УДК 004
ББК 32.811
Б 81

Рецензенты: В. И. Мисюткин, канд. техн. наук, доцент кафедры информатики Гомельского государственного технического университета имени П. О. Сухого;
М. А. Грибовская, канд. физ.-мат. наук, доцент кафедры информационно-вычислительных систем Белорусского торгово-экономического университета потребительской кооперации

Рекомендован к изданию научно-методическим советом учреждения образования «Белорусский торгово-экономический университет потребительской кооперации». Протокол № 2 от 13 декабря 2011 г.

Бондарева, В. В.
Б 81 Теория информации : курс лекций для студентов специальности 1-26 03 01 «Управление информационными ресурсами» / В. В. Бондарева. – Гомель : учреждение образования «Белорусский торгово-экономический университет потребительской кооперации», 2012. – 84 с.
ISBN 978-985-461-982-8

УДК 004
ББК 32.811

ISBN 978-985-461-982-8

ВВЕДЕНИЕ

Теория информации – раздел прикладной математики, определяющий понятие информации, ее свойства и устанавливающий предельные соотношения для систем передачи данных. Теория информации, как и любая математическая теория, оперирует с математическими моделями, а не с реальными физическими объектами (источниками и каналами связи). Она основывается на теории случайных событий, для описания которых применяются понятия «вероятность» и «энтропия».

Примеры использования теории информации можно найти в информатике, технике, психологии, биологии, физике, педагогике, лингвистике и т. д. Однако теория информации применима для решения практических задач в той мере, в какой описываемые материальные системы или процессы соответствуют исходным положениям теории.

Математическое понятие информации связано с возможностью ее количественного измерения. Количественная мера информации не привязывается к ее смысловой основе. Таким образом, теория информации применима для решения лишь тех практических задач, в которых допустимо игнорирование смысловой стороны информации.

Курс лекций состоит из пяти тем: «Исходные понятия теории информации», «Количественная оценка информации», «Кодирование символьной информации», «Представление и обработка чисел в компьютере», «Передача информации». По каждой теме представлены основные формулы, теоретические сведения, вопросы и типовые задачи. Числовые данные в задачах в основном носят условный характер. Теоретический материал сопровождается схемами и таблицами.

1. ИСХОДНЫЕ ПОНЯТИЯ ТЕОРИИ ИНФОРМАЦИИ

1.1. Предмет теории информации

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

В XX в. бурное развитие получили различные средства связи: телефон, телеграф, радио. Их назначение заключалось в передаче сообщений. В процессе их эксплуатации возник ряд проблем:

- как обеспечить надежность связи при наличии помех;
- какой применять способ кодирования сообщения;
- как закодировать сообщение, чтобы при минимальной его длине обеспечить передачу с определенной степенью надежности;
- какой должна быть пропускная способность канала связи, для того чтобы передавать поступающую информацию без задержек и искажений.

Эти проблемы требовали разработки теории передачи сообщений или теории информации.

Теория информации как самостоятельная дисциплина возникла в ходе решения следующей задачи: *обеспечить надежную и эффективную передачу информации от источника к приемнику при условии, что передаче препятствуют помехи.* Формулировка этой задачи нуждается в уточнениях:

- в процессе передачи не должно происходить потери информации;
- передача должна осуществляться наиболее быстрым способом;
- помехи присутствуют в любой реальной линии связи.

Решение этой задачи ведется по двум направлениям:

- *техническое*, связанное с практической разработкой линий связи и технических устройств, обеспечивающих быструю и надежную связь, и с обеспечением защиты от помех или уменьшением их воздействия (в основе этих разработок лежат законы, определяющие способы кодирования информации и условия надежной передачи информации);
- *математическое*, основанное на теории случайных событий и связаное с возможностью ее количественного измерения.

Попытки количественного измерения информации предпринимались неоднократно. Первые отчетливые предложения об общих способах измерения количества информации были сделаны Р. Фишером (1921) в процессе решения вопросов математической статистики.

Проблемами хранения информации, передачи ее по каналам связи и задачами определения количества информации занимались Р. Хартли (1928) и Х. Найквист (1924). Так, Р. Хартли заложил основы теории информации, определив меру количества информации для некоторых задач. Наиболее убедительно эти вопросы были разработаны и обобщены американским инженером К. Шенноном в основополагающей работе «Математическая теория связи» (1948). С этого времени началось интенсивное развитие теории информации и углубленное исследование вопроса об измерении ее количества.

Чтобы применить математические средства для изучения информации, потребовалось отвлечься от смысла и содержания информации. Этот подход был общим для указанных исследователей, так как математика оперирует с количественными соотношениями, рассматривая физическую природу изучаемых объектов. Например, если находится сумма двух чисел 5 и 10, то она будет одинакова для любых объектов, определяемых этими числами.

Примеры практической применимости теории информации можно найти в информатике, технике, психологии, биологии, физике, педагогике, лингвистике и т. д. Важными для передачи и хранения являются количественные характеристики информации и способы их оценки. Теория информации применима для решения лишь тех практических задач, в которых допустимо игнорирование смысловой стороны информации.

1.2. Начальные определения

Информация *(лат. informatio) – это сведения, разъяснения, изложение.*

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

• *в быту* – сведения об окружающем мире и протекающих в нем процессах, воспринимаемые человеком или специальными устройствами;

• *в технике* – сообщения, передаваемые в форме знаков или сигналов;

• *в теории информации* – сведения, которые снимают полностью или уменьшают существующую до их получения неопределенность.

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

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

Материальным носителем информации может быть бумага, воздух, электромагнитное поле и пр. *Хранение и передача* информации связаны с некоторой характеристикой носителя, которая в первом случае не меняется с течением времени (например, буква на бумаге), а во втором – изменяется (например, амплитуда колебаний звуковой волны).

Таким образом, хранение информации связано *с фиксацией состояния* носителя, а распространение – *с процессом*, который протекает в носителе. Информацию можно связать не с любым процессом. *Стационарный процесс*, т. е. процесс с неизменными в течение времени характеристиками (например, ровное горение лампы), информацию не переносит. Для передачи необходим *нестационарный процесс*, т. е. процесс, характеристики которого могут изменяться. Если лампу включать и выключать, т. е. изменять характеристику носителя, то чередованием вспышек и пауз можно передать информацию (например, как в азбуке Морзе).

Изменение характеристики носителя, которое используется для представления информации, называется ***сигналом***.

Одиночный сигнал не содержит много информации. Поэтому для передачи информации используется ряд следующих друг за другом сигналов.

Последовательность сигналов называется ***сообщением***.

От источника к приемнику информация передается в виде сообщений. Сообщение служит переносчиком информации, а информация является содержанием сообщения.

Соответствие между сообщением и содержащейся в нем информацией называется ***правилом интерпретации сообщения***.

Данное соответствие может быть *однозначным*, когда сообщение имеет лишь одно правило интерпретации (как, например, в азбуке Морзе), и *неоднозначным*, в частности:

- информация может передаваться различными сообщениями (например, прогноз погоды по радио, в газете, по телефону);

• сообщение может содержать различную информацию для разных приемников (например, переданная в 1936 г. по радио фраза «Над всей Испанией безоблачное небо» для непосвященных людей имела смысл прогноза погоды, а для знакомых с правилом интерпретации это был сигнал к началу военных действий).

С передачей информации связана еще одна пара понятий информации – «источник» и «приемник».

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. К примеру, если человек что-то придумал, но держит это в своем уме, он не является источником информации, однако он им становится, как только свою идею изложит на бумаге или выскажет словами.

Факт приема сообщения еще не означает, что информация получена. Она может считаться полученной только в том случае, если приемнику известно правило интерпретации сообщения. Слыша речь на незнакомом языке, человек оказывается приемником сообщения, но не приемником информации.

1.3. Общая схема передачи информации в линии связи

Способов передачи информации существует множество: почта, телефон, радио, телевидение, компьютерные сети и пр.

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

Общая схема системы связи представлена на рисунке 1.

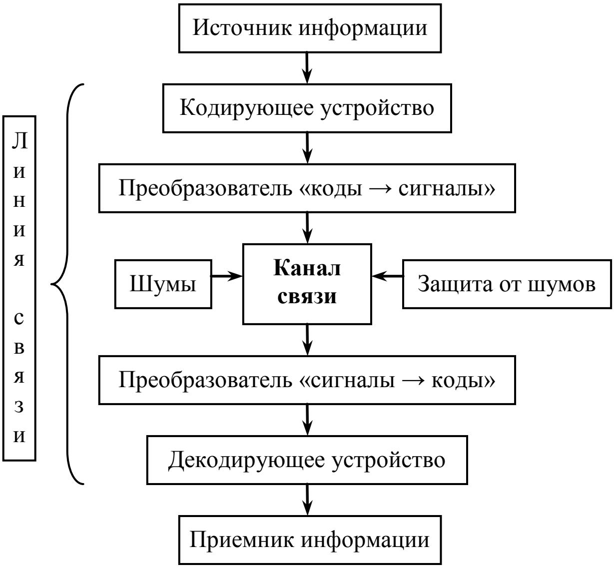


Рисунок 1 – **Общая схема передачи информации**

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[1] или внешним устройством по отношению к источнику информации[2].

Преобразователь переводит коды в последовательность материальных сигналов, т. е. помещает их на материальный носитель. Преобразователь может быть совмещен с кодирующим устройством (например, телефонным аппаратом), но может быть и самостоятельным элементом линии связи (например, модем). При преобразовании часть информации теряется. Так, полоса пропускания частот при телефонной связи находится в диапазоне от 300 до 3 400 Гц, в то время как частоты, воспринимаемые человеческим ухом, лежат в интервале от

---

[1] Мозг порождает информацию и кодирует ее с помощью языка, а затем представляет в виде речевого сообщения посредством органов речи; компьютер обрабатывает и хранит информацию в двоичном представлении, но при выводе ее на экран монитора производит ее перекодировку к виду, удобному пользователю.

[2] Телеграфный аппарат или компьютер по отношению к работающему на нем оператору.

16 до 20 000 Гц; в черно-белом телевидении при преобразовании теряется цвет изображения.

В связи с этим встает задача выработки способа кодирования сообщения, который обеспечивал бы возможно более полное представление исходной информации при преобразовании и был бы согласован со скоростью передачи информации по данной линии 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). К внешним помехам относятся «наводки» от мощных потребителей электричества или атмосферных явлений, одновременное действие нескольких близкорасположенных однотипных источников; к внутренним – физические неоднородности носителя, процессы затухания сигнала в линии связи из-за большой удаленности.

После прохождения сообщения по каналу 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 (например, радиоприемником или телевизором) или существовать самостоятельно (например, модем).

Понятие *линия связи* объединяет все элементы: от источника до приемника информации. Характеристиками линии связи являются скорость, с которой возможна передача сообщений, и степень искажения сообщения в процессе передачи.

1.4. Формы представления информации

Как было сказано ранее, передача информации производится с помощью сигналов, а самим сигналом является изменение некоторой характеристики носителя с течением времени. В зависимости от особенностей изменения этой характеристики выделяют два типа сигналов – *непрерывные* и *дискретные*.

*Сигнал называется **непрерывным** (аналоговым), если его параметр может принимать любое значение в пределах некоторого интервала.*

Сигнал называется ***дискретным****, если его параметр может принимать конечное число значений в пределах некоторого интервала*.

Примеры графиков непрерывных и дискретных сигналов приведены на рисунке 2. Значение параметра сигнала обозначено буквой *Z*, времени – буквой *t*.

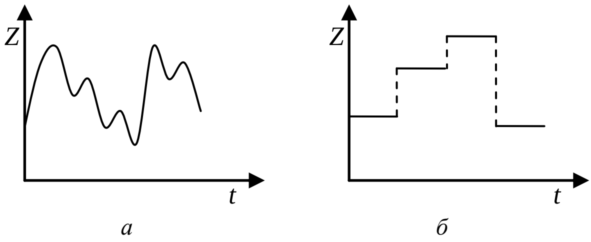


Рисунок 2 – **Непрерывные (*а*) и дискретные (*б*) сигналы**

К непрерывным сигналам относятся речь, музыка, изображение, показание термометра и пр. Примерами устройств, использующих дискретные сигналы, являются часы, цифровые измерительные приборы, книги, табло и пр.

Принципиальным различием непрерывных и дискретных сигналов является то, что дискретные сигналы можно *обозначить,* т. е. приписать каждому значению сигнала *знак* (жест, рисунок, букву), который будет отличать один сигнал от другого.

Знак *– это элемент некоторого конечного множества отличных друг от друга сущностей*.

Вся совокупность знаков, используемых для представления дискретной информации, называется *набором знаков.*

Алфавитом *называется набор знаков, в котором установлен порядок их следования*.

Порядок между знаками устанавливается отношением «больше–меньше»: для двух знаков α и β принимается, что $\alpha < \beta$, если порядковый номер α в алфавите меньше чем β.

Примером алфавита может служить совокупность арабских цифр 0, 1, ..., 9. С его помощью можно записать любое целое число в системах счисления: от двоичной до десятичной. Если к этому алфавиту добавить знаки «+», «–», «.» и «,», то сформируется набор знаков, который позволит записать любое вещественное число. Этот набор 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. Поэтому алфавит должен содержать 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.*

Сам по себе знак, или буква, не несет никакого смыслового содержания. Если знаку приписано содержание, то он будет называться *символом.* Например, массу в физике обозначают буквой m, в таком случае m является символом физической величины «масса» в формулах. Другим примером символов могут служить пиктограммы, обозначающие в компьютерных программах объекты или действия.

Понятия «*знак*», «*буква*» и «*символ*» нельзя считать тождественными.

1.5. Преобразование сообщений

Поскольку имеется два типа сообщений – дискретные (обозначим D) и непрерывные (обозначим N), то между ними возможны четыре варианта преобразований (рисунок 3). Технически это осуществимо.

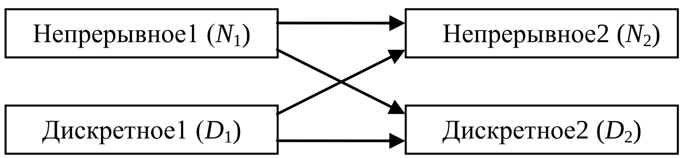


Рисунок 3 – **Виды преобразований сообщений**

Дискретный сигнал лучше поддается преобразованиям, поэтому имеет преимущества перед непрерывным. В то время как в технических системах и в реальных процессах преобладает непрерывный сигнал. Это вынуждает разрабатывать способы преобразования непрерывного сигнала в дискретный. На практике применяются все четыре вида преобразований.

1. ***Преобразование $N_1 \to N_2$.*** Примерами устройств реализации такого типа преобразований являются микрофон, магнитофон и видеомагнитофон, телекамера, радио- и телевизионный приемник. Преобразование $N_1 \to N_2$ всегда сопровождается частичной потерей информации. Потери связаны с внешними помехами и помехами, которые производит само техническое устройство. 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 человеческого голоса.

2. ***Преобразование $N \to D$.*** Примерами устройств реализации такого типа преобразований являются сканер, модем, устройства для цифровой записи звука и изображения, лазерный проигрыватель. Перевод $N \to D$ означает замену описывающей его непрерывной функции $Z(t)$ на некотором отрезке $[t_1, t_2]$ конечным множеством $\{Z_i, t_i\}$ ($i = 0, 1, ..., n$, где n – количество точек разбиения временного интервала). Такое преобразование называется *дискретизацией* непрерывного сигнала и производится с помощью двух операций: *развертки по времени* и *квантования по величине* сигнала.

Развертка по времени состоит в том, что наблюдение за значением величины Z производится не непрерывно, а лишь в определенные моменты времени с интервалом:

$$\Delta t = \frac{t_n - t_0}{n}.$$

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

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

При такой замене очевидно, что чем меньше n, тем меньше число узлов, но и точность замены $Z(t)$ значениями Z_i будет меньшей. При дискретизации может происходить *потеря* части информации. При увеличении n потери информации избежать не удастся, так как n – конечное число.

