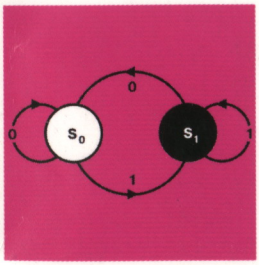
программирования

М. ВЕРНЕР

Основы кодирования



ТЕХНОСФЕРА

МИР программирования

М. ВЕРНЕР

Основы кодирования.

Учебник для ВУЗов.

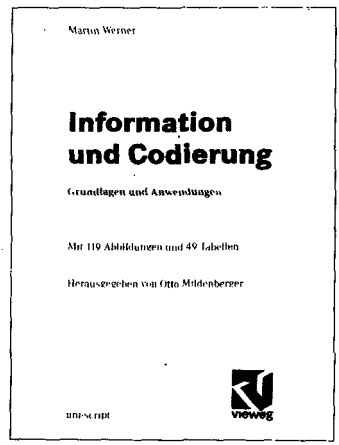
Перевод с немецкого
Д. К. Зигангирова

*Рекомендовано ИППИ РАН
в качестве учебника
для студентов, обучающихся
по направлению
"Прикладные математика и физика"*

ТЕХНОСФЕРА
Москва

М. Вернер
Основы кодирования. ***Учебник для ВУЗов.***
Москва:
Техносфера, 2004. – 288с. ISBN 5-94836-019-9

Первое на русском языке массовое пособие для будущих инженеров-связистов и проектировщиков радиоэлектронной аппаратуры, включая системы на кристалле. Даны основы теории информации и сжатия данных, доходчиво изложены современные алгоритмы помехоустойчивого кодирования, реализации циклических и сверточных кодов.



ISBN 5-94836-019-9
ISBN 3-528-03951-5 (нем.)

Предисловие

Информация и кодирование -- два основных понятия современной информационной техники. Информация в техническом смысле этого слова и методы защиты информации от ошибок, возникающих в результате передачи сообщений, являются сегодня основой при подготовке специалистов, работающих в области информационных технологий. В данной книге предпринята попытка изложить эти основы в компактной форме. «Информация и кодирования» базируется на курсе лекций, прочитанных в четвертом семестре на факультете «Электротехника и информационная техника» университета г. Фулда. В первой части вводятся понятия информации, энтропии и избыточности. Подход, при котором информация является мерой неопределенности, ведет от случайных экспериментов к понятию энтропии. Таким образом, мысленно подвергая информационные источники случайным испытаниям, мы вводим понятие энтропии, как измеряемой величины. При этом формулируются ряд важнейших вопросов, касающихся оптимизации информационных потоков в технических системах и, оставляя пока в стороне конкретные методы оптимизации, на эти вопросы даются ответы. При этом центральное место отводится дискретным марковским цепям, с помощью которых источники и каналы без памяти могут быть описаны.

Во второй части представлены методы, с помощью которых информация, путем добавления проверочных разрядов, может быть защищена от ошибок, возникающих при передаче по каналам связи. Представлены два семейства кодов, нашедших широкое применение - циклические коды и сверточные коды. Первые – часто используются при передаче данных в локальных сетях и в интернете. Они особенно эффективны для обнаружения пакетов ошибок в системах передачи данных с переспросом. Сверточные коды популярны в сильно зашумленных каналах, например, в мобильной связи. С их помощью исправляются ошибки, которые возникают при приеме.

Книга составлена таким образом, что обе части «Информация» и «Кодирование» могут быть прочитаны независимо друг от друга. Понятия теории информации и кодирования базируются на методах теории вероятностей и алгебры конечных полей. С этими двумя областями математики большинство студентов мало знакомо. Многие

годы преподавания показывают, что трудности лежат в непривычном материале и повышенных математических требованиях. В связи с этим, особую ценность для понимания представляет подробный разбор примеров и решений заданий с привлечением всего учебного материла. Я желаю учащимся успешного овладения книгой «Информация и кодирование».

Мартин Вернер

Часть I

Информация и кодирование

ГЛАВА I

ВВЕДЕНИЕ

Теория информации описывается с помощью вероятностных диаграмм кодирования и передачи информации в конкретных, естественно - научных приложениях. При этом появляется возможность для анализа и оптимизации потоков информации в технических системах. Отделить техническое представление информации от бытового могут помочь лингвистические понятия: синтаксис, семантика и прагматика. В этих понятиях синтаксис и семантика являются аналогом технических данных. Синтаксис определяет все допустимые символы и последовательности символов. Семантика объясняет их значения. Истинный смысл и область применения поясняются прагматикой. Прагматика связывает воедино такие понятия, как данные, техническая информация, информация в бытовом понимании этого слова. Окончательно, термин «информация», понимаемый обычно как известие или руководство к действию, превращается в сообщение, которое должно быть передано.

- Синтаксис + семантика → Данные.
- Данные + прагматика → Сообщение.

Основой дальнейших рассуждений является понимание информации, как некоторой, экспериментально устанавливаемой величины. Толчком к этому послужила работа Клода Шеннона «Математическая теория связи» [1], опубликованная в 1948 г. В ней К. Шеннон дал определение понятий современной теории информации и набросал основы сегодняшней техники связи. Шеннон, в качестве примера, привел широко распространенные в то время перфокарты. Одна перфокарта с N возможными позициями для отверстий может содержать в точности 2^N сообщений. Если имеются две перфокарты, то число возможностей равно уже 2^{2N}. Таким образом, число возможных сообщений, которые несут две перфокарты, равно квадрату числа сообщений, содержащихся на одной перфокарте. С другой стороны, можно было бы ожидать, что две перфокарты могут хранить вдвое больше информации, чем одна. Здесь для описания меры информации напрашивается логарифмическая функция, которая дает ожидаемое удвоение:

$$\log 2^N = N \log 2 \log 2^{2N} = 2N \log 2.$$

Общая модель связи по К. Шеннону приведена на рис. 1.1.

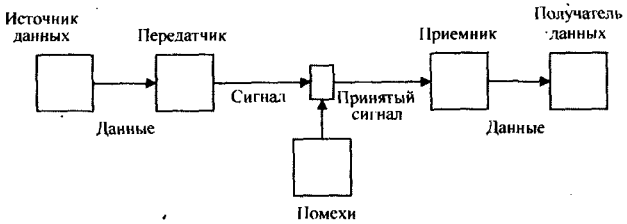


Рис. 1.1. Модель передачи информации по каналу связи по К. Шеннону.

Исходным пунктом является источник информации. Его сообщения поступают на передатчик. При этом, сообщения могут представлять собой отдельные буквы связанного текста, значения функций времени, функции изменения напряжения на выходе микрофона, телевизионные сигналы и т.д. Передатчик вырабатывает сигналы, согласованные с физическими свойствами канала. Канал изображен на рисунке как источник помех, которые оказывает на передаваемый сигнал некоторое влияние. Зашумленный сигнал поступает в приемник, на который возложена самая сложная задача. Он должен из зашумленного сигнала выделить переданное сообщение и отправить его потребителю.

Вторая большая тематика этой книги – кодирование. Под кодированием понимают некоторое отображение сообщения по известным правилам. При этом, в шенноновской модели передачи информации, блоки передатчика и приемника нужно расширить в соответствии с этими правилами. Между кодированием источников и кодированием канала существует четкое различие. Примерами кодирования источников могут служить передача связанного текста кодом Морзе, оцифровка аудио сигнала при записи на компакт диски. При кодировании источников избыточность сообщений снижается и такое кодирование часто называют сжатием данных. Кодирование каналов, наоборот, увеличивает избыточность сообщений. Внесение дополнительных проверочных символов позволяет обнаруживать и даже исправлять ошибки, возникающие при передаче информации по каналу. Кодирование канала в дальнейшем мы будем называть помехоустойчивым кодированием. Без помехоустойчивого кодирования было бы невозможным создание накопителей огромной емкости, таких, как CD-ROM, DVD или жестких дисков. Дополнительные затраты

на помехоустойчивое кодирование, которое обеспечивает приемлемые вероятности ошибок записи/чтения, становятся пренебрежимо малыми по сравнению с выигрышем от достигаемой при этом плотности записи. Рассмотренные примеры показывают, что информация и кодирование являются центральными понятиями, без которых современная информационная техника просто не существовала бы. Следующие главы углубят эти понятия и их приложения.

ГЛАВА 2

ИНФОРМАЦИЯ, ЭНТРОПИЯ И ИЗБЫТОЧНОСТЬ

2.1. Информация одного события

Обмен информацией, несмотря на свою нематериальную природу, является неотъемлемой частью нашей жизни. Норберт Виннер, один из основателей современной теории информации, охарактеризовал информацию следующим образом [9]:

«Информация есть информация, а не материя или энергия».

Согласно Н. Виннеру, информация является новым элементом в дополнении к материи и энергии. Информация для людей настолько же важна, насколько трудно представить себе это понятие в естественнонаучной форме. Мы говорим, например: «Эта информация для меня важна», имея в виду некоторую конкретную ситуацию. Такое субъективное восприятие не подходит для технического представления информации. Как строго научное понятие, информация должна быть введена в технику в качестве измеряемