключом.

Обобщенная схема асимметрической криптосистемы шифрования с открытым ключом показана на рисунке 6.4.

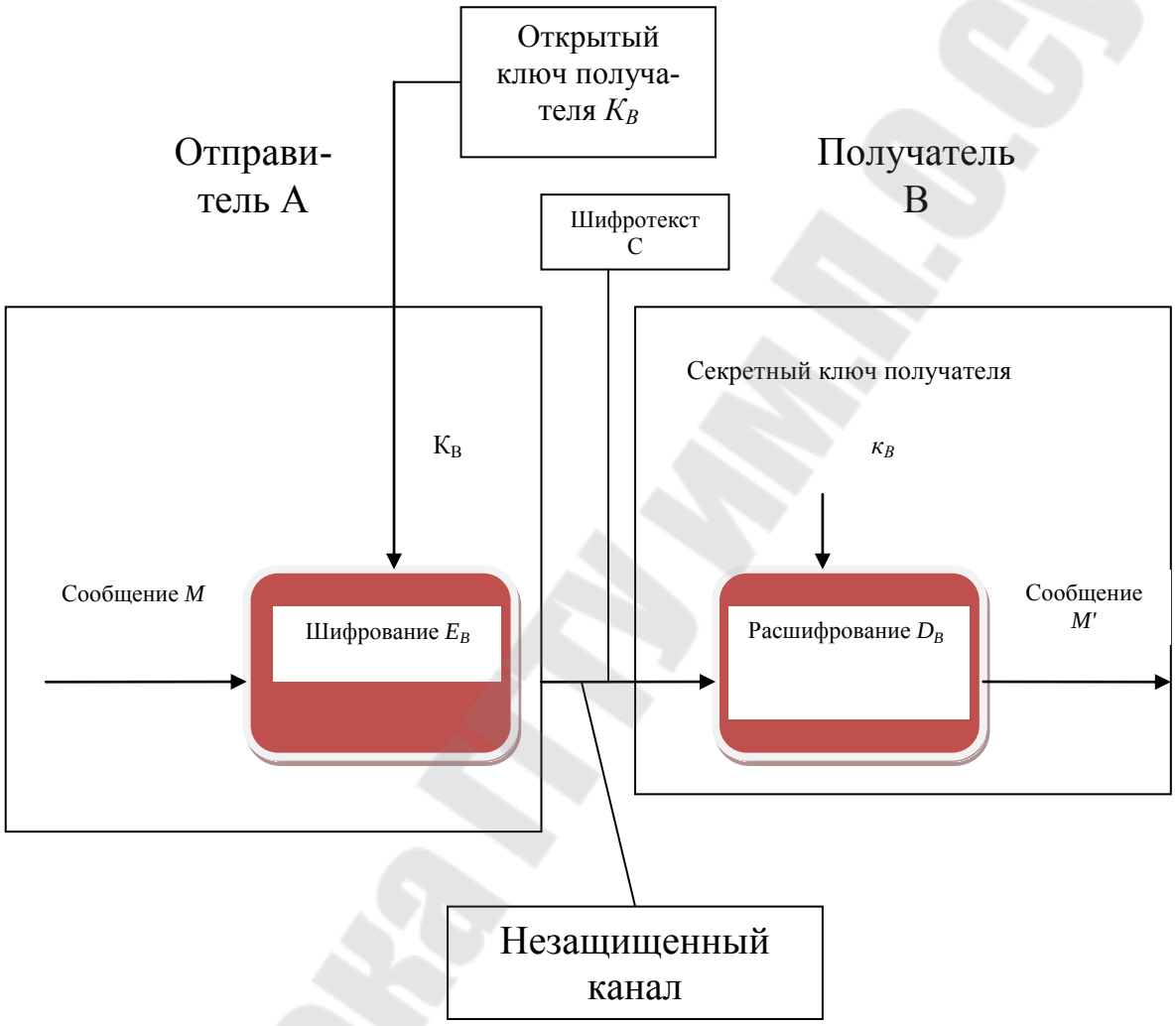


Рисунок 6.4 - схема асимметрической криптосистемы шифрования с открытым ключом

Процесс передачи шифрованной информации в такой системе осуществляется следующим образом:

1. *Подготовительный этап.*

   Абонент В генерирует пару секретных ключей: секретный ключ k_B и открытый ключ K_B. Открытый ключ K_B посылается абонен-

ту А и остальным абонентам (или делается доступным, например, на разделяемом ресурсе).

2. *Этап использования* – обмен информацией между абонентами А и В.

   Абонент А зашифровывает сообщение с помощью открытого ключа K_B абонента В и отправляет шифротекст абоненту В.

   Абонент В расшифровывает сообщение с помощью своего секретного ключа k_B. Никто другой (в том числе абонент А) не может расшифровать данное сообщение, так как не имеет секретного ключа абонента В. Защита информации в ассиметричной криптосистеме основана на секретности ключа k_B получателя сообщения.

Как и в случае симметричных криптографических систем, с помощью ассиметричных криптосистем обеспечивается не только конфиденциальность, но также подлинность и целостность передаваемой информации.

Преимущества ассиметричных криптосистем (АК) перед симметричными (СК):

- в АК решена сложная проблема распределения ключей между пользователями, так как каждый пользователь может сгенерировать свою пару ключей сам, а открытые ключи пользователей могут свободно публиковаться и распространяться по сетевым коммуникациям;
- исчезает квадратическая зависимость числа ключей от числа пользователей; в АК количество используемых ключей связано с количеством абонентов N линейной зависимостью ($2\times N$, а не N^2 как в СК);
- АК позволяют реализовать протоколы взаимодействия сторон, которые не доверяют друг другу, поскольку при использовании АК закрытый ключ должен быть известен только его владельцу.

Недостатки АК:

- на настоящий момент нет математического доказательства необратимости используемых в асимметричных алгоритмах функций;

- по сравнению с СК, АК существенно медленнее, поскольку при шифровании и расшифровке используются весьма ресурсоемкие операции. По этой же причине реализовать аппаратные шифратор с асимметричным алгоритмом существенно сложнее, чем симметричным;
- необходимо защищать открытые ключи от подмены.

Наиболее известные алгоритмы асимметричного шифрования:

- криптоалгоритм RSA (1978г., авторы: Р. Райвест, А. Шамир и А. Адельман) – первый алгоритм с открытым ключом;
- алгоритм ECES – основан на базе так называемых эллиптических кривых.
-

6.4. Функции хэширования

Функция хэширования (хэш-функция) представляет собой преобразование, на вход которого подается сообщение переменной длины M, а выходом является строка фиксированной длины h(M). Иначе говоря, хэш-функция h(.) принимает в качестве аргумента сообщение (документ) M произвольной длины и возвращает хэш-значение (хэш) H=h(M) фиксированной длины (рисунок 6.5)

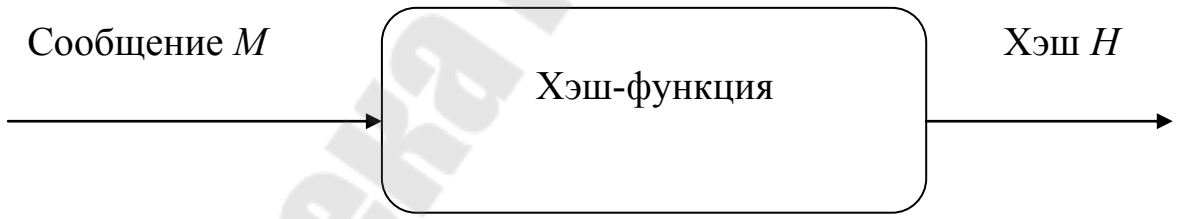


Рисунок 6.5 – схема хэширования

Хэш-значение – это дайджест сообщения *M*, то есть сжатое двоичное представление основного сообщения *M* произвольной длины. Хэш-значение *h(M)* формируется функцией хэширования.

Функция хэширования позволяет сжать подписываемый документ *M* до 128 бит и более (в частности, 128 или 256 бит), тогда как *M* может быть размером в мегабайт и более. Следует отметить, что значение хэш-функции *h(M)* зависит сложным образом от документа *M* и

не позволяет восстановить сам документ *M*.

Таким образом, функция хэширования может использоваться для обнаружения изменений сообщения, то есть она может служить для формирования *криптографической контрольной суммы*. В этом качестве хэш-функция используется для контроля целостности сообщения, при формировании и проверке электронной цифровой подписи. Хэш-функция широко используется также в целях аутентификации пользователей.

Известные функции хэширования:

- российский стандарт ГОСТ Р34.11-94. Вычисляет хэш размером 32 бит;
- MD (Message Digest) – ряд алгоритмов хэширования, наиболее распространенных в мире. Каждый из них вырабатывает 128-битовый хэш-код;
- SHA (Secure Hash Algorithm) – этот алгоритм вычисления дайджеста сообщений, вырабатывающий 160-битовый хэш-код входных данных. Широко распространен в мире; используется во многих сетевых протоколах защиты информации.

6.5. Электронная цифровая подпись (ЭЦП)

ЭЦП используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. При таком обмене электронными документами существенно снижаются затраты на обработку и хранение документов, ускоряется их поиск. Но возникает проблема аутентификации автора электронного документа и самого документа, то есть, установления подлинности автора и отсутствия изменений в полученном электронном документе.

Целью аутентификации электронных документов является их защита от возможных видов злоумышленных действий:

- *активного перехвата* - подключение к сети и изменение файлов с документами;
- *маскарада* - абонент С посылает документ абоненту В от имени абонента А;
- *подмены* - абонент В изменяет или формирует новый документ и заявляет, что получил его от абонента А;
- *повтор* - абонент С повторяет ранее переданный документ, который абонент А посылал абоненту В.

Эти виды злоумышленных действий могут нанести существенный ущерб банковским и коммерческим структурам, государственным предприятиям и организациям, частным лицам, применяющим в своей деятельности компьютерные информационные технологии. Эффективно решить проблему целостности сообщения и подлинности автора сообщения позволяет методология ЭЦП.

Функционально цифровая подпись аналогична обычной рукописной подписи и обладает ее основными достоинствами:

- удостоверяет, что подписанный текст исходит от лица, поставившего подпись;
- не дает самому этому лицу возможности отказаться от обязательств, связанных с подписанным текстом;
- гарантирует целостность подписанного текста.

ЭЦП представляет собой относительно небольшое количество дополнительной цифровой информации, передаваемой вместе с подписываемым текстом.

ЭЦП основана на обратимости асимметричных шрифтов, а также на взаимосвязанности содержимого сообщения, самой подписи и пары ключей. Изменение хотя бы одного из этих элементов сделает невозможным подтверждение подлинности цифровой подписи. ЭЦП реализуется при помощи асимметричных алгоритмов шифрования и хэш-ключей.

Технология применения системы ЭЦП предполагает наличие системы абонентов, посылающих друг другу подписанные электронные документы. Для каждого абонента генерируется пара ключей: секретный и открытый. Секретный ключ хранится абонентом в тайне и используется им для формирования ЭЦП. Открытый ключ известен всем другим пользователям и предназначен для проверки ЭЦП получателем подписанного электронного документа.

Система ЭЦП включает две основные процедуры:

- процедуру формирования цифровой подписи;
- процедуру проверки цифровой подписи.

В процедуре формирования подписи используется секретный ключ отправителя сообщения, а в процедуре проверки подписи – открытый ключ отправителя.

В настоящее время существует большое количество алгоритмов ЭЦП:

- DSA (digital Signature Algorithm) – предложен в 1991 году Национальным институтом стандартов и технологий

США, подписи DSA имеют длину 320 бит;

- стандарт цифровой подписи ГОСТ Р34.10-94 – первый российский стандарт. Его алгоритм концептуально близок к алгоритму DSA. В этом стандарте длина подписи составляет 256 бит. Этот стандарт вступил в действие с 1995 года;
- стандарт цифровой подписи ГОСТ Р34.10-2001 – был принят в 2001 году взамен вышеупомянутого. Необходимость новой разработки была вызвана потребность в повышении стойкости ЭЦП к несанкционированным изменениям. Внедрение цифровой подписи на базе этого стандарта повышает по сравнению с предшествующей схемой, уровень защищенности передаваемых сообщений от подделок и искажений. Этот стандарт рекомендуется использовать в новых системах обработки информации различного назначения, а также при модернизации действующих систем.

Тема 7. Сжатие информации

7.1. Методы сжатия

Во многих случаях информация, содержащаяся в файлах, избыточна. Для устранения избыточности используются специальные методы сжатия данных, основанные на поиске в файле избыточной информации и последующем ее кодировании с целью получения минимального объема.

Различают следующие виды избыточности:

Естественная избыточность связана с первичным алфавитом, а *искусственная* с вторичным алфавитом. Мысль, выраженная более кратко без потери информации, обуславливает *семантическую избыточность*. *Статистическая избыточность* обуславливается не равновероятностным распределением качественных признаков первичного алфавита и их взаимозависимостью.

Сжатие информации – это процесс преобразования информации, хранящейся в файле, к виду, при котором уменьшается избыточность в ее представлении и соответственно требуется меньший объем памяти для хранения; процесс сокращения количества битов, необходимых для хранения и передачи некоторого объема информации.

Существует *сжатие без потерь*, когда информация, восстановленная из сжатого сообщения, в точности соответствует исходной (применяется при обработке текстов, записанных на естественном или искусственном языках), и *сжатие с потерями* (необратимое), когда восстановленная информация только частично соответствует исходной (применяется при обработке изображений и звука, для цифровой записи аналоговых сигналов).

Классификация видов сжатия по виду информации:

1. Побуквенное сжатие
2. Сжатие слов и словосочетаний
3. Сжатие и свертывание текста

4. Сжатие массивов чисел
5. Сжатие графической информации

1) *Побуквенное сжатие:*

- коды Шеннона – Фано;
- оптимальные коды (коды Хаффмена);
- блочное кодирование;
- переход к кодированию с основанием больше двух:

$$k \geq \frac{\log 32}{\log 2} = 5 \qquad k \geq \frac{\log 32}{\log 3} = 3.15$$

2) *Сжатие слов и словосочетаний*:

- аббревиатура;
- иероглифы;
- отбрасывание окончаний слов;
- отбрасывание часто повторяющихся букв;
- выборочное отбрасывание букв, например:

КИБЕРНЕТИКА $\rightarrow$ КБРЕИА $\rightarrow$ КРИ

- лексическое кодирование, в котором отдельные лексемы заменяются двоичными кодами, например:

Наименование лексем	**Длина в байтах**	**N-кол. лексем**	**$\log_2 N$ бит**
Фамилия	20	1000	10 бит
Имя	15	100	7 бит
Отчество	20	100	7 бит
Должность	30	50	6 бит
Отдел	50	30	5 бит
	135байт		»5 байт

Таким образом, получили сжатие в 135/5=27 раз.

3) Сжатие и свертывание текста:

- библиографическое описание (УДК, Автор, наименование, издательство);
- аннотация (до 2/3 страницы);

- реферат (до 16 стр., автореферат - один печатный лист);

4*)Сжатие массивов чисел*

При сжатии массивов чисел широко используется следующий метод. На предприятиях номенклатура (это изделия, материалы, инструменты и др.) кодируются десятичными номерами. Массивы таких чисел могут составлять десятки тысяч. Если этот массив чисел упорядочить в порядке возрастания, то последующие числа будут отличаться от предыдущих чисел только младшими разрядами. Тогда можно все повторяющиеся цифры заменить одним символом, например w,z,y.

5 5 3 8 1 4 2	wz2
5 5 3 8 1 4 3	wz3
5 5 3 8 1 4 5	wz5
5 5 3 8 1 6 1	wy1
5 5 3 8 1 6 3	wy3

где – w=5538, z=14, y=16.

5)Сжатие графической информации:

- кодирование серий "0" и "1";

L(0): $M(0) = 2^{L(0)}-1$
L(1): $M(1) = 2^{L(1)}-1$

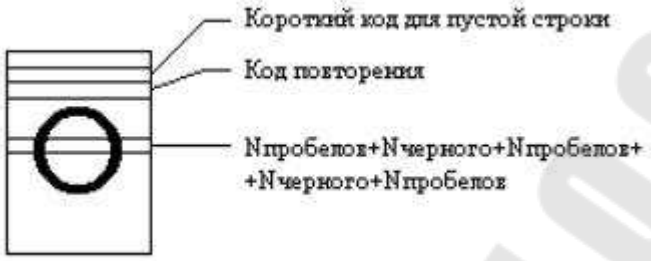
0	000
00	001
000	010
0000	011
00000	100
больше 5-ти нулей	101
больше 10-ти нулей	110
больше 15-ти нулей	111

Пример :
18 нулей
Код - 111010

- замена повторяющихся символов их количеством, например:

AAAABBBCCCCCDD
Код - 4A3B5C2D

- короткие коды и коды повторения.



Основные способы сжатия: статистический и словарный.

При *статистическом* способе каждому символу присваивается код, основанный на вероятности его появления в тексте. Высоко вероятные символы получают короткие коды и наоборот. Одним из самых ранних и широко известных статистических методов является алгоритм Хаффмена, при котором символы заменяются кодом, состоящим из целого количества битов. Позднее он был вытеснен арифметическим кодированием, имеющим схожую с кодом Хаффмена функцию и основанным на идее кодирования символов дробным числом битов. Арифметическое сжатие может быть использовано в тех случаях, когда степень сжатия важнее, чем временные затраты на сжатие информации.

При *словарном способе* группы последовательных символов или "фраз" заменяются кодом. Замененная фраза может быть найдена в некотором словаре. В 1977 году Лемпель и Зив предложили свою модификацию словарного метода, отличающуюся от хаффменовского и арифметического методов, в которой сжатие основано на свойстве "потока символов" иметь повторяющиеся участки. Поток символов - это исходные данные для сжатия (например, текстовый файл, массив). Основная идея алгоритма Лемпеля и Зива состоит в том, что второе и последующие вхождения некоторой строки символов в сообщении заменяются ссылкой на ее первое появление в сообщении.

В последнее время было показано, что любая практическая схема словарного сжатия может быть сведена к соответствующей статистической схеме сжатия, и найден общий алгоритм преобразования словарного метода в статистический. Поэтому при поиске лучшего сжатия статистическое кодирование обещает быть наиболее плодотворным, хотя словарные методы являются более быстрыми.

Еще одним из наиболее простых и наглядных является метод сжатия последовательностей одинаковых символов, не относящийся к названным основным методам.

Метод основан на представлении последовательности одинаковых символов в виде двух величин K и S, где K - количество повторяющихся символов, S - код этого символа. Основным недостатком данного метода является то, что он обеспечивает сжатие лишь в случае, когда в исходном файле основную часть составляют повторяющиеся символы. В противном случае сжатый файл может занимать больше места, чем исходный неуплотненный файл. Наиболее эффективно метод сжатия последовательностей одинаковых символов работает в случае двоичных файлов.

Синтетические алгоритмы.

Рассмотренные выше алгоритмы в «чистом виде» на практике не применяются из-за того, что эффективность каждого из них сильно зависит от начальных условий. В связи с этим, современные средства архивации, используют более сложные алгоритмы, основанные на комбинации нескольких теоретических методов. Общим принципом в работе таких «синтетических» алгоритмов является предварительный просмотр и анализ исходных данных для индивидуальной настройки алгоритма на особенности обрабатываемого материала.

7.2. Программы-архиваторы

На основе методов сжатия данных созданы различные программы, называемые архиваторами или упаковщиками. Существует много программ-архиваторов, имеющих различные показатели по степени и времени сжатия. Среди самых известных и часто используемых программ выделяются следующие: ARJ, PKZIP, RAR, HA и т. д. для DOS и WinARJ, WinZip, WinRAR, 7-ZIP, Zip Magic для Windows.

Программы для архивации файлов позволяют помещать копии файлов на диске в сжатом виде в архивный файл, извлекать файлы из архива, просматривать оглавление архива и др. В настоящее время архиваторы, работающие под Windows, вытесняют конкурентов в основном за счет использования 32-х битной шины данных, более удобного и интеллектуального интерфейса, расширенных возможностей и более совершенных алгоритмов сжатия. Они также поддерживают не один, как раньше, а сразу несколько различных форматов архивных файлов.

Программы-архиваторы позволяют создавать и такие архивы, для извлечения из которых содержащихся в них файлов не требуются какие-либо программы-распаковщики, так как они сами содержат программу распаковки. Такие архивные файлы называются *самораспаковывающимися*.

Самораспаковывающийся архивный файл – это загрузочный, исполняемый модуль, который способен к самостоятельной разархивации находящихся в нем файлов без использования программы-архиватора.

К базовым функциям, которые выполняют большинство современных архиваторов, относятся:

- извлечение файлов из архивов;
- создание новых архивов или добавление файлов в имеющийся архив;
- создание самораспаковывающихся архивов;
- создание распределенных архивов на носителях малой емкости;
- тестирование целостности структуры архивов;
- полное или частичное восстановление поврежденных архивов;
- защита архивов от просмотра или несанкционированной модификации.

Список использованных источников

1. Базлов Е.Ф. Основы теории информации. Уч. пособие. – Казань, 1990.
2. Дмитриев В.И. Прикладная теория информации: Учебник для вузов. - М: Высшая школа, 1989.
3. Игнатов В.А. Теория информации и передачи сигналов. - М.: Советское радио, 1979.
4. Колесник В.Д., Полтырев Г.Ш. Курс теории информации. – М.: Наука, 1982. – 416 с.
5. Теория информации: практикум для студентов спец. 1-26 03 01 "Управление информационными ресурсами" / авт.-сост. В. В. Бондарева. – Гомель: учреждение образования "Белорусский торгово-экономический университет потребительской кооперации", 2008. – 44с.
6. Хэмминг Р.В. Теория кодирования и теория информации и. - М: Радио и связь, 1983.
7. Шаньгин В.Ф. Защита информации. Эффективные методы и свойства. – М.: ДМК, 2008.
8. Мельников В.П. Информационная безопасность и защита информации. Учебное пособие для вузов. /Под ред. С.А.Клейменова. – М: Академия, 2009.
9. Теория информации: практикум для студентов спец. 1-26 03 01 "Управление информационными ресурсами" / авт.-сост. В. В. Бондарева. – Гомель: учреждение образования "Белорусский торгово-экономический университет потребительской кооперации", 2008. – 44с.
10. В. О. Лукьяненко, В. И. Мисюткин. Элементы теории информации. Практическое пособие по одноименной дисциплине для слушателей ИПК и ПК специальности 1-40 01 73 «Программное обеспечение информационных систем». – Гомель: ГГТУ им. П. О. Сухого, 2014. – 34 с.

Мисюткин Виктор Иванович

ЭЛЕМЕНТЫ ТЕОРИИ ИНФОРМАЦИИ

ПОСОБИЕ
по одноименной дисциплине
для слушателей специальности 1-40 01 73
«Программное обеспечение информационных систем»
заочной формы обучения

Подписано к размещению в электронную библиотеку
ГГТУ им. П. О. Сухого в качестве электронного
учебно-методического документа 13.11.15.
Рег. № 53Е.
http://www.gstu.by