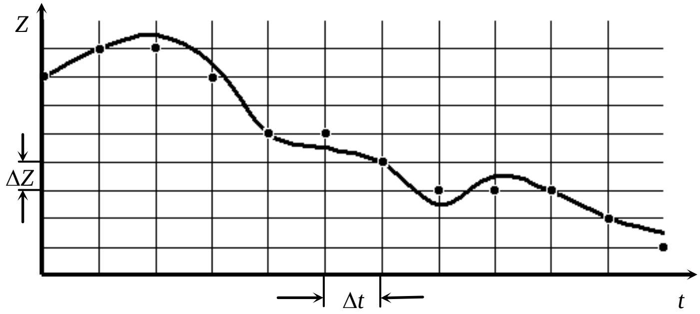


Рисунок 4 – **Дискретизация аналогового сигнала**

Теорема отсчетов (теорема В. А. Котельникова, 1933). *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 колебания – 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:

$$\Delta t \leq \frac{1}{2\nu_m}.$$

Например, для точной передачи речевого сигнала с частотой ν_m до 4 000 Гц при дискретной записи должно производиться не менее 8 тыс. отсчетов в секунду; в телевизионном сигнале ν_m = 4 МГц, следовательно, для его точной передачи потребуется около 8 млн отсчетов в секунду.

Шаг квантования ΔZ определяется следующим образом. Получатель сообщения (человек или устройство) всегда имеет конечную предельную точность распознавания величины сигнала. Например, человеческий глаз в состоянии различить около 16 млн цветов. Это означает, что при квантовании цвета нет смысла делать большее число градаций. Поэтому шаг квантования определяется чувствительностью приемного устройства.

Указанные соображения по выбору шага развертки по времени и квантования по величине сигнала лежат в основе оцифровки звука и изображения. Термины *цифровая запись*, *цифровой сигнал* следует понимать как *дискретное представление с применением двоичного цифрового алфавита*.

Таким образом, преобразование $N \to D$ может осуществляться без потери содержащейся в них информации.

3. ***Преобразование*** $\boldsymbol{D \to N}$. Примером реализации такого типа преобразований является воспроизведение дискретизированных непрерывных сообщений (звука, изображения и т. д.). Такое преобразование осуществляется без потери информации.

4. ***Преобразование*** $\boldsymbol{D_1 \to D_2}$. Примерами реализации такого типа преобразований являются запись-считывание с компьютерных носителей информации, шифровка и дешифровка текста, вычисления на калькуляторе. Такое преобразование состоит в переходе при представлении сигналов от одного алфавита к другому. Эта операция носит название «перекодировка» и может осуществляться без потери информации.

Таким образом, за исключением преобразования $N_1 \to N_2$, оказывается возможным преобразование сообщений без потерь информации. При этом непрерывные и дискретные сообщения не являются равноправными. Сохранение информации в преобразованиях $N \to D$ и $D \to N$ 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:

- высокую помехоустойчивость;
- простоту, надежность и относительную дешевизну устройств по обработке информации;
- точность обработки информации, которая определяется количеством обрабатывающих элементов и не зависит от точности их изготовления;
- универсальность устройств.

Универсальность дискретных сообщений проявляется в том, что сообщения, составленные в различных алфавитах, можно привести к единому алфавиту с помощью обратимого кодирования. Это позволит выделить базовый алфавит и представить 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.

Несимметричность непрерывной и дискретной информации связана с их образованием. Непрерывная информация существует в природе и связана с материальным миром – это размеры, форма, цвет и другие характеристики и свойства объектов. Данная информация передается посредством физических и иных взаимодействий и процессов. Эту природную информацию можно считать хаотической и неупорядоченной, поскольку никем и ничем не регулируется ее появление, существование, использование.

Дискретная информация – это информация, прошедшая отбор, упорядочение, преобразование. Она предназначена для дальнейшего применения человеком или техническим устройством. Дискретная информация – это уже частично осмысленная информация, т. е. имеющая для кого-то смысл и значение.

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

2. КОЛИЧЕСТВЕННАЯ ОЦЕНКА ИНФОРМАЦИИ

2.1. Понятие энтропии

2.1.1. Энтропия как мера неопределенности

При ожидании случайного сообщения существует неопределенность относительно его содержания. Степень неопределенности различна для разных ситуаций.

Пример 1. В шахматы будут играть два игрока: один из них – белыми фигурами, другой – черными. Неопределенность заключается в том, кто начнет игру. В данном случае возможны два исхода с вероятностью $\frac{1}{2}$, т. е. игроком будет выбрана белая фигура или черная. Так как неопределенность одинакова для обоих исходов, то и количество информации тоже будет одинаковым.

Пример 2. На 32 карточках написано по одной букве русского алфавита. Вынимают одну карточку. Вынутой может оказаться любая карточка с вероятностью $\frac{1}{32}$. Уничтоженная неопределенность, т. е. какая буква окажется на карточке, является большей по сравнению с предыдущим примером, так как возможны 32 случая (состояния сигнала). Но на сколько большей, пока ответить невозможно.

Для практики важно иметь возможность произвести численную оценку неопределенности разных опытов.

Пусть опыт имеет n равновероятных исходов. Неопределенность каждого из них зависит от n, т. е. мера неопределенности является функцией числа исходов $f(n)$.

Эта функция обладает следующими свойствами:

1. $f(1) = 0$ при $n = 1$, так как исход опыта не является случайным, и неопределенность отсутствует.

2. Значение функции $f(n)$ возрастает с ростом n, т. е. чем больше число возможных исходов, тем более затруднительным становится предсказание результата опыта.

3. Если два опыта α и β с количествами исходов n_α и n_β *независимы,* тогда неопределенность сложного опыта, который состоит в одновременном выполнении опытов α и β с количеством исходов $n_\alpha \cdot n_\beta$, равна

$$f(n_\alpha \cdot n_\beta) = f(n_\alpha) + f(n_\beta),$$

т. е. функция аддитивна.

Указанному набору свойств удовлетворяет единственная функция $\log n$. Основание логарифма может быть любым, так как

$$\log_b a = \frac{\log_c a}{\log_c b}.$$

Удобным основанием оказывается 2. За единицу измерения принимается неопределенность, содержащаяся в опыте, имеющем лишь два равновероятных исхода (пример 1), которые можно обозначить ИСТИНА и ЛОЖЬ. Единица измерения неопределенности при двух возможных равновероятных исходах опыта называется *бит* [1].

Таким образом, функция, описывающая меру неопределенности опыта с n равновероятными исходами, имеет следующий вид:

$$f(n) = \log_2 n.$$

Эта величина получила название *энтропия* и обозначается H.

Таким образом энтропия опыта с n равновероятными исходами принимает следующее значение:

$$H(\alpha) = \log_2 n. \tag{1}$$

В опыте с n равновероятными исходами каждый исход случаен и вносит свой вклад в неопределенность всего опыта. Так как все n исходов равнозначны, то их неопределенности одинаковы. Из свойства аддитивности неопределенности и формулы (1) следует, что неопределенность, вносимая одним исходом, составляет

$$H = \frac{1}{n}\log_2 n = -\frac{1}{n}\log_2 \frac{1}{n} = -p \cdot \log_2 p, \tag{2}$$

где $p = \dfrac{1}{n}$ – вероятность любого из отдельных исходов.

Рассмотрим другой случай. Пусть опыт α имеет n неравновероятных исходов A_1, A_2 и т. д. Для примера в опыте α два исхода – A_1 и A_2, вероятности исходов обозначим $p(A_1)$ и $p(A_2)$. Тогда с учетом формулы (2) энтропия каждого исхода составит

$$H(A_1) = -p(A_1) \cdot \log_2 p(A_1)$$

и

$$H(A_2) = -p(A_2) \cdot \log_2 p(A_2),$$

а энтропия всего опыта –

$$H(\alpha) = H(A_1) + H(A_2) = -(p(A_1) \cdot \log_2 p(A_1) + p(A_2) \cdot \log_2 p(A_2)).$$

В общем виде, если опыт α имеет n неравновероятных исходов A_1, A_2, ..., A_n, то

[1] Название «бит» происходит от английского выражения binary digit – двоичный разряд или двоичная единица.

$$H(\alpha) = -\sum_{i=1}^{n} p(A_i) \cdot \log_2 p(A_i). \qquad (3)$$

Формула (3) позволяет сравнить неопределенности различных опытов со случайными исходами.

Энтропия *является мерой неопределенности опыта, в котором проявляются случайные события, и равна средней неопределенности всех возможных его исходов*.

Пример 3. Имеются два ящика, в каждом из которых 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.

В первом опыте вероятности вытаскивания шаров разные. Вероятность вытащить белый шар равна $\frac{3}{12}$, черный – $\frac{3}{12}$, красный – $\frac{6}{12}$. Согласно формуле (3) энтропия первого опыта равна

$$H_1 = -\left(\frac{3}{12}\log_2\frac{3}{12} + \frac{3}{12}\log_2\frac{3}{12} + \frac{6}{12}\log_2\frac{6}{12}\right) = 1{,}5 \text{ бита.}$$

Во втором опыте вероятности вытаскивания шаров одинаковые. Опыт имеет три исхода, т. е. можно вытащить белый, красный или черный шар. Согласно формуле (1) энтропия второго опыта равна

$$H_2 = \log_2 3 = 1{,}58 \text{ бита.}$$

Таким образом $H_2 > H_1$, т. е. неопределенность результата во втором опыте выше, и, следовательно, предсказать его можно с меньшей долей уверенности, чем результат первого опыта.

Свойства энтропии:

1. Из формулы (3) следует, что $H = 0$ только в двух случаях:

• если какая-либо из $p(A_j)$ равна единице, тогда из свойства вероятности $\sum_{i=1}^{n} p_i = 1$ следует, что все остальные $p(A_i)$ равны нулю ($i \neq j$), т. е. если один из исходов является *достоверным*, то итог опыта не случайный;

• если все $p(A_i)$ равны нулю, или все рассматриваемые исходы опыта невозможны.

Во всех остальных случаях $H > 0$.

2. Энтропия сложного опыта, состоящего из нескольких независимых опытов, равна сумме энтропии отдельных опытов, т. е. для двух независимых опытов α и β из формулы (1) следует:

$$H(\alpha \wedge \beta) = H(\alpha) + H(\beta). \tag{4}$$

3. Энтропия максимальна в опытах, где все исходы равновероятны, т. е. для двух опытов с числом исходов *n*, в одном случае равновероятном, а в другом – нет, соотношение энтропии опытов имеет следующий вид:

$$-\sum_{i=1}^{n} p(A_i) \cdot \log_2 p(A_i) \le \log_2 n.$$

2.1.2. Условная энтропия

Условная энтропия проявляется в сложном опыте $\alpha \wedge \beta$, когда на исход опыта β оказывает влияние результат опыта α. Например, если в ящике всего два разноцветных шара и опыт α состоит в извлечении первого, а опыт β – второго из них, то α полностью снимает неопределенность сложного опыта $\alpha \wedge \beta$. В этом случае оказывается, что $H(\alpha \wedge \beta) = H(\alpha)$, а не сумме энтропии, как следует из формулы (4).

Связь между α и β состоит в том, что какие-то из исходов A_i опыта α могут оказывать влияние на исходы из B_j опыта β, т. е. некоторые пары событий $A_i \wedge B_j$ не являются независимыми.

Энтропию сложного опыта отражает отношение

$$H(\alpha \wedge \beta) = H(\alpha) + H_\alpha(\beta), \tag{5}$$

где $H_\alpha(\beta)$ – средняя условная энтропия опыта β при условии, что в опыте α реализовался исход A_i:

$$H_\alpha(\beta) = \sum_{i=1}^{n} p(A_i) \cdot H_{A_i}(\beta),$$

где $H_{A_i}(\beta)$ – средняя условная энтропия опыта β при условии выполнения опыта α:

$$H_{A_i}(\beta) = -\sum_{i=1}^{n} p_{A_i}(B_j) \cdot \log_2 p_{A_i}(B_j).$$

Свойства условной энтропии:

1. Условная энтропия является величиной неотрицательной.

2. Условная энтропия не превосходит безусловную, т. е. если опыты α и β независимы, то

$$H_{\alpha}(\beta) = H(\alpha),$$

причем это оказывается наибольшим значением условной энтропии.

3. Из соотношения (5) и предыдущего свойства следует, что

$$H(\alpha \wedge \beta) \leq H(\alpha) + H(\beta),$$

т. е. равенство реализуется только в том случае, если опыты α и β независимы.

Пример 4. В ящике имеются 2 белых шара и 4 черных. Из ящика последовательно извлекают два шара без возврата. Необходимо найти энтропию, связанную с первым и вторым извлечениями, а также энтропию обоих извлечений.

Будем считать опытом α извлечение первого шара. Он имеет два исхода:

- A_1 – вынут белый шар, его вероятность равна

$$p(A_1) = \frac{2}{6} = \frac{1}{3};$$

- A_2 – вынут черный шар, его вероятность составляет

$$p(A_2) = \frac{4}{6} = \frac{2}{3}.$$

Эти данные позволяют с помощью формулы (3) найти энтропию, связанную с первым извлечением $H(\alpha)$:

$$H(\alpha) = -\left(\frac{1}{3}\log_2\frac{1}{3} + \frac{2}{3}\log_2\frac{2}{3}\right) = 0{,}918 \text{ бита.}$$

Опыт β по извлечению второго шара также имеет два исхода: B_1 – вынут белый шар; B_2 – вынут черный шар. Вероятности исходов опыта β будут зависеть от того, каким был исход опыта α.

Если в опыте α был реализован исход A_1, то вероятности исхода опыта β составят

$$p_{A_1}(B_1) = \frac{1}{5}; \quad p_{A_1}(B_2) = \frac{4}{5}.$$

Если в опыте α был реализован исход A_2, то вероятности исхода опыта β составят

$$p_{A_2}(B_1) = \frac{2}{5}; \quad p_{A_2}(B_2) = \frac{3}{5}.$$

Энтропия, связанная со вторым опытом, является условной, она равна

$$H_{A_1}(\beta) = -\left(\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}\right) = 0{,}722 \text{ бита};$$

$$H_{A_2}(\beta) = -\left(\frac{2}{5}\log_2\frac{2}{5} + \frac{3}{5}\log_2\frac{3}{5}\right) = 0{,}971 \text{ бита}.$$

Энтропия, связанная с первым и вторым извлечениями, равна

$$H_\alpha(\beta) = p(A_1)H_{A_1}(\beta) + p(A_2)H_{A_2}(\beta) = \frac{1}{3}\cdot 0{,}722 + \frac{2}{3}\cdot 0{,}971 = 0{,}888 \text{ бита}.$$

Энтропия обоих извлечений определяется по формуле (5):

$$H(\alpha \wedge \beta) = 0{,}918 + 0{,}888 = 1{,}806 \text{ бита}.$$

2.2. Энтропия и информация

Предшествующий опыт α может уменьшить количество исходов и неопределенность последующего опыта β. Разность $H(\alpha)$ и $H_\alpha(\beta)$ показывает, какие новые сведения относительно опыта β получаем, произведя опыт α. Эта величина называется *информацией относительно опыта* β *содержащейся в опыте* α, и выражается следующим образом:

$$I(\alpha,\ \beta) = H_\alpha(\beta).$$

Это выражение дает возможность *численного измерения количества информации*, которое обозначается I.

Следствие 1. Единицей измерения энтропии является *бит*, следовательно, в этих же единицах может быть измерено количество информации.

Следствие 2. Энтропия опыта равна той информации, которую получаем в результате его осуществления.

Пусть опыт $\alpha = \beta$, или просто произведен опыт β. Поскольку он несет полную информацию о себе самом, то неопределенность его исхода полностью снимается, т. е. $H_{\beta}(\beta) = 0$, тогда $I(\beta, \beta) = H(\beta)$.

Свойства информации:

1. Количество информации величина неотрицательная:

$$I(\alpha, \beta) \geq 0,$$

причем $I(\alpha, \beta) = 0$ тогда и только тогда, когда опыты α и β независимы.

2. Информация симметрична относительно последовательности опытов:

$$I(\alpha, \beta) = I(\beta, \alpha).$$

3. Информация опыта равна среднему значению количества информации, содержащейся в каком-либо одном его исходе:

$$I = -\sum_{i=1}^{n} p(A_i) \cdot \log_2 p(A_i). \tag{6}$$

Это вытекает из следствия 2 и представления энтропии в виде формулы (3).

Пример 5. Необходимо установить, какое количество информации требуется, чтобы узнать исход броска монеты.

В данном случае $n = 2$, и события равновероятны, т. е. $p_1 = p_2 = 0{,}5$.

Из формулы (6) следует, что

$$I = -(0{,}5\log_2 0{,}5 + 0{,}5\log_2 0{,}5) = 1 \text{ бит.}$$

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

Из формулы (6) для случая, когда все n исходов равновероятны, т. е. $p(A_i) = \dfrac{1}{n}$, следует, что

$$I = -\sum_{i=1}^{n} \frac{1}{n} \cdot \log_2 n = \log_2 n. \tag{7}$$

Эта формула была выведена в 1928 г. американским инженером

Ральфом Хартли и носит его имя. Ее смысл заключается в следующем: если некоторое множество содержит n элементов, то для однозначной идентификации одного элемента среди прочих требуется количество информации, равное $\log_2 n$.

Частным случаем применения формулы (7) является ситуация, когда $n = 2^k$. Подставляя это значение в формулу (7), получим

$$I = k \text{ бит.}$$

Отсюда следует вывод, что число k равно количеству вопросов с бинарными равновероятными ответами, которые и определяли количество информации.

Пример 6. Случайным образом вынимается карта из колоды в 32 карты. Необходимо выяснить, какое количество информации требуется, чтобы узнать, что это за карта, и как построить угадывание.

В опыте 32 исхода, т. е. $n = 32 = 2^5$, значит, $k = 5$, и, следовательно, $I = 5$ бит.

Вопросы должны быть сформулированы так, чтобы при каждом ответе на вопрос половина исходов отрицалась. Пример вопроса: «Карта красная?»

2.3. Информация и алфавит

При передаче сообщения возникает проблема *распознавания знака*, т. е. установление по полученным сигналам исходной последовательности знаков первичного алфавита. В устной речи это достигается использованием звуков, в письменности – начертанием букв и дальнейшим анализом написанного. Узнавание знака требует получения некоторой порции информации. Можно связать эту информацию с самим знаком и считать, что знак несет в себе некоторое количество информации. Оценим это количество.

Нулевое приближение. Пусть появление всех знаков алфавита в сообщении *равновероятно.* Тогда для английского алфавита $n_e = 27$ (с учетом пробела), для русского алфавита $n_r = 34$. Количество информации, требуемое для распознавания знака, определяется по формуле Хартли (7):

$I_0^e = 4{,}755$ бита;

$I_0^r = 5{,}087$ бита.

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

Первое приближение. Вероятность появления различных букв в тексте различна. Пусть для русского языка буква «е» = «ё», буква «ь» = «ъ», как принято в телеграфном кодировании. В таком алфавите 32 знака с пробелом. Вероятности появления знаков такого алфавита в русских текстах представлены в таблице 1.

Таблица 1 – **Вероятности знаков русского алфавита**

Буква	Относительная частота	Буква	Относительная частота
пробел	0,174	я	0,018
о	0,090	ы	0,016
е	0,072	з	0,016
а	0,062	ь, ъ	0,014
и	0,062	б	0,014
т	0,053	г	0,013
н	0,053	ч	0,012
с	0,045	й	0,010
р	0,040	х	0,009
в	0,038	ж	0,007
л	0,035	ю	0,006
к	0,028	ш	0,006
м	0,026	ц	0,004
д	0,025	щ	0,003
п	0,023	э	0,003
у	0,021	ф	0,002

Из формулы (6) следует, что если p_i – вероятность знака номер i алфавита из n знаков, то *среднее количество информации, приходящейся на один знак*, будет рассчитываться по следующей формуле:

$$I = -\sum_{i=1}^{n} p_i \cdot \log_2 p_i. \tag{8}$$

Это знаменитая формула К. Шеннона. С его работы «Математическая теория связи» (1948) принято начинать отсчет возраста информатики как самостоятельной науки.

Сообщения, в которых вероятность появления каждого отдельного знака не меняется со временем, называются ***шенноновскими***, *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 (8) дает значение средней информации на знак к алфавиту:

I_1^e = 4,04 бита;

I_1^r = 4,36 бита.

Второе приближение. Значение средней информации на букву может быть уменьшено с учетом связей между буквами в словах. В словах буквы появляются не в любых сочетаниях, что 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 $\frac{1}{10}$, а не $\frac{1}{33}$.

Учет в английских словах двухбуквенных сочетаний понижает среднюю информацию на знак до значения:

I_2^e = 3,32 бита;

I_2^r = 3,52 бита.

Учет трехбуквенных сочетаний понижает среднюю информацию на знак до значения:

I_3^e = 3,10 бита;

I_3^r = 3,01 бита.

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

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

где I_∞ – *предельная 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, и определяется структурой самого языка.

Исследования Шеннона для английского языка дали значение $I_\infty^e \approx 1{,}4 \div 1{,}5$ бита, что по отношению к $I_0^e = 4{,}755$ бита создает избыточность около 0,68. Для русского языка 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, уменьшает возможность понимания речи при наличии шума. Избыточность языка позволяет легко восстановить текст, даже если он содержит большое число ошибок или неполон (например, кроссворды, игра «Поле чудес»). В этом смысле избыточность есть определенная страховка и гарантия разборчивости.

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, преобразующее и представляющее его в другом алфавите, который называется *вторичным*.

Код *– (1) это правило, описывающее соответствие знаков или их сочетаний первичного алфавита знакам или их сочетаниям вторичного алфавита*; *(2) это набор знаков вторичного алфавита, используемый для представления знаков или их сочетаний первичного алфавита*.

Кодирование *– это перевод информации, представленной сообщением в первичном алфавите, в последовательность кодов*.

Декодирование *– это восстановление информации в первичном алфавите по полученной последовательности кодов*.

Операции кодирования и декодирования называются ***обратимы-***

***ми**, если их последовательное применение обеспечивает возврат к исходной информации без каких-либо ее потерь.*

Примером обратимого кодирования может выступать 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 $I^{(B)}$. Пусть исходное сообщение, представленное в первичном алфавите, содержит n знаков, а закодированное сообщение – m знаков. Условие обратимости кодирования:

$$n \cdot I^{(A)} \le m \cdot I^{(B)}$$

или

$$I^{(A)} \le \frac{m}{n} \cdot I^{(B)}.$$

Операция обратимого кодирования может увеличить количество информации в сообщении, но не может его уменьшить. Отношение $m : n$ 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

$$K(A,B) \ge \frac{I^{(A)}}{I^{(B)}}. \qquad (9)$$

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 A и B существует множество, возникает проблема выбора *оптимального кода,* который при передаче информации позволяет затратить на передачу сообщения меньше времени.

Из выражения (9) следует, что минимально возможным значением средней длины кода будет

$$K^{\min}(A,B) = \frac{I^{(A)}}{I^{(B)}}. \qquad (10)$$

Формула (10) устанавливает нижний предел длины кода, но неясно, какое возможно приближение $K(A, B)$ 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 близок к отношению средних информаций 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 выражения (10), для минимальной средней длины кода оказывается справедливым следующее соотношение:

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

Превышение $K(A, B)$ над $K^{\min}(A, B)$ называется *относительной избыточностью кода* и обозначается $Q(A, B)$:

$$Q(A,B) = \frac{K(A,B) - K^{\min}(A,B)}{K^{\min}(A,B)} = \frac{K(A,B)}{K^{\min}(A,B)} - 1 = \frac{K(A,B) \cdot I^{(B)}}{I^{(A)}} - 1. \quad (11)$$

Величина $Q(A, B)$ показывает, на сколько операция кодирования увеличила длину исходного сообщения.

$Q(A, B) \to 0$ при $K(A, B) \to K^{\min}(A, B)$, т. е. 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 первой 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. используется два типа сигналов. Такое кодирование называется *двоичным.* Технически это наиболее просто реализуемый вариант, например: существование напряжения в проводе – *импульс* (0), отсутствие – *пауза* (1). Удобство двоичных кодов состоит и в том, что при равных длительностях и вероятностях каждый элементарный сигнал несет в себе 1 бит информации ($\log_2 2 = 1$).

Применение формулы (11) для двоичных сообщений источника без памяти при кодировании знаками равной вероятности дает

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

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 соотносить некоторую последовательность сигналов с эталонной (*таблицей кодов*).

Возможны следующие особенности вторичного алфавита, используемого при кодировании:

- элементарные сигналы (0 и 1) могут иметь одинаковые длительности ($\tau_0 = \tau_1$) или разные ($\tau_0 \neq \tau_1$);
- 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

В случае неравномерного кодирования необходимо *построить такую схему кодирования, в 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 бы их в соответствии с кодовой таблицей. При использовании неравномерного кодирования возможны два подхода к обеспечению различимости кодов:

• использование комбинации элементарных сигналов, интерпретируемой декодером как *разделитель знаков*;

• применение *префиксных кодов* без разделителя.

Неравномерный код с разделителем

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

• коды всех букв будут заканчиваться «00»;

• коды букв не должны содержать двух и более нулей в подряд в середине;

• код буквы (кроме пробела) всегда должен начинаться с единицы;

• разделителю слов «000» всегда предшествует признак конца знака, при этом реализуется последовательность «00000».

В соответствии с правилами построим кодовую таблицу 2 для букв русского алфавита, основываясь на приведенных ранее (таблица 1) вероятностях появления отдельных букв.

Таблица 2 – **Неравномерный код с разделителем**

Знак	Код	Длина кода, k_i	Знак	Код	Длина кода, k_i
пробел	000	3	я	1011000	7
о	100	3	ы	1011100	7
е	1000	4	з	1101000	7
а	1100	4	ь,ъ	1101100	7
и	10000	5	б	1110000	7
т	10100	5	г	1110100	7
н	11000	5	ч	1111000	7
с	11100	5	й	1111100	7
р	101000	6	х	10101000	8
в	101100	6	ж	10101100	8
л	110000	6	ю	10110000	8
к	110100	6	ш	10110100	8

Знак	Код	Длина кода, k_i	Знак	Код	Длина кода, k_i
м	111000	6	ц	10111000	8
д	111100	6	щ	10111100	8
п	1010000	7	э	11010000	8
у	1010100	7	ф	11010100	8

Среднюю длину кода можно найти по формуле

$$K(A,B)=\sum_{i=1}^{n} p_i k_i. \tag{12}$$

Для русского языка длина кода $K(r,\ 2)$ при данном способе кодирования, рассчитанная по формуле (12), составит

$$K(r,2)=\sum_{j=1}^{32} p_j k_j = 4{,}964.$$

Средняя информация на знак $I^{(r)}{}_1 = 4{,}356$ бита, а избыточность кода равна

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

Это означает, что при данном способе кодирования будет передаваться на 14% больше информации, чем содержит исходное сообщение.

Аналогичные вычисления для английского языка дают следующие значения:

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

3.2.2. Префиксные коды

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

Условие 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.

Код Шеннона-Фано

Рассмотрим схему кодирования на следующем примере. Пусть имеется первичный алфавит A, состоящий из шести знаков a_1, a_2, ..., a_6 с вероятностями появления в сообщении 0,3; 0,2; 0,2; 0,15; 0,1; 0,05.

Код строится по следующим правилам:

1. Расположить знаки в порядке убывания вероятностей.
2. Разделить 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) присвоить первый знак кода «0», второй группе (с суммой вероятностей также 0,5) – первый знак кода «1».
4. Продолжить деление каждой из групп на подгруппы по этой же схеме.

В результате получается код, представленный в таблице 3.

Таблица 3 – **Построение кода Шеннона-Фано**

Знак	Вероятность, p_i	Разряды кода				Код	Длина кода, k_i
		I	II	III	IV		
a_1	0,30	0	0			00	2
a_2	0,20	0	1			01	2
a_3	0,20	1	0			10	2
a_4	0,15	1	1	0		110	3
a_5	0,10	1	1	1	0	1110	4
a_6	0,05	1	1	1	1	1111	4

Код удовлетворяет условию Фано и, следовательно, является префиксным.

Средняя длина кода $K(A, 2) = 2{,}45$, средняя информация на знак $I^{(A)}{}_1 = 2{,}309$ бита, избыточность кода $Q(A, 2) = 0{,}0249$, или около 2,5%.

Данный код не оптимальный, так как вероятности появления «0» и «1» неодинаковы (0,35 и 0,65). Применение схемы к русскому языку дает избыточность кода 0,0147, или 1,47%.

Код Хаффмана

Рассмотрим построение кода Хаффмана на том же примере.

Код строится по следующим правилам:

1. Расположить знаки в порядке убывания вероятностей.

2. Создать новый алфавит A_1, объединив два знака с наименьшими вероятностями (a_5 и a_6) и заменив их одним знаком $a^{(1)}$ с вероятностью, равной сумме вероятностей знаков a_5 и a_6.

3. Остальные знаки включить в новый алфавит без изменения. Общее число знаков в новом алфавите будет на единицу меньше, чем в исходном.

4. Переупорядочить знаки по убыванию вероятностей.

5. Продолжить создавать новые алфавиты, пока в последнем не останется два знака.

Вся процедура построения представлена в виде таблицы 4.

6. Провести кодирование в обратном направлении. Двум знакам последнего алфавита присвоить коды «0» и «1» (верхний знак – код «0», нижний – код «1»). В примере знак $a_1^{(4)}$ алфавита $A^{(4)}$, имеющий вероятность 0,6, получит код «0», а знак $a_2^{(4)}$ с вероятностью 0,4 – код «1». В алфавите $A^{(3)}$ знак $a_1^{(3)}$ получает от $a_2^{(4)}$ 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», т. е. «0», а вторая цифра, как было сказано выше, у верхнего знака «0», у нижнего – «1». Таким образом, знак $a_2^{(3)}$ будет иметь код «00», знак $a_3^{(3)}$ – код «01» и т. д. Процедура кодирования представлена в таблице 5.

Таблица 4 – **Построение кода Хаффмана (шаги 1–5)**

Знак	Вероятности				
	Исходный алфавит (A)	Промежуточные алфавиты			
		$A^{(1)}$	$A^{(2)}$	$A^{(3)}$	$A^{(4)}$
a_1	0,3	0,3	0,3	0,4	0,6
a_2	0,2	0,2	0,3	0,3	0,4
a_3	0,2	0,2	0,2	0,3	
a_4	0,15	0,15	0,2		
a_5	0,1	0,15			
a_6	0,05				

Таблица 5 – **Построение кода Хаффмана (шаг 6)**

Знак	Вероятности				
	Исходный алфавит (A)	Промежуточные алфавиты			
		$A^{(1)}$	$A^{(2)}$	$A^{(3)}$	$A^{(4)}$

a_1	0,3	00	0,3	00	0,3	00	0,4	1	0,6	0
a_2	0,2	10	0,2	10	0,3	01	0,3	00	0,4	1
a_3	0,2	11	0,2	11	0,2	10	0,3	01		
a_4	0,15	010	0,15	010	0,2	11				
a_5	0,1	0110	0,15	011						
a_6	0,05	0111								

Коды удовлетворяют условию Фано. Средняя длина кода $K(A, 2) = 2{,}45$, средняя информация на знак $I^{(A)}{}_1 = 2{,}309$ бита, избыточность $Q(A, 2) = 0{,}0249$, вероятности «0» и «1» сблизились (0,47 и 0,53).

Более высокая эффективность кодов Хаффмана по сравнению с кодами Шеннона-Фано становится очевидной, если сравнить избыточности кодов для естественного языка. Применение метода Хаффмана для букв русского алфавита порождает коды, представленные в таблице 6.

Таблица 6 – **Код Хаффмана для букв русского алфавита**

Знак	Код	Длина кода, k_i	Знак	Код	Длина кода, k_i
пробел	000	3	я	001101	6
о	111	3	ы	010110	6
е	0100	4	з	010111	6
а	0110	4	ь,ъ	100001	6
и	0111	4	б	101100	6
т	1001	4	г	101101	6
н	1010	4	ч	110011	6
с	1101	4	й	0011001	7
р	00101	5	х	1000000	7
в	00111	5	ж	1000001	7
л	01010	5	ю	1100101	7
к	10001	5	ш	00110000	8
м	10111	5	ц	11001000	8
д	11000	5	щ	11001001	8
п	001000	6	э	001100010	9
у	001001	6	ф	001100011	9

Средняя длина кода $K(r, 2) = 4{,}395$, избыточность кода $Q(r, 2) = 0{,}0090$, т. е. не превышает 1%, что меньше избыточности кода

Шеннона-Фано.

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

Метод Хаффмана широко применяется в программах-архиваторах, программах резервного копирования файлов и дисков, в системах сжатия информации в модемах и факсах.

3.2.3. Равномерное алфавитное двоичное кодирование. Байтовый код

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

- прописные и строчные буквы латинского алфавита ($26 \cdot 2 = 52$ буквы);
- прописные и строчные буквы русского алфавита ($33 \cdot 2 = 66$ букв);
- цифры от 0 до 9 (всего 10 шт.);
- знаки математических операций, знаки препинания, спецсимволы (≈ 20 шт.).

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 фактическая длина кода равна 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 8 связанных бит получила название ***байт****, а представление таким образом символов –* ***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.

Для совместимости технических устройств и обеспечения возможности обмена информацией между многими потребителями недостаточно только определить длину кода, требуется согласование кодов. Подобное согласование осуществляется в форме *стандартизации кодовых таблиц*.

В персональных компьютерах и телекоммуникационных системах применяется международный байтовый код *ASCII* – американский стандартный код обмена информацией (приложение А). 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, цифр, знаков препинания и математических операций, а также некоторые управляющие коды (номера от 0 до 31), вырабатываемые при использовании клавиатуры.

Вторая часть кодовой таблицы (расширение основной) охватывает коды в интервале от 128 до 255 (первый бит всех кодов – «1»). Она используется для представления символов национальных алфавитов, например русского, а также символов псевдографики. Для этой части также имеются стандарты, например для символов русского языка это стандарты *КОИ*-8, *КОИ*-7 и др.

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

В настоящее время находит все более широкое применение еще один международный стандарт кодировки – *Unicode*. В нем использовано 16-битное кодирование, при котором для каждого символа отводится 2 байта. Такая длина кода обеспечивает включение в первичный алфавит 65 536 знаков. Это позволяет создать и использовать единую для всех распространенных алфавитов кодовую таблицу.

3.2.4. Алфавитное кодирование с неравной длительностью элементарных сигналов. Код Морзе

Примером алфавитного кодирования с неравной длительностью элементарных сигналов является 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*».

Таким образом код оказывается *троичным*.

Свой код (таблица 7) Морзе разработал в 1838 г., т. е. задолго до исследований относительной частоты появления различных букв в текстах. Им был правильно выбран принцип кодирования: буквы, которые 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 «е» получила код *«точка»*. При составлении кодов Морзе для букв русского алфавита учет относительной частоты букв не производился, что повысило его избыточность. Произведем оценку избыточности. Признак конца буквы «0» в кодах не отображается, но учтен в величине k_i – длине кода i-й буквы.

Таблица 7 – **Код Морзе**

Знак	Код	Длина кода, k_i	Знак	Код	Длина кода, k_i
пробел	00	2	я	. – . –	5
о	– – –	4	ы	– . – –	5
е	.	2	з	– – .	4
а	. –	3	ь,ъ	– . . –	5
и	..	3	б	– . . .	5
т	–	2	г	– – .	4
н	– .	3	ч	– – – .	5
с	. . .	4	й	. – – –	5
р	. – .	4	х	. . . .	5
в	. – –	4	ж	. . . –	5
л	. – . .	5	ю	. . – –	5
к	– . –	4	ш	– – – –	5
м	– –	2	ц	– . – .	5
д	– . .	4	щ	– – . –	5
п	. – –	4	э	. . – . .	6

Знак	Код	Длина кода, k_i	Знак	Код	Длина кода, k_i
у	. . –	4	ф	. . – .	5

Среднее значение длины кода $K(r, 3) = 3{,}361$, средняя информация на знак $I_1^{(r)} = 1{,}585$ бита. Для русского алфавита избыточность $Q(r, 3) = 0{,}223$, для английского алфавита $Q(e, 3) = 0{,}19$.

Код Морзе был широко распространен, когда источником и приемником сигналов являлся человек, и на первый план выдвигалось удобство восприятия человеком.

3.2.4. Блочное двоичное кодирование

Наилучший результат кодирования знаков был получен по методу Хаффмана. Для русского алфавита избыточность оказалась менее 1%. Возможны варианты кодирования, при которых кодовый знак относится сразу к нескольким буквам или слову (блоку) первичного алфавита или языка. Кодирование блоков понижает избыточность.

Пусть имеется словарь некоторого языка, содержащий 16 тыс. 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. 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 $K(r) = 6{,}3$ знака (5,3 буквы + пробел), тогда средняя информация на знак первичного алфавита оказывается равной

$$I^{(A)} = \frac{K(A, 2)}{K^{(r)}} = \frac{14}{6{,}3} = 2{,}222 \text{ бита},$$

что меньше чем 5 бит при равномерном алфавитном кодировании. Для английского языка такой метод кодирования дает длину кода 2,545 бита на знак. Таким образом, кодирование слов оказывается более выгодным, чем алфавитное кодирование.

Еще более эффективным окажется кодирование, если сначала установить относительную частоту появления различных слов в текстах и затем использовать код Хаффмана. Подобные исследования провел в свое время Шеннон: по относительным частотам 8 727 наиболее употребительных в английском языке слов он установил, что средняя информация на знак первичного алфавита оказывается равной 2,15 бита.

Вместо слов можно кодировать сочетания букв, т. е. блоки. Блоки можно считать словами равной длины смыслового содержания. Удлиняя блоки и применяя код Хаффмана, 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 (3-буквенных и более) избыточность стремится к нулю, в соответствии с первой теоремой Шеннона.

Пример 7. Пусть первичный алфавит A состоит из двух знаков a и b с вероятностями $p_a = 0{,}75$ и $p_b = 0{,}25$. Требуется сравнить избыточность кода Хаффмана при алфавитном и блочном 2-буквенном кодировании.

Код Хаффмана при алфавитном кодировании представлен в таблице 8.

Таблица 8 – **Код Хаффмана при алфавитном кодировании**

Знак	Вероятность, p_i	Код
a	0,75	0
b	0,25	1

Среднее значение длины кода $K(A, 2) = 1$, средняя информация на знак $I(A) = 0{,}811$ бита, избыточность $Q(A, 2) = 0{,}233$.

Код Хаффмана при блочном 2-буквенном кодировании представлен в таблице 9. Вероятности блоков p_{ij} рассчитаны по формуле

$$p_{ij} = p_i \cdot p_j,$$

где p_i и p_j – вероятности знаков a и b.

Таблица 9 – **Код Хаффмана при блочном 2-буквенном кодировании**

Знак	Вероятность, p_{ij}	Код
aa	0,562	0

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

Среднее значение длины кода $K(A, 2) = 1{,}688$ (в пересчете на знак – 0,844), средняя информация на блок $I_1^{(A)} = 1{,}623$ бита (в пересчете на 1 знак – 0,811 бита), избыточность $Q(A, 2) = 0{,}040$.

Таким образом, при использовании блочного 2-буквенного кодирования избыточность снизилась почти в 6 раз.

3.2.6. Двоичное кодирование графической информации

В процессе кодирования изображения производится его пространственная дискретизация. Этот процесс можно сравнить с построением изображения из мозаики (большого количества маленьких разноцветных стекол). Изображение разбивается на отдельные маленькие фрагменты (точки), причем каждому фрагменту присваивается значение его цвета, т. е. код цвета (красный, зеленый, синий и т. д.).

Качество кодирования изображения зависит от двух параметров:

- *размера точки* (качество кодирования изображения тем выше, чем меньше размер точки и, соответственно, большее количество точек составляет изображение);
- *количества цветов* (чем большее количество цветов или возможных состояний точки изображения используется, тем более качественно кодируется изображение; совокупность используемых в наборе цветов образует палитру цветов).

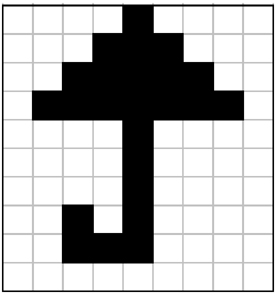
Графическая информация на экране монитора представляется в виде *растрового* изображения, которое формируется из определенного количества строк, которые, в свою очередь, содержат определенное количество точек (пикселей).

Качество изображения определяется разрешающей способностью монитора, т. е. количеством точек, из которых оно складывается. Чем больше разрешающая способность, или строк растра и точек в строке, тем выше качество изображения. В современных персональных компьютерах используются следующие разрешающие способности экрана: 800×600, 1024×768 и 1280×1024 точек и др.

В простейшем случае при черно-белом изображении без градаций серого цвета каждая точка экрана может иметь одно из двух состояний – «черная» или «белая», т. е. для хранения ее состояния необходим 1 бит (рисунок 5).

```
000010000
000111000
001111100
011111110
000010000
000010000
000010000
001010000
001110000
000000000
```

а



б

Рисунок 5 – **Кодирование черно-белого изображения:**
а – код; *б* – рисунок

Если черно-белое изображение содержит оттенки серого цвета, то требуется больше бит для кодирования оттенка. Два бита позволяют закодировать 4 оттенка точек ($2^2 = 4$): код «00» – белый цвет, код «01» – светло-серый, код «10» – темно-серый, код «11» – черный цвет. Три бита – 8 оттенков ($2^3 = 8$) и т. д.

Цветное изображение на экране монитора формируется за счет смешивания трех базовых цветов: красного, зеленого и синего. Такая цветовая модель называется RGB-моделью по первым буквам английских названий цветов Red, Green, Blue.

Цветные изображения формируются в соответствии с двоичным кодом цвета каждой точки, хранящимся в видеопамяти. Цветные изображения могут иметь различную глубину цвета, которая задается количеством битов, используемым для кодирования цвета точки.

Если все три составляющие имеют одинаковую интенсивность (глубина цвета = 1 бит), то из их сочетаний можно получить 8 различных цветов (2^3), т. е. 3 бита позволяют закодировать 8 оттенков. В таблице кодировки 8-цветной палитры (таблица10) наличие базового цвета обозначено «1», а отсутствие – «0».

Таблица 10 – **Двоичный код 8-цветной палитры**

Цвет	Красный	Зеленый	Синий

Черный	0	0	0
Синий	0	0	1
Зеленый	0	1	0
Бирюзовый	0	1	1
Красный	1	0	0
Сиреневый	1	0	1
Оливковый	1	1	0
Белый	1	1	1

Если все три составляющие имеют разную интенсивность, то можно получить большее количество оттенков. Наиболее распространенными значениями глубины цвета являются 8, 16, 24 или 32 бита. Например, в случае когда каждый цвет кодируется 8 битами, можно использовать 256 вариантов интенсивности каждого цвета ($2^8 = 256$).

4. ПРЕДСТАВЛЕНИЕ И ОБРАБОТКА ЧИСЕЛ В КОМПЬЮТЕРЕ

4.1. Системы счисления

Одним из основных направлений применения компьютеров были и остаются разнообразные вычисления. Обработка числовой информации ведется и при решении задач, на первый взгляд не связанных с какими-то расчетами, например при использовании компьютерной графики или звука. В связи с этим встает вопрос о выборе оптимального представления чисел в компьютере.

Представление определяет не только способ записи данных (букв или чисел), но и допустимый набор операций над ними. Например, буквы можно только разместить в некоторой последовательности (или исключить из нее) без изменения их самих, над числами же возможны операции, изменяющие само число (извлечение корня или сложение с другим числом).

Представление чисел в компьютере по сравнению с формами, известными со школы, имеет два важных отличия:

- числа записываются в двоичной системе счисления;
- для записи и обработки чисел отводится конечное количество разрядов.

Любое число имеет *значение* и *форму представления.* Значение числа задает его отношение к значениям других чисел (>, <, =). Форма представления определяет порядок записи числа с помощью предназначенных для этого знаков. *Значение числа не зависит от способа его представления.* Способ представления числа определяется *системой счисления*.

Система счисления *– это правило записи чисел с помощью заданного набора специальных знаков – цифр*.

Исторически использовались различные способы записи чисел, которые можно объединить в группы: *унарная, непозиционная* и *позиционная*.

Унарная система счисления – это система, 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. Обозначение числа в унарной системе – Z_1.

Непозиционной системой счисления называется система, в которой базовые числа обозначены символами (словами или знаками). Все другие числа строятся из комбинаций базовых. В непозиционных системах счисления величина не зависит от положения символа в записи числа, которую она обозначает.

Из непозиционных наиболее распространенной можно считать *римскую* систему счисления. В ней базовые числа обозначены латинскими буквами: 1 – I, 5 – V, 10 – X, 50 – L, 100 – C, 500 – D, 1000 – M (например, XIX – 19, MDXLIX – 1549).

В такой системе неудобно записывать числа и выполнять простые арифметические операции. Отсутствие нуля и знаков для чисел больше M не позволяют римскими цифрами записать любое число. Римская система используется для нумерации.

Позиционной системой счисления называется система, в которой значение каждой цифры в изображении числа определяется ее позицией в ряду других цифр. Наиболее распространенной является система счисления, в которой для записи чисел используется 10 цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8 и 9.

Число представляет собой краткую запись многочлена, в который входят степени некоторого другого числа – *основания системы счисления*. Например:

$$272{,}12 = 2 \cdot 10^2 + 7 \cdot 10^1 + 2 \cdot 10^0 + 1 \cdot 10^{-1} + 2 \cdot 10^{-2}.$$

В данном числе цифра 2 встречается трижды, но значения этих цифр различны и определяются их положением (позицией) в числе, а основанием системы счисления является число 10. Такая система счисления называется десятичной. Количество цифр для построения чисел равно основанию системы счисления. Максимальная цифра на единицу меньше основания.

Десятичная система счисления не единственная позиционная система. Существуют и используются на практике системы счисления с другими основаниями. Происхождение и использование систем счисления описано в приложении Б.

В унарной и непозиционной системах счисления значение числа определяется с помощью сложения и вычитания базисных цифр, из которых составлено число, независимо от их позиции в числе. Такая система называется *аддитивной*.

Позиционное представление считается *аддитивно-мультипликативным*, так как значение числа определяется операциями умножения и сложения. Особенность позиционного представления заключается в том, что с *помощью конечного набора знаков можно записать неограниченное количество различных чисел.* В позиционной системе легко осуществляются операции умножения и деления. Поэтому человеком и компьютером в основном используется позиционная система.

По принципу десятичной системы счисления можно построить системы с другим основанием. Пусть p – основание системы счисления. Вещественное число X_p можно представить в виде суммы целого числа и правильной дроби:

$$Z_p = a_{k-1} p^{k-1} + a_{k-2} p^{k-2} + \ldots + a_1 p^1 + a_0 p^0 + a_{-1} p^{-1} + \ldots + a_{-n} p^{-n}, \qquad (13)$$

где $Z < p^k$;

k – число цифр целой части;

n – число цифр дробной части;

a_i – целые числа;

$0 \le a_i \le p - 1$.

Каждая степень p^i ($i = k - 1, k - 2, \ldots, 1, 0, -1, \ldots, -n$) в такой записи называется разрядом (позицией), старшинство разрядов и соответствующих им цифр определяется значением показателя степени i.

Из коэффициентов a_i при степенях основания строится сокращенная запись числа:

$$Z_p = (a_{k-1}a_{k-2}\dots\ a_1a_0a_{-1}\dots\ a_{-n}).$$

Можно определить минимальное значение основания системы счисления p. Если p = 1, тогда все a_i = 0, и формула (13) теряет смысл. Первое допустимое значение p = 2 является минимальным для позиционной системы. Система счисления с основанием 2 называется двоичной. Цифрами являются 0 и 1, а формула (13) строится по степеням числа 2.

4.2. Представление чисел в различных системах счисления

4.2.1. Перевод целых чисел из одной системы счисления в другую

Целое число Z_p можно представить в следующем виде:

$$Z_p = a_{k-1}p^{k-1} + a_{k-2}p^{k-2} + \dots + a_1p^1 + a_0p^0. \qquad (14)$$

Перевод целого числа из системы с основанием p в систему с основанием q обозначается $Z_p \to Z_q$. Его возможно произвести при любых q и p, но он затруднен тем, что придется выполнять операции по правилам арифметики недесятичных систем счисления. Поэтому удобными являются следующие преобразования:

$$Z_p \to Z_r \to Z_q,$$

где Z_r – число в системе счисления с основанием r, для которого арифметические операции выполнить легко; такими основаниями являются 1 и 10, наиболее удобным основанием является 10.

При преобразовании $Z_p \to Z_{10} \to Z_q$ первая и вторая часть преобразования не связаны друг с другом, поэтому рассмотрим их по отдельности.

Алгоритм преобразования $Z_{10} \to Z_q$ следующий:

1. Исходное число Z_{10} целочисленно разделить на основание новой системы счисления q и найти остаток от деления – это будет цифра нулевого разряда числа Z_q.

2. Частное от деления снова целочисленно разделить на q с выделением остатка. Процедуру продолжать до тех пор, пока частное от деления не окажется меньше Z_q.

3. Образовавшиеся остатки от деления поставить в порядок, об-

ратный порядку их получения. Они и будут представлять Z_q.

Пример 8. Выполнение преобразования $123_{10} \to Z_5$.

Для перевода числа 123_{10} в пятеричную систему счисления следует выполнить преобразование, как показано на рисунке 6.

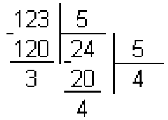


Рисунок 6 – **Преобразование $\mathbf{123_{10} \to Z_5}$**

Остатки от деления 3, 4 и результат последнего целочисленного деления 4 образуют обратный порядок цифр нового числа. Таким образом, $123_{10} = 443_5$[*].

Алгоритм перевода $Z_p \to Z_{10}$ следующий:

1. Необходимо Z_p представить в форме многочлена (14).
2. Выполнить все операции по правилам десятичной арифметики.

Пример 9. Выполнение преобразования $443_5 \to Z_{10}$.

Данное преобразование осуществляется следующим образом:

$$443_5 = 4 \cdot 5^2 + 4 \cdot 5^1 + 3 \cdot 5^0 = 4 \cdot 25 + 4 \cdot 5 + 3 \cdot 1 = 123_{10}.$$

4.2.2. Перевод дробных чисел из одной системы счисления в другую

Простая дробь может быть представлена в следующем виде:

$$0,Z_p = a_{-1}p^{-1} + \ldots + a_{-n}p^{-n}. \tag{15}$$

Перевод простой дроби из системы с основанием p в систему с основанием q обозначается $0,Z_p \to 0,Z_q$. Осуществлять преобразование удобно через промежуточный переход к десятичной системе, т. е. $0,Z_p \to 0,Z_{10} \to 0,Z_q$. Это, в свою очередь, разбивает задачу на две составляющие: преобразование $0,Z_p \to 0,Z_{10}$ и $0,Z_{10} \to 0,Z_q$, каждое из которых может рассматриваться независимо.

[*] Полученное число следует читать «*число четыре, четыре, три в пятеричной системе счисления*».

Алгоритм перевода $0,Z_{10} \to 0,Z_q$ следующий:

1. Умножить исходную дробь в десятичной системе счисления на q, выделить целую часть – она будет первой цифрой новой дроби; отбросить целую часть.

2. Для оставшейся дробной части операцию умножения с выделением целой и дробной части повторять, пока в дробной части не окажется 0 или не будет достигнута желаемая точность конечного числа. Появляющиеся при этом целые станут цифрами новой дроби.

3. Записать дробь в виде последовательности цифр после нуля с разделителем в порядке их появления.

Пример 10. Выполнение преобразования $0{,}375_{10} \to 0,Z_2$.

Для перевода числа $0{,}375_{10}$ в двоичную систему счисления следует выполнить преобразование, как показано на рисунке 7.

$$\begin{array}{r|c|l} 0{,}375 \cdot 2 = & 0 & ,750 \\ 0{,}75 \cdot 2 = & 1 & ,50 \\ 0{,}5 \cdot 2 = & 1 & ,0 \end{array}$$

Рисунок 7 – **Преобразование $0{,}375_{10} \to 0,Z_2$**

Таким образом, $0{,}375_{10} = 0{,}011_2$.

Перевод $0,Z_p \to 0,Z_{10}$, как и в случае с целыми числами, сводится к вычислению значения формулы (15) в десятичной системе счисления. Например:

$$0{,}011_2 = 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0{,}25 + 0{,}125 = 0{,}375_{10}.$$

Пример 11. Выполнение преобразования $5{,}3(3)_{10} \to X_3$.

Сначала осуществляется перевод целой части, а затем – дробной:

$$5_{10} = 12_3; \quad 0{,}3(3)_{10} = 0{,}1_3.$$

Окончательный вид преобразования таков:

$$5{,}3(3)_{10} = 12{,}1_3.$$

4.2.3. Понятие экономичности системы счисления

Число в системе счисления p с k разрядами будет иметь наибольшее значение в том случае, если все цифры числа окажутся максимальными, т. е. равными $p-1$, тогда

$$(Z_p)^{\max} = \underbrace{\langle p-1\rangle \ldots \langle p-1\rangle}_{k\ \text{цифр}} = p^k - 1. \tag{16}$$

Количество разрядов числа при переходе от одной системы счисления к другой в общем случае меняется. Если $p = q^{\sigma}$ (σ – необязательно целое), то

$$(Z_p)^{\max} = p^k - 1 = q^{\sigma k} - 1,$$

т. е. количество разрядов числа в системах счисления p и q будут различаться в σ раз. Очевидно следующее соотношение:

$$\sigma = \frac{\log p}{\log q}.$$

Основание логарифма любое, так как σ определяется отношением логарифмов.

Сравним количество цифр в числе 99_{10} и его представлении в двоичной системе счисления – 1100011_2. Двоичная запись требует 7 цифр вместо 2 в десятичной. Количество разрядов числа отличается в σ раз, равное

$$\sigma = \frac{\log 10}{\log 2} = 3{,}322.$$

Для определения количества цифр в двоичном представлении числа нужно количество цифр в десятичном представлении умножить на 3,322 и округлить в большую сторону:

$$2 \cdot 3{,}322 = 6{,}644 = 7.$$

Под экономичностью системы счисления *понимают то количество чисел, которое можно записать в данной системе с помощью определенного количества цифр.*

Например, пусть имеется 12 цифр. Можно разбить их на 6 групп по 2 цифры (0 и 1) и получить 6-разрядное двоичное число. Общее количество таких чисел равно 2^6. Можно разбить заданное количество цифр на 4 группы по 3 цифры и воспользоваться троичной системой счисления – общее количество различных сочетаний составит 3^4. Аналогично можно произвести другие разбиения. При этом число групп определит разрядность числа, а количество цифр в группе – основание системы счисления. Результаты различных разбиений можно проиллюстрировать с помощью таблицы 11.

Таблица 11 – **Результаты различных разбиений 12 цифр**

Основание системы счисления (p)	2	3	4	6	12
Разрядность числа (k)	6	4	3	2	1
Общее количество различных чисел (N)	$2^6 = 64$	$3^4 = 81$	$4^3 = 64$	$6^2 = 36$	$12^1 = 12$

Наиболее экономичной оказывается троичная система счисления. Предпочтение все же отдается двоичной системе, поскольку по экономичности она оказывается второй за троичной, а технически она реализуется гораздо проще остальных.

4.2.4. Перевод чисел между системами счисления 2 ↔ 8 ↔ 16

Двоичная система используется для представления чисел в компьютере. Однако двоичная запись оказывается громоздкой и плохо воспринимается человеком. Поэтому в нумерации ячеек памяти компьютера записи кодов команд, нумерации регистров, устройств и прочего используются системы счисления с основаниями 8 и 16. Выбор этих систем счисления обусловлен тем, что переход от них к двоичной системе и обратно простой.

В двоичной системе счисления для записи чисел используется две цифры: 0 и 1.

В восьмеричной системе счисления для записи чисел используется восемь цифр: 0, 1, …, 7.

В шестнадцатеричной системе счисления для записи чисел используется 16 цифр: 0, 1, ..., 9, A, B, C, D, E, F. При этом знак «А» является шестнадцатеричной цифрой, равной 10_{10}, т. е. $A_{16} = 10_{10}$; аналогично $B_{16} = 11_{10}$; $C_{16} = 12_{10}$; $D_{16} = 13_{10}$; $E_{16} = 14_{10}$; $F_{16} = 15_{10}$.

Пользуясь алгоритмами, сформулированными выше, можно заполнить таблицу 12.

Таблица 12 – **Представление чисел в системах счисления**

Системы счисления			
десятичная	двоичная	восьмеричная	шестнадцатеричная
0	0	0	0
1	1	1	1

Системы счисления			
десятичная	двоичная	восьмеричная	шестнадцатеричная
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

Теорема 1. *Для преобразования целого числа $Z_p \to Z_q$ в том случае, если системы счисления связаны соотношением*

$$q = p^r,$$

где r – целое число больше 1, достаточно Z_p разбить справа налево на группы по r цифр и каждую из них независимо перевести в систему q.

Пример 12. Выполнение преобразований $110001_2 \to Z_8$ и $110001_2 \to$ $\to Z_{16}$.

Исходное число 110001_2 разбивается на группы по 3 разряда ($8 = 2^3$, следовательно, $r = 3$) справа налево, и каждая тройка в соответствии с таблицей 12 переводится в восьмеричную систему счисления независимо от остальных троек (рисунок 8).

$$\underbrace{110}_{6}\ \underbrace{001}_{1}$$

Рисунок 8 – **Преобразование $110001_2 \to Z_8$**

Следовательно, $110001_2 = 61_8$.

Аналогично, разбивая Z_2 на группы по 4 двоичные цифры и дополняя старшую группу незначащими нулями слева, получим $110001_2 = 31_{16}$.

Теорема 2. *Для преобразования целого числа $Z_p \to Z_q$ в том случае, если системы счисления связаны соотношением*

$$p = q^r,$$

где r – целое число больше 1, достаточно каждую цифру Z_p заменить соответствующим r-разрядным числом в системе счисления q, дополняя его при необходимости незначащими нулями слева до группы в r цифр.

Пример 13. Выполнение преобразования $D3_{16} \to Z_2$.

Как следует из таблицы 12, $D_{16} = 1101_2$, $3_{16} = 0011_2$ (рисунок 9).

$$\underbrace{1101}_{D}\ \underbrace{0011}_{3}$$

Рисунок 9 – **Преобразование $D3_{16} \to Z_2$**

Следовательно, $D3_{16} = 11010011_2$.

Переходы $Z_8 \to Z_{16}$ и $Z_{16} \to Z_8$ удобнее осуществлять через промежуточный переход к двоичной системе. Например:

$$123_8 = 001\ 010\ 011_2 = 53_{16}.$$

4.2.5. Преобразование нормализованных чисел

Вещественное число X может быть представлено в двух формах – естественной и нормализованной. *В естественной форме* у числа X имеется целая и дробная части, между которыми помещается разделитель «,», например 123,4567. Такая форма записи называется *представление числа с фиксированной запятой*. Она неудобна для слишком больших или слишком малых чисел. Использование такой формы в компьютере вызвало бы снижение точности вычислений из-за необходимости приведения в соответствие разрядов обрабатываемых чисел и связанных с этим округлений или могло бы породить ситуацию, называемую *переполнением*, когда старший разряд числа не умещается в отведенной разрядной сетке. Поэтому вещественные числа в компьютере представляются *в нормализованном виде*, который называется *представление числа с плавающей запятой*.

Главным достоинством нормализованного вида является автоматическое масштабирование числа на каждом этапе обработки. Это обеспечивает максимально возможную точность вычислений, а также избавляет от необходимости принимать меры по предотвращению переполнения. Нормализованная форма – это универсальная форма записи всех чисел, кроме типа, определенного как Integer.

Число X_p называется ***нормализованным****, если оно представлено в виде*

$$X_p = \pm M_p \cdot p^{\pm k_p}, \tag{17}$$

где M_p – ***мантисса***[1] *нормализованного числа, значения которой лежат в интервале $p^{-1} < M_p < 1$ (первая значащая цифра мантиссы всегда ненулевая);*

k_p – записанный в системе p порядок нормализованного числа, которое является целым положительным числом.

Для $p = 10$ выражение (17) принимает вид

$$X_{10} = \pm M_{10} \cdot 10^{\pm k_{10}},$$

где значения мантиссы лежат в интервале $0,1 < M_{10} < 1$. Например:

$$-1234_{10} = -0,1234_{10} \cdot 10^4;$$
$$0,03456_{10} = 0,3456_{10} \cdot 10^{-1}.$$

Для $p = 2$ выражение (17) принимает вид

$$X_2 = \pm M_2 \cdot 2^{\pm k_2},$$

где значения мантиссы лежат в интервале $0,1_2 < M_2 < 1_2$. Например:

$$-101,01_2 = -0,1010_2 \cdot 2^{11}.$$

При нормализации происходит разделение «составляющих» числа с выделением *знака числа*, *мантиссы*, *знака порядка и порядка* – это создает определенные удобства при хранении и обработке чисел в компьютере.

В компьютере все вещественные числа хранятся и обрабатываются в нормализованном двоичном представлении. Следовательно, при их вводе осуществляется сначала преобразование вещественного числа из естественной формы к нормализованному виду, а затем перевод чисел в нормализованной форме $X_{10} \rightarrow X_2$, а при выводе – обратный

[1] Мантисса («прибавка») – дробная часть десятичного логарифма числа.

перевод чисел в нормализованной форме $X_2 \to X_{10}$ и преобразование вещественного числа из нормализованного вида к естественной форме.

Рассмотрим, как это производится. При нормализации различаются следующие ситуации:

• При $X_p > 1$ для нормализации необходимо перемещать разделитель разрядов *влево* по числу до тех пор, пока не исчезнет целая часть числа, но первая цифра после разделителя будет ненулевой. Каждое перемещение разделителя на 1 разряд влево эквивалентно делению числа на p, и, чтобы число не менялось, показатель должен возрастать на единицу при каждом сдвиге. Если обозначить эту операцию $N_{\leftarrow}$ (нормализация влево), то $N_{\leftarrow}[(123{,}45)_{10}] = 0{,}12345 \cdot 10^3$.

• При $X_p < p^{-1}$ проводится нормализация вправо $N_{\rightarrow}$, обеспечивающая нормализацию чисел, меньших p. Такие числа необходимо *умножать* на p с одновременным уменьшением показателя на 1 до тех пор, пока первая цифра после разделителя станет ненулевой. Например: $N_{\rightarrow}[(0{,}000987)_{10}] = 0{,}987 \cdot 10^{-3}$.

При переводе нормализованного числа $X_p = \pm M_p \cdot p^{\pm k_p}$ из одной системы счисления в другую необходимо найти соответствующее ему $X_q = \pm M_q \cdot q^{\pm k_q}$.

Это преобразование не затронет знаков мантиссы и показателя степени. Таким образом, для осуществления преобразования необходимо установить соответствие между (M_p, k_p) и (M_q, k_q). Оно получается достаточно просто исходя из того, что $X_p = X_q$, откуда

$$M_q = M_p \frac{p^{k_p}}{q^{k_q}}. \tag{18}$$

Из формулы (18) следует, что для осуществления преобразования можно M_p умножить на p^{k_p}, т. е. перейти к естественной форме числа в системе p, перевести его в систему q, а затем нормализовать.

Однако в таком варианте действий теряется точность числа и возможно переполнение на промежуточных этапах преобразования. Во избежание этого необходимо чередовать умножение (или деление) на p и нормализацию по основанию q. При этом, поскольку все операции выполняются по правилам арифметики в системе p, будут получены не (M_q, k_q) в окончательном варианте, а их представления в системе p – обозначим их $(M_q)_p$ и $(k_q)_p$, которые затем нужно будет перевести в систему q.

Различаются также ситуации $k_p \geq 0$ и $k_p < 0$. В первом случае необходимо умножать начальное и промежуточные значения мантиссы на

p и для нормализации делить на q, во втором – наоборот. Каждый раз при умножении или делении на p показатель k_p будет менять свое значение на 1. Продолжать действия следует до тех пор, пока не выполнится условие $k_p = 0$.

Пример 14. Выполнение преобразования $X_{10} = 16{,}5_{10} \to X_2$.

Перевод можно осуществить для целой и дробной части по отдельности, а затем их объединить:

$$16_{10} = 10000_2; \quad 0{,}5_{10} = 0{,}1_2,$$

следовательно,

$$16{,}5_{10} = 10000{,}1_2 = 0{,}100001 \cdot 2_2^{101}.$$

Пример 15. Выполнение преобразования $X_2 = 0{,}11 \cdot 2_2^{110} \to X_{10}$.

Так как

$$0{,}11_2 = 0{,}75_{10}; \quad 2_2^{110} = 2_{10}^{6} = 64_{10},$$

то получим

$$0{,}11 \cdot 2_2^{110} = 0{,}75 \cdot 64 = 48_{10}.$$

4.3. Кодирование чисел в компьютере и действия над ними

Компьютерные ячейки имеют ограниченный размер и вынуждают использовать при записи чисел и действиях с ними конечное количество разрядов. Поэтому бесконечное множество вещественных чисел заменяется конечным множеством их представлений (кодами чисел), а арифметические операции с числами заменяются операциями с кодами.

Способы кодирования и допустимые над ними действия оказываются различными для следующих чисел:

- целых чисел без знака;
- целых чисел со знаком;
- вещественных нормализованных чисел.

4.3.1. Кодирование и обработка в компьютере целых чисел без знака

Для записи числа в устройствах компьютера выделяется фиксированное количество двоичных разрядов. Память компьютера имеет байтовую структуру, размер одной адресуемой ячейки обычно составляет несколько байт. Например, в компьютерах IBM ячейка памяти объединяет 2 байта или 16 бит. Комбинация связанных соседних ячеек, обрабатываемая совместно, называется машинным словом. Для представления числа в регистре арифметико-логического устройства процессора, где формируется результат операции, имеется еще один дополнительный одноразрядный регистр, который называется *регистром переноса* и который можно рассматривать в качестве продолжения (т. е. 17-го бита) регистра результата.

Конечный размер разрядной сетки порождает понятие *наибольшее целое число,* которого в обычном представлении чисел не существует.

Если количество разрядов k и $p = 2$, то согласно формуле (16)

$$(Z_2)^{\max} = 2^k - 1.$$

В частности, при $k = 16$

$$(Z_2)^{\max} = 2^{16} - 1 = 1111\ 1111\ 1111\ 1111_2 = 65535_{10}.$$

Таким образом, целого числа 65636 и более в компьютере просто не может существовать. Появление в ходе вычислений чисел, превышающих $(Z_2)^{\max}$, должно интерпретироваться как ошибка. Минимальным целым числом в беззнаковом представлении является

$$(Z_2)^{\min} = 0000\ 0000\ 0000\ 0000_2 = 0_{10}.$$

Выход за границу 65535 возможен только путем увеличения количества разрядов для записи числа. Это порождает новый тип со своим $Z^{\max}$, например с максимальным значением 2147483647_{10}, числа которого занимают 4 байта.

К арифметическим операциям с беззнаковыми числами, не меняющими типа числа, относятся сложение и умножение.

Сложение беззнаковых чисел производится согласно таблице сложения, которая для двоичных чисел имеет следующий вид:

$$0 + 0 = 0; \qquad 0 + 1 = 1;$$
$$1 + 0 = 1; \qquad 1 + 1 = 10.$$

В последнем случае в том разряде, где находились слагаемые, оказывается нуль, а единица переносится в старший разряд. Место, где сохраняется переносимая в старший разряд единица, до того как она будет использована в операции, называется *битом переноса*.

Пример 16. Поиск суммы $1594_{10} + 17563_{10}$ при беззнаковой двоичной кодировке и 16-битном машинном слове.

Слагаемые в двоичной системе счисления и выполнение сложения представлены на рисунке 10.

Переносы →		1 11
	+	0010 0110 1001 0100
Регистр		0011 0000 0011 1001
переполнения →	$\boxed{0}$	0101 0110 1100 1101

Рисунок 10 – **Сложение чисел при беззнаковой двоичной кодировке и 16-битном машинном слове (регистр переполнения равен нулю)**

Пример 17. Поиск суммы $65534_{10} + 3_{10}$.

Слагаемые в двоичной системе счисления и выполнение сложения представлены на рисунке11.

Переносы →		1111 1111 1111 11
	+	1111 1111 1111 1110
Регистр		0000 0000 0000 0011
переполнения →	$\boxed{1}$	0000 0000 0000 0001

Рисунок 11 – **Сложение чисел при беззнаковой двоичной кодировке и 16-битном машинном слове (регистр переполнения равен единице)**

В примере 17 в результате сложения получилось число, превышающее максимально возможное. Результат ошибочен, о чем свидетельствует появление единицы в регистре переполнения. В программах, предназначенных для обработки числовой информации (Excel, MathCAD), при переполнении разрядной сетки производится автоматическое преобразование целого числа в вещественный тип, т. е. регистр переноса в данном случае служит индикатором корректности процесса вычислений.

Умножение беззнаковых чисел производится согласно таблице умножения, которая для двоичных чисел имеет вид:

$$0 \cdot 0 = 0; \qquad 0 \cdot 1 = 0;$$
$$1 \cdot 0 = 0; \qquad 1 \cdot 1 = 1.$$

Пример 18. Поиск произведения $13_{10} \cdot 5_{10}$ в двоичной системе счисления.

Слагаемые в двоичной системе счисления и выполнение умножения представлены на рисунке 12.

$$
\begin{array}{r}
1101\\
\times\ \ 101\\
\hline
1101\\
+\ 1101\\
\hline
1000001
\end{array}
$$

Рисунок 12 – **Умножение двоичных чисел**

Так, $13_{10} \cdot 5_{10} = 1000001_2$.

Умножение двоичных чисел сводится к операциям сдвига на один двоичный разряд влево и повторения первого сомножителя в тех разрядах, где второй сомножитель содержит единицу, и сдвига без повторения в разрядах с нулем. Сдвиг всегда чередуется со сложением из-за ограниченности числа регистров, которые имеются в процессоре для размещения операндов. Таким образом, реализации отдельной операции умножения в процессоре не требуется.

Как и в операции сложения, при умножении чисел с ограниченной разрядной сеткой может возникнуть переполнение. Решается проблема рассмотренными выше способами.

4.3.2. Кодирование и обработка в компьютере целых чисел со знаком

Кодирование целых чисел со знаком можно осуществить двумя способами. В первом варианте один (старший) разряд в машинном слове отводится для записи знака числа. Знак «+» кодируют *нулем*, знак «–» – *единицей*. Под запись самого числа остается 15 двоичных разрядов, что обеспечивает наибольшее значение числа:

$$Z^{\max} = 2^{15} - 1 = 32767_{10}.$$

Такое представление чисел называется *прямым кодом*, однако его применение усложняет порядок обработки чисел. Например, операция сложения двух чисел с разными знаками должна быть заменена операцией вычитания меньшего из большего с последующим присвоением результату знака большего по модулю числа. Другими словами, операция сопровождается большим количеством проверок условий и выработкой признаков, в соответствии с которыми выбирается то или иное действие.

Альтернативным вариантом является представление чисел со знаком *в дополнительном коде.* Идея построения дополнительного кода достаточно проста: на оси целых положительных чисел, помещающихся в машинное слово ($0 \div 65535$), нужно сместить положение «0» на середину интервала. Числа, попадающие в первую половину ($0 \div 32767$), считаются положительными, а числа из второй половины ($32768 \div 65535$) – отрицательными. В этом случае судить о знаке числа можно будет по его величине, и в явном виде выделение знака не потребуется.

Например, $1000\,0000\,0000\,0001_2 = 32769_{10}$ – код отрицательного числа, а $0000\,0000\,0000\,0001_2 = 1_{10}$ – код положительного.

Принадлежность к интервалу кодов положительных или отрицательных чисел видна по состоянию старшего бита – у кодов положительных чисел его значение «0», отрицательных – «1». Это напоминает представление со знаком, но не является таковым, поскольку используется другой принцип кодирования. Его применение позволяет заменить вычитание чисел их суммированием в дополнительном коде.

Перевод в дополнительный код происходит автоматически при вводе чисел. В таком виде числа хранятся в оперативном запоминающем устройстве и затем участвуют в арифметических операциях.

При выполнении вычитания отрицательного числа оно из дополнительного кода переводится в прямой, и вновь вместо вычитания производится сложение. Подобным же образом число из дополнительного кода переводится в прямой при выполнении операции умножения.

Над множеством целых чисел со знаком операция деления не определена, поскольку в общем случае ее результатом будет вещественное число. Однако допустимыми являются операции целочисленного деления и нахождения остатка от целочисленного деления.

4.3.3. Кодирование и обработка в компьютере вещественных чисел

Ситуация меняется при представлении и обработке вещественных чисел. Вещественные числа образуют непрерывное множество, т. е. два числа могут находиться сколь угодно близко друг к другу, и на любом отрезке содержится бесконечно много значений чисел. В машинном представлении количество возможных значений чисел *конечно*, для двоичной системы счисления оно определяется как 2^k, где k – количество двоичных разрядов в представлении мантиссы. Вещественные числа в компьютере заменяются их *кодами,* которые образуют *конечное дискретное множество*, каждый код оказывается представителем целого интервала значений континуума.

Так как основной формой представления кодов вещественных чисел в компьютере является двоичная нормализованная форма, то в памяти компьютера должны записываться и храниться все составляющие нормализованной формы (знак числа, мантисса, знак порядка и порядок), что требует нескольких ячеек памяти. Непосредственное распределение компонентов нормализованного числа по разрядам определяется конструктивными особенностями компьютера и программным обеспечением. Пример размещения числа в двух ячейках памяти (32 разряда) представлен на рисунке 13.

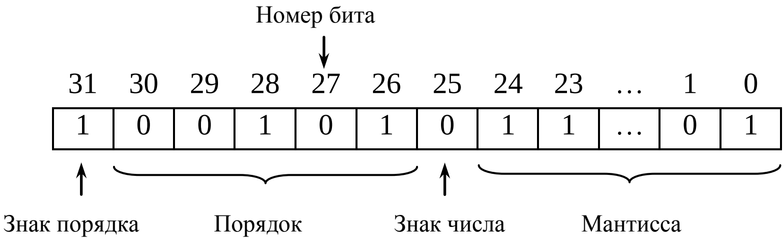


Рисунок 13 – **Размещение числа в двух ячейках памяти**

В процессе выполнения арифметических действий с нормализованными числами отдельно обрабатываются мантиссы и порядки.

Пусть имеются два числа $X_1 = M_1 \cdot p^{k_1}$ и $X_2 = M_2 \cdot p^{k_2}$ (здесь индексы у мантиссы и порядка означают не систему счисления, а служат номерами чисел). Сложение нормализованных чисел должно начинаться с выявления большего из k_1 и k_2, нахождения модуля их разности $\Delta k = |k_1 - k_2|$ и сдвига *вправо* на Δk разрядов мантиссы того числа, у которого k оказался меньше. После этого выполняется сложение мантисс. Порядку результата присваивается значение большего из имеющихся, и при необходимости производится нормализация результата. При сдвиге вправо мантиссы меньшего числа происходит потеря Δk младших значащих цифр, что приводит к появлению погрешности сложения.

Вычитание нормализованных чисел, как и целых, не является самостоятельной операцией и сводится к сложению с дополнительным кодом числа.

Умножение нормализованных чисел производится в соответствии с правилами: мантисса произведения вычисляется по формуле

$$M = M_1 \cdot M_2,$$

а порядок – по формуле

$$k = k_1 + k_2,$$

при необходимости полученное число нормализуется.

Деление приводит в общем случае к появлению вещественного числа, поэтому целые числа предварительно преобразуются в вещественный тип, т. е. переводятся в нормализованную форму. При делении мантисса частного вычисляется по формуле

$$M = \frac{M_1}{M_2},$$

а порядок – по формуле

$$k = k_1 - k_2.$$

Результат деления при необходимости нормализуется.

Время выполнения операций с кодами вещественных чисел в форме с плавающей запятой гораздо больше, чем с целыми числами или числами с фиксированной запятой. По этой причине для ускорения обработки на компьютерах IBM с процессорами Intel 80286 и 80386 ставились сопроцессоры. В современных компьютерах команды обработки вещественных чисел включены в перечень команд центрального процессора.

5. ПЕРЕДАЧА ИНФОРМАЦИИ

5.1. Характеристики канала связи

Рассмотрим каналы связи, передача сообщений по которым осуществляется за счет электрических импульсов.

Ширина полосы пропускания

Любой преобразователь, работа которого основана на использовании колебаний, может формировать и пропускать сигналы из ограниченной области частот. В радио и телевизионной связи 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****.*

Для построения теории важна не сама ширина полосы пропускания, а максимальное значение частоты из данной полосы (ν_m), так как именно она определяет возможную скорость передачи информации по каналу.

Длительность элементарного импульса

Длительность элементарного импульса определяют из следующих соображений.

Если параметр сигнала меняется синусоидально, то за один период колебания (T) сигнал будет иметь одно максимальное и одно минимальное значение.

Если аппроксимировать синусоиду прямоугольными импульсами и сместить начало отсчета на уровень минимального значения, получится, что сигнал принимает всего два значения: максимальное (обозначается «1») – *импульс*, и минимальное («0») – *пауза*. Импульс и пауза – элементарные сигналы (рисунок 14).

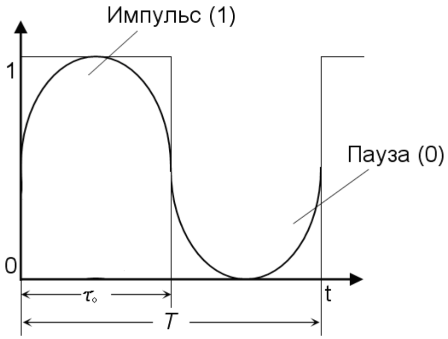


Рисунок 14 – **Аппроксимация синусоидального сигнала прямоугольными импульсами**

При выбранной аппроксимации их длительности одинаковы:

$$\tau_0 = \frac{T}{2} = \frac{1}{2\nu_m}.$$

Если же импульсы порождаются тактовым генератором, имеющим частоту ν_m, то

$$\tau_0 = \frac{1}{\nu_m}. \quad (19)$$

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, $2\tau_0$), не приведет к потере информации, но снизит скорость передачи. Использование сигналов более коротких, чем τ_0, может привести к информационным потерям, так как тогда сигналы будут принимать промежуточные значения между минимальным и максимальным, что затруднит их интерпретацию.

Таким образом, ν_m определяет *длительность элементарного сигнала* τ_0, используемого для передачи сообщения.

Пропускная способность канала связи

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*:

$$C = \frac{I_{imp}}{\tau_0}. \tag{20}$$

Единица измерения пропускной способности канала – бит/с.

Величину I_{imp} в формуле (20) можно установить из следующих соображений: если первичный алфавит содержит N знаков с вероятностями появления их в сообщении p_i, то по формуле Шеннона (8) можно найти среднюю информацию на знак первичного алфавита I_1, для представления которого используется двоичный код длиной $K^{(2)}$:

$$I_{imp} = \frac{I_1}{K^{(2)}}. \tag{21}$$

При подстановке формулы (21) в формулу (20) получим:

$$C = \frac{I_1}{K^{(2)}\tau_0}. \tag{22}$$

Пример 19. 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 используется равномерный двоичный код. Частота тактового генератора – 500 Гц. Необходимо выяснить пропускную способность канала.

Число знаков первичного алфавита равно 3. По формуле Шеннона (8) находим среднюю информацию:

$$I_1 = -(0{,}2\log_2 0{,}2 + 0{,}7\log_2 0{,}7 + 0{,}1\log_2 0{,}1) = 1{,}16 \text{ бита};$$

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

Затем, подставив данные в формулу (22), получаем

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

Скорость передачи информации

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

$$J = \frac{I}{t}.$$

Единица измерения скорости передачи информации – бит/с.

Поскольку τ_0 – минимальная длительность элементарного сигнала, то величина пропускной способности канала соответствует *максимальной скорости передачи информации* по данной линии связи:

$$J \le C \text{ или } C = J_{\max}.$$

Таким образом, максимальная скорость передачи информации по каналу связи равна его пропускной способности.

5.2. Влияние шумов на пропускную способность канала

В реальном канале связи, в отличие от идеального, всегда есть помехи или шумы, которые приводят *к снижению пропускной способности канала*.

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

В реальном канале из-за шумов при передаче может произойти искажение сигнала, в результате чего вместо «1» на выходе может быть 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 (3), может быть охарактеризована средней энтропией:

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

Эта неопределенность приведет к уменьшению количества информации, содержащейся в сигнале, на величину H:

$$(I_{imp})' = I_{imp} - H.$$

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

$$C_R = \frac{(I_{imp})'}{\tau_0} = \frac{I_{imp} + p \cdot \log_2 p + (p-1) \cdot \log_2 (p-1)}{\tau_0} \leq C. \qquad (23)$$

На нижеприведенном графике (рисунок 15) показана зависимость $C_R(p)$.

Как следует из выражения (23), при p = 0,5 получим $(I_{imp})' = 0$ и $C_R = 0$. Это связано с тем, что в данном случае на приемном конце линии связи с одинаковой вероятностью получаются «0» и «1» независимо от того, какой был сигнал на входе, поэтому передача информации по такой линии оказывается вообще невозможной.

Максимального значения пропускная способность достигает при $p = 0$, что соответствует отсутствию помех, а также при $p = 1$, 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 $C > C_R$.

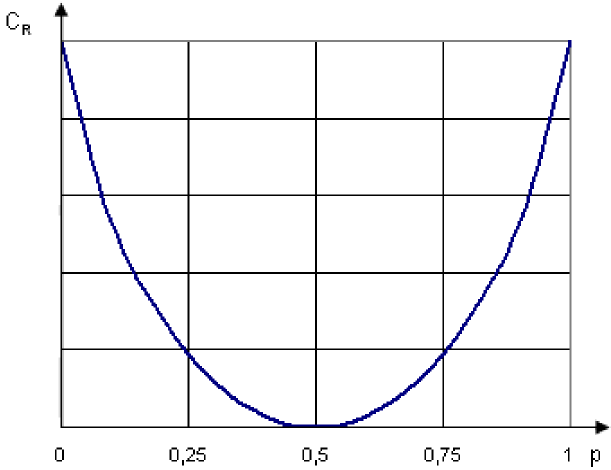


Рисунок 15 – **Пропускная способность реального канала при наличии помех**

Пример 20. Необходимо выяснить, каково отношение C_R к C, если средняя частота появления ошибки при передаче сообщения по линии связи с шумом составляет 1 ошибочный знак на 100 переданных при использовании двоичного кодирования.

Вероятность появления ошибки передачи p = 0,01. Поскольку код двоичный (I_{imp} = 1 бит), то из формул (20) и (23) получаем

$$\frac{C_R}{C} = 1 + 0{,}01 \cdot \log_2 0{,}01 + 0{,}99 \cdot \log_2 0{,}99 = 0{,}92.$$

Таким образом, пропускная способность канала снизилась приблизительно на 8%.

5.3. Обеспечение надежности передачи и хранения информации

5.3.1. Постановка задачи

Все реальные каналы связи подвержены воздействию помех. Шеннон доказал *теоретическую возможность* передачи сообщения без потерь информации по реальным каналам, если при этом выполнен ряд условий.

Вторая теорема Шеннона. *При передаче информации по каналу с шумом всегда имеется 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.

Оценим, насколько велика вероятность возникновения ошибки при хранении информации. Пластмассовые корпуса микросхем памяти компьютеров содержат в малых количествах тяжелые элементы, которые в результате радиоактивного распада испускают α-частицы. Попадая в полупроводниковые элементы памяти, они могут вызвать изменение их состояния, т. е. искажение хранимой информации. Лабораторные эксперименты показали, что «время наработки на отказ» отдельно взятого элемента превышает миллион лет. Общее количество подобных элементов в памяти компьютера большое. Память емкостью 1 Мбайт содержит 8 388 608 двоичных элементов, тогда среднее время появления ошибки

$$t = \frac{10^6}{8 \cdot 10^6} = \frac{1}{8}\,\text{года} = 45\,\text{дней}.$$

Если память персонального компьютера составляет 32–64 Мбайта, то t сокращается до 1 дня. Экономичных технических способов исключения влияния этих α-частиц нет, поэтому проблема защиты информации от искажения даже при ее хранении актуальна.

Решение проблемы состоит в использовании таких методов кодирования информации, которые позволили бы контролировать правильность передачи и при обнаружении ошибки – исправлять ее. При этом можно в качестве самостоятельных рассмотреть две ситуации:

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

Общим условием является использование равномерных кодов. Надежность обеспечивается тем, что вместе с информационными битами, кодирующими сообщение, 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, что $k > k_i$. Подобно введенной ранее величине Q, характеризующей относительную избыточность кода при передаче по идеальному каналу, определим *избыточность сообщения* (L) для реального канала.

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

$$L = \frac{k}{k_i} = \frac{k_i + k_c}{k_i} = 1 + \frac{k_c}{k_i}. \quad (24)$$

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

5.3.2. Коды, обнаруживающие ошибку

Задача обнаружения ошибки может быть решена довольно легко. Достаточно передавать каждую букву сообщения дважды.

Например, слово «гора» можно передать как «ггоорраа». При получении сообщения «гготрраа» можно догадаться, каким было исходное слово. 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 битов длиной k_i. Добавим к ним один контрольный бит k_c, значение которого определяется тем, что новая кодовая цепочка $k_i + k_c = k_i + 1$ бит должна содержать *четное количество единиц*. 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 будет равен 0.

В случае одиночной ошибки передачи число 1 перестает быть четным, что и служит свидетельством сбоя. Например, если получена цепочка $\underline{1}10110111$, где контрольный бит выделен подчеркиванием, то становится 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, рассчитанная по формуле (24), равна

$$L = 1 + \frac{k_c}{k_i} = 1 + \frac{1}{8} = 1{,}125.$$

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

5.3.3. Коды, исправляющие одиночную ошибку

По аналогии с предыдущим подразделом можно было бы предложить простой способ установления ошибки – передавать каждый символ трижды, например «гггоооррраaa», тогда при получении сообщения «гггоооnрраaa» ясно, что ошибочной оказывается буква «п» и ее следует заменить на «р». При этом предполагается, что вероятность появления парной ошибки невелика. Такой метод кодирования приводит к избыточности сообщения: L = 3, – что неприемлемо с экономической точки зрения.

Произведем количественные оценки. Шумы в канале связи ведут к частичной потере передаваемой информации на величину возникающей неопределенности H, которая при передаче одного бита исходного сообщения составляет

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

Для восстановления информационного содержания сообщения следует дополнительно передать количество информации не менее величины ее потерь, т. е. вместо передачи каждого бита информации следует передавать 1 + H бит. В этом случае избыточность сообщения составит

$$L_{\min} = \frac{1+H}{1} = 1-(p \cdot \log_2 p + (p-1) \cdot \log_2 (p-1)). \quad (25)$$

Избыточность $L_{\min}$ считается *минимальной*, так как при передаче сообщения по каналу, характеризуемому вероятностью искажения p, при избыточности меньше $L_{\min}$ восстановление информации оказывается невозможным.

Пример 21. Необходимо определить, какое минимальное количество контрольных битов должно передаваться вместе с 16 информационными битами для обеспечения восстановимости информации, если вероятность искажения составляет 1%.

Подставляя $p = 0{,}01$ в формулу (25), находим $L_{\min} = 1{,}081$.

Если $k_i = 16$, то из формулы (24) получаем $k = k_i \cdot L_{\min} = 17{,}29$.

С учетом того, что количество контрольных битов выражается целым числом $k_c > k - k_i = 2$, реальная избыточность L согласно выражению (24) составит 1,125.

Коды Хемминга

Рассмотрим метод кодирования, позволяющий определять, в каком бите находится ошибка. Метод был предложен в 1948 г. Р. Хеммингом. Построенные по этому методу коды получили название *коды Хемминга*.

Идея состоит в добавлении к информационным битам нескольких битов четности, каждый из которых контролирует определенные информационные биты. Если пронумеровать все передаваемые биты, начиная с единицы слева направо (информационные биты нумеруются с нуля и справа налево), то контрольными оказываются биты, номера которых равны степеням числа 2, а все остальные биты являются информационными. Например, для 8-битного информационного кода контрольными окажутся биты с номерами 1, 2, 4 и 8 (рисунок 16).

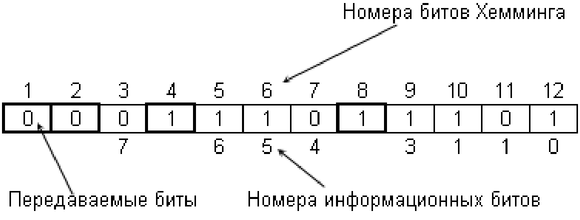


Рисунок 16 – **Коды Хемминга для 8-битного информационного кода**

Номера контролируемых битов для каждого проверочного бита приведены в таблице 13. В перечень контролируемых битов входит и тот, в котором располагается проверочный. При этом состояние проверочного бита устанавливается таким образом, чтобы суммарное количество единиц в контролируемых им битах было четным.

Таблица 13 – **Номера контролируемых битов**

Проверочные 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	

Для любого номера проверочного бита *n*, начиная с него, *n* бит подряд оказываются проверяемыми, затем – группа *n* непроверяемых бит. Далее происходит чередование групп.

Пример 22. Пусть вместо последовательности 000111011101 пришла следующая последовательность 000101011101. В 5-м бите единица заменилась нулем:

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

Анализируем состояние контрольных битов в соответствии с таблицей 13.

Бит 1 – *неверно*, т. е. ошибка находится в каком-либо бите с нечетным номером.

Бит 2 – *верно*, следовательно, биты с нечетными номерами 3, 7 и 11 верны, т. е. ошибка в 5 или 9-м.

Бит 4 – *неверно*, значит, ошибка может содержаться только в 5-м бите.

Таким образом, однозначно устанавливается, что ошибочным является 5-й бит. Нужно исправить его значение на противоположное и восстановить правильную последовательность.

Номер бита, содержащего ошибку (5), равен сумме номеров контрольных битов, указавших на ее существование (1 и 4), – это общее свойство кодов Хемминга.

Алгоритм проверки и исправления передаваемой последовательности битов в представлении Хемминга следующий:

1. Произвести проверку всех битов четности.
2. Если все биты четности верны, то перейти к пункту 5.
3. Вычислить сумму номеров всех неправильных битов четности.
4. Инвертировать содержимое бита, номер которого равен сумме, найденной в пункте 3.
5. Исключить биты четности, передать правильный информационный код.

Избыточность кодов Хемминга для различных длин передаваемых последовательностей приведена в таблице 14.

Таблица 14 – **Избыточность кодов Хемминга**

Число информационных битов	Число контрольных битов	Избыточность
8	4	1,50
16	5	1,31
37	6	1,06

Видно, что выгоднее передавать и хранить более длинные последовательности битов. При этом избыточность не должна оказаться меньше $L_{\min}$ для выбранного канала связи.

5.4. Способы передачи информации в компьютерных линиях связи

В компьютерных линиях связи используются два способа передачи:

- *параллельный*, когда передаются одновременно все биты машинного слова;
- *последовательный*, когда биты передаются поочередно, начиная с младшего.

5.4.1. Канал параллельной передачи

Для одновременной передачи нескольких сигналов требуется линия связи, количество проводников в которой совпадает с числом передаваемых сигналов. Такая линия связи называется *шиной.* Количество проводников определяет *ширину* или *разрядность* шины. Во внутренних линиях компьютера могут использоваться 16-, 32- и 64-

раз-
рядные шины. На рисунке 17 показана линия параллельной передачи, связывающая регистр арифметико-логического устройства и ячейку памяти компьютера.

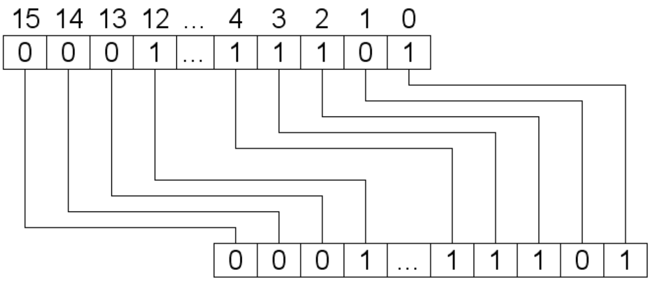


Рисунок 17 – **Параллельный способ передачи данных в компьютере**

Шина обеспечивает наиболее быстрый способ передачи информации, поскольку за два такта синхрогенератора компьютера передается целое машинное слово. В общем случае, если ν_m – частота генератора, h – разрядность шины, а n – число тактов, за которые осуществляется передача, то согласно формулам (19) и (22) пропускная способность канала параллельной передачи будет равна

$$C = \frac{\nu_m \cdot h}{n}.$$

Например, при тактовой частоте генератора 300 МГц и ширине шины 32 бита максимальная скорость передачи составит

$$C = \frac{32 \cdot 300 \cdot 10^6}{2} = 4{,}8 \cdot 10^9 \text{ бит/с} = 4{,}8 \text{ Гбита/с}.$$

Параллельный способ передачи используется во внутренних линиях связи компьютера: на материнской плате, при обмене информацией с устройствами внешней памяти (магнитными дисками, компакт-дисками), а также для связи с внешними устройствами, подключаемыми к LPT-порту (принтером, плоттером и др.).

К недостаткам параллельного способа передачи относят следующие:

- невозможна передача на большие расстояния (более нескольких метров);
- требуются многожильные специальные провода для связи, что

существенно повышает стоимость линии.

5.4.2. Последовательная передача данных

При объединении компьютеров в сети используется последовательный способ передачи. Возможны два режима последовательной передачи – *синхронный* и *асинхронный*.

При *синхронной* передаче каждый передаваемый бит сопровождается импульсом синхронизации, информирующим приемник о наличии в линии информационного бита. Синхронизирующие импульсы управляют приемом информации, поэтому между передатчиком и приемником должны быть протянуты как минимум три провода: для передачи данных, для передачи синхроимпульсов, общий заземленный.

Если же расстояние между источником и приемником составляет несколько метров, то каждый из сигналов приходится посылать по экранированному кабелю, что значительно увеличивает стоимость линии связи. Синхронный способ связи не получил широкого распространения.

Асинхронный способ передачи не требует синхронизации действий приемника и передатчика, поэтому для связи достаточно линии из двух проводников, и можно использовать телефонные линии. При этом источник и приемник информации должны быть согласованы по формату и скорости передачи. Передача производится машинными словами, дополненными несколькими служебными битами.

Рассмотрим пример передачи 8-битного слова с одним контрольным битом (рисунок 18). В отсутствии передачи в линии поддерживается уровень сигнала, соответствующий логической единице.

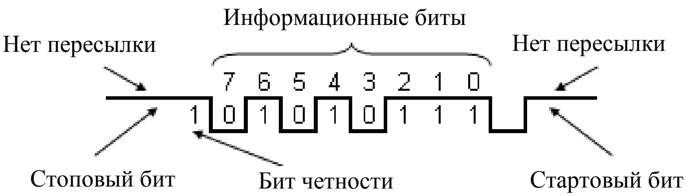


Рисунок 18 – **Передача 8-битного слова с одним контрольным битом**

Передатчик может начать пересылку в любой момент посредством генерации *стартового бита,* который переводит линию в состояние логического нуля на время продолжительности элементарного сигнала. По нему приемник узнает, что передача началась. Затем происходит передача информационных битов, начиная с младшего (0-го). За ними передается контрольный бит четности, за которым следует *стоповый бит*, который, в свою очередь, вновь переводит линию в состояние ожидания, т. е. логической единицы. Вся передаваемая цепочка сигналов от стартового до стопового бита называется *кадром*.

Дополнение кадра граничными битами приводит к увеличению избыточности кода и суммарного времени передачи. Поскольку биты передаются по очереди, скорость передачи ниже, чем в параллельном способе, но скорость может достигать нескольких Гбит/с – этого достаточно для передачи телевизионного сигнала.

5.4.3. Связь компьютеров по телефонным линиям

Экономически выгодно применение телефонных линий для осуществления связи между компьютерами. Использование двухпроводных линий и большие расстояния определяют последовательный способ передачи. Телефонные линии предназначены для передачи аналоговых электрических сигналов с частотой человеческого голоса (верхняя граница составляет 3400 Гц). При передаче по таким линиям сигналов с частотами, на которых работает компьютер (100–500 МГц), они будут испытывать значительные искажения и быстро затухать. По этой причине передача осуществляется в соответствии со схемой, представленной на рисунке 19.

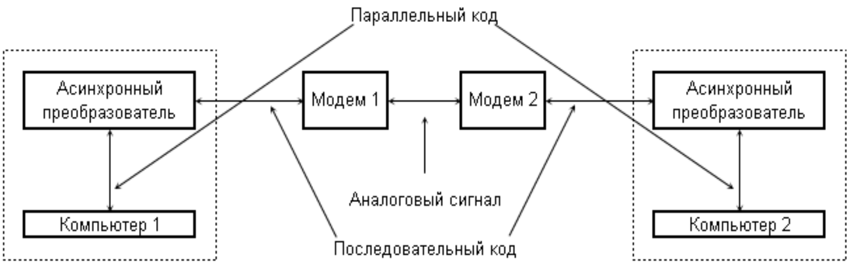


Рисунок 19 – **Компьютерная линия связи с применением модемов**

На передающем конце сначала осуществляется преобразование параллельного компьютерного кода в асинхронный последовательный. Затем посредством другого устройства – *модема* производится *модуляция* (преобразование) двухуровневых импульсных компьютерных сигналов в аналоговые сигналы, которые без большого затухания и искажения могут распространяться в телефонных линиях. На приемном конце происходит обратная цепочка: модемом аналоговый сигнал переводится в двухуровневый последовательный код (демодуляция), у которого затем удаляются все неинформационные биты, и он преобразуется в параллельный.

Асинхронный преобразователь располагается в самом компьютере в виде блока, осуществляющего обмен последовательным способом с любым устройством. Этот блок называется *последовательным портом* (COM-портом). Кроме модема к нему присоединяются также внешние манипуляторы (мышь, джойстик).

Для увеличения скорости передачи применяют более сложные методы модуляции, при которых каждый передаваемый элементарный сигнал кодирует не один, а несколько бит.

СПИСОК ЛИТЕРАТУРЫ

Галлагер, Р. Теория информации и надежная связь / Р. Галлагер. – М. : Совет. радио, 1974. – 720 с.

Дмитриев, В. И. Прикладная теория информации : учеб. / В. И. Дмитриев. – М. : Высш. шк., 1989. – 320 с.

Колесник, В. Д. Курс теории информации : учеб. пособие / В. Д. Колесник, Г. Ш. Полтырев. – М. : Наука, 1982. – 416 с.

Стратонович, Р. Л. Теория информации / Р. Л. Стратонович. – М. : Совет. радио, 1975. – 424 с.

Стариченко, Б. Е. Теоретические основы информатики : учеб. пособие / Б. Е. Стариченко. – М. : Горячая линия – Телеком, 2004. – 312 с.

Чисар, И. Теория информации: теоремы кодирования для дискретных систем без памяти : [пер. с англ.] / И. Чисар, Я. Кернер. – М. : Мир, 1985. – 400 с.

ПРИЛОЖЕНИЯ

Приложение А

Таблица кодов ASCII

Код	Символ	Код	Символ	Код	Символ	Код	Символ
0	NUL	32		64	@	96	`
1	SOH	33	!	65	A	97	a
2	STX	34	"	66	B	98	b
3	ETX	35	#	67	C	99	c
4	EOT	36	$	68	D	100	d
5	ENQ	37	%	69	E	101	e
6	ACK	38	&	70	F	102	f
7	BEL	39	'	71	G	103	g
8	BS	40	(	72	H	104	h
9	TAB	41	)	73	I	105	i
10	LF	42	*	74	J	106	j
11	VT	43	+	75	K	107	k
12	FF	44	,	76	L	108	l
13	CR	45	–	77	M	109	m
14	SO	46	.	78	N	110	n
15	SI	47	/	79	O	111	o
16	DLE	48	0	80	P	112	p
17	DC1	49	1	81	Q	113	q
18	DC2	50	2	82	R	114	r
19	DC3	51	3	83	S	115	s
20	DC4	52	4	84	T	116	t
21	NAK	53	5	85	U	117	u
22	SYN	54	6	86	V	118	v
23	ETB	55	7	87	W	119	w
24	CAN	56	8	88	X	120	x
25	EM	57	9	89	Y	121	y
26	SUB	58	:	90	Z	122	z
27	ESC	59	;	91	[	123	{
28	FS	60	<	92	\	124	|
29	GS	61	=	93	]	125	}
30	RS	62	>	94	^	126	~

31	US	63	?	95	_	127	DEL

Окончание

Код	Символ	Код	Символ	Код	Символ	Код	Символ
128	А	160	а	192	└	224	р
129	Б	161	б	193	┴	225	с
130	В	162	в	194	┬	226	т
131	Г	163	г	195	├	227	у
132	Д	164	д	196	─	228	ф
133	Е	165	е	197	┼	229	х
134	Ж	166	ж	198	╞	230	ц
135	З	167	з	199	╟	231	ч
136	И	168	и	200	╚	232	ш
137	Й	169	й	201	╔	233	щ
138	К	170	к	202	╩	234	ъ
139	Л	171	л	203	╦	235	ы
140	М	172	м	204	╠	236	ь
141	Н	173	н	205	═	237	э
142	О	174	о	206	╬	238	ю
143	П	175	п	207	╧	239	я
144	Р	176	░	208	╨	240	Е
145	С	177	▒	209	╤	241	е
146	Т	178	▓	210	╥	242	Є
147	У	179	│	211	╙	243	є
148	Ф	180	┤	212	╘	244	Ї
149	Х	181	╡	213	╒	245	ї
150	Ц	182	╢	214	╓	246	Ў
151	Ч	183	╖	215	╫	247	ў
152	Ш	184	╕	216	╪	248	°
153	Щ	185	╣	217	┘	249	·
154	Ъ	186	║	218	┌	250	·
155	Ы	187	╗	219	█	251	√
156	Ь	188	╝	220	▄	252	№
157	Э	189	╜	221	▌	253	¤
158	Ю	190	╛	222	▐	254	■
159	Я	191	┐	223	▀	255	sp

Назначение специализированных символов

Форматирование

BS (Backspace) – указывает на движение механизма печати или курсора дисплея назад на одну позицию.

HT (Horizontal Tabulation) – указывает на движение механизма печати или курсора дисплея до следующей предписанной «позиции табуляции».

LF (Line Feed) – указывает на движение механизма печати или курсора дисплея к началу следующей строки (на одну строку вниз).

VT (Vertical Tabulation) – указывает на движение механизма печати или курсора дисплея к следующей группе строк.

FF (Form Feed) – указывает на движение механизма печати или курсора дисплея к исходной позиции следующей страницы, формы или экрана.

CR (Carriage Return) – указывает на движение механизма печати или курсора дисплея к исходной (крайней левой) позиции текущей строки.

Передача данных

SOH (Start of Heading) – используется для указания начала заголовка, который может содержать информацию о маршрутизации или адрес.

STX (Start of Text) – указывает на начало текста и одновременно на конец заголовка.

ETX (End of Text) – используется при завершении текста, который был начат с символа STX.

ENQ (Enquiry) – запрос идентификационных данных от удаленной станции.

ACK (Acknowledge) – приемное устройство передает этот символ отправителю в качестве подтверждения успешного приема данных.

NAK (Negative Acknowledgement) – приемное устройство передает этот символ отправителю в случае отрицания (неудачи) приема данных.

SYN (Synchronous/Idle) – используется в синхронизированных системах передачи. В моменты отсутствия передачи данных система непрерывно посылает символы SYN для обеспечения синхронизации.

ETB (End of Transmission Block) – указывает на конец блока данных для коммуникационных целей. Используется для разбиения на отдельные блоки больших объемов данных.

Разделительные знаки при передаче информации

FS (File Separator) – разделитель файлов.
GS (Group Separator) – разделитель групп.
RS (Record Separtator) – разделитель записей.
US (Unit Separator) – разделитель элементов.

Другие символы

NUL (Null) – используется для передачи в случае отсутствия данных.

BEL (Bell) – используется для управления устройствами сигнализации.

SO (Shift Out) – указывает, что все последующие кодовые комбинации должны интерпретироваться согласно внешнему набору символов до прихода символа SI.

SI (Shift In) – указывает, что последующие кодовые комбинации должны интерпретироваться согласно стандартному набору символов.

DLE (Data Link Escape) – изменение значения идущих следом символов. Используется для дополнительного контроля или для передачи произвольной комбинации бит.

DC1, DC2, DC3, DC4 (Device Controls) – символы для управления вспомогательными устройствами (специальными функциями).

CAN (Cancel) – указывает, что данные, которые предшествовали этому символу в сообщении или блоке, должны игнорироваться (обычно в случае обнаружения ошибки).

EM (End of Medium) – указывает на физический конец ленты или другого носителя информации.

SUB (Substitute) – используется для подмены ошибочного или недопустимого символа.

ESC (Escape) – используется для расширения кода, указывая на то, что последующий символ имеет альтернативное значение.

SP (Space) – непечатаемый символ для разделения слов или перемещения механизма печати или курсора дисплея вперед на одну позицию.

DEL (Delete) – используется для удаления (стирания) предыдущего знака в сообщении.

Происхождение систем счисления

Десятичная система счисления

Появилась десятичная система в Индии. Выбор графических изображений для цифр не принципиален. Современные изображения цифр – простая стилизация древних арабских цифр. Мароканский историк Абкелькари Боужибар считает, что арабским цифрам в их первоначальном варианте было придано значение в строгом соответствии с числом углов, которые образуют фигуры. Если посмотреть на рисунок Б.1, это предположение кажется не лишенным глубокого смысла.

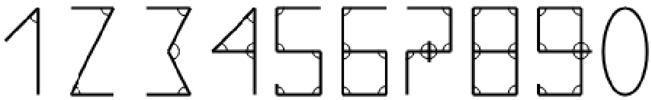


Рисунок Б.1 – **Начертание арабских цифр**

Так, единица создает лишь один угол, тройка – три, пятерка – пять и т. п., нуль не образует никакого угла, поэтому он не имеет никакого содержания.

Другие системы счисления

Десятичная система счисления далеко не сразу заняла то господствующее положение, которое она имеет сейчас. В разные исторические периоды многие народы пользовались системами счисления, отличными от десятичной.

Так, например, довольно широкое распространение имела двенадцатеричная система. Ее происхождение связано, несомненно, тоже со счетом на пальцах, а именно: так как четыре пальца руки (кроме большого) имеют в совокупности 12 фаланг (рисунок Б.2), то по этим фалангам, перебирая их по очереди большим пальцем, и ведут счет от 1 до 12. Затем 12 принимается за единицу следующего разряда и т. д.

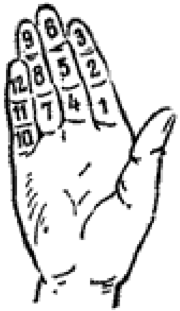


Рисунок Б.2 – **Происхождение двенадцатеричной системы**

В устной речи остатки двенадцатеричной системы сохранились и до наших дней: вместо того чтобы сказать «двенадцать», мы часто говорим «дюжина». Многие предметы: ножи, вилки, тарелки, носовые платки и т. п. – очень часто считают именно дюжинами, а не десятками. Например, сервиз бывает, как правило, на 12 или на 6 чел. и значительно реже – на 10 или 5. Сейчас уже крайне редко встречается слово «гросс», означающее дюжину дюжин (144), но еще несколько десятков лет тому назад оно было довольно широко распространено, особенно в торговом мире. Дюжиной гроссов называлась масса (1728), однако сейчас такое значение слова «масса» мало кому известно. Возможно, именно в нем лежит корень таких употребительных выражений, как «масса дел», «масса людей» и т. п. (можно сравнить с выражением «тысяча дел» и др.).

Остатки двенадцатеричной системы счисления имеются у англичан: в системе мер, например, 1 фут = 12 дюймам, и в денежной системе 1 шиллинг = 12 пенсам.

С математической точки зрения двенадцатеричная система имела бы некоторые преимущества перед десятичной, поскольку число 12 делится на 2, 3, 4 и 6, а число 10 – только на 2 и 5. Больший запас делителей у числа, служащего основанием системы счисления, создает удобства в ее использовании.

В древнем Вавилоне, культура которого, в том числе и математическая, была довольно высока, существовала весьма сложная шестидесятиричная система. Мнения историков по поводу того, как именно возникла такая система, расходятся. Одна из гипотез, впрочем не особенно достоверная, состоит в том, что произошло смешение двух племен, одно из которых пользовалось шестеричной системой, а другое – десятичной. Шестидесятеричная система возникла как компромисс между этими двумя системами. Другая гипотеза состоит в том, что вавилоняне считали продолжительность года равной 360 сут, что, естественно, связывалось с числом 60. Однако это предположение тоже нельзя считать достаточно обоснованным: астрономические познания древних вавилонян были довольно значительны, поэтому следует думать, что погрешность, с которой они определяли продолжительность года, была значительно меньше чем 5 сут. Несмотря на то что происхождение шестидесятеричной системы остается неясным, самый факт ее существования и широкого распространения в Вавилонском государстве достаточно хорошо установлен. Эта система, как и двенадцатеричная, в какой-то степени сохранилась и до наших дней (например, в делении часа на 60 мин, а минуты – на 60 с и в аналогичной системе измерения углов – 1 градус = 60 мин, 1 мин = 60 с). В целом, однако, эта система, требующая шестидесяти различных «цифр», довольно громоздка и менее удобна, чем десятичная.

По свидетельству известного исследователя Африки Стенли, у ряда африканских племен была распространена пятеричная система счисления. Связь этой системы со строением человеческой руки – первоначальной «счетной машины» – достаточно очевидна.

У ацтеков и майя – народов, населявших в течение многих столетий обширные области американского континента и создавших там высокую культуру, почти полностью уничтоженную испанскими завоевателями в XVI–XVII вв., – была принята двадцатеричная система. Та же двадцатеричная система была принята и у кельтов, населявших Западную Европу начиная со второго тысячелетия до нашей эры. Некоторые следы двадцатеричной системы кельтов сохранились в современном французском языке. Например, «восемьдесят» по-французски будет quatre-vingts, что буквально – «четырежды двадцать». Число 20 встречается и во французской денежной системе: основная денежная единица – франк делится на 20 су.

Из четырех перечисленных выше систем счисления (двенадцатеричной, пятеричной, шестидесятеричной и двадцатеричной), сыгравших наряду с десятичной заметную роль в развитии человеческой культуры, все, кроме шестидесятеричной, источники которой неясны, связаны с тем или иным способом счета по пальцам рук (или и рук, и ног), т. е. имеют, подобно десятичной системе, несомненное «анатомическое» происхождение.

Как показывают приведенные выше примеры, многочисленные следы этих систем счисления сохранились до наших дней и в языках многих народов, и в принятых денежных системах, и в системах мер. Однако для записи чисел и для выполнения вычислений всегда используется десятичная система счисления.

СОДЕРЖАНИЕ

Учебное издание

Бондарева Валентина Викторовна

ТЕОРИЯ ИНФОРМАЦИИ

Курс лекций
для студентов специальности 1-26 03 01
«Управление информационными ресурсами»

Редактор М. П. Герасенко
Технический редактор Т. В. Гавриленко
Компьютерная верстка Н. Н. Короедова

Подписано в печать 19.10.12. Бумага типографская № 1.
Формат 60 × 84 $^{1}/_{16}$. Гарнитура Таймс. Ризография.
Усл. печ. л. 4,88. Уч.-изд. л. 5,00. Тираж 170 экз.
Заказ №

Учреждение образования
«Белорусский торгово-экономический университет
потребительской кооперации».
246029, г. Гомель, просп. Октября, 50.
ЛИ № 02330/0494302 от 04.03.2009 г.

Отпечатано в учреждении образования
«Белорусский торгово-экономический университет
потребительской кооперации».
246029, г. Гомель, просп. Октября, 50.

БЕЛКООПСОЮЗ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БЕЛОРУССКИЙ ТОРГОВО-ЭКОНОМИЧЕСКИЙ
УНИВЕРСИТЕТ ПОТРЕБИТЕЛЬСКОЙ КООПЕРАЦИИ»

В. В. БОНДАРЕВА

ТЕОРИЯ ИНФОРМАЦИИ

Курс лекций
для студентов специальности 1-26 03 01
«Управление информационными ресурсами»

Гомель 2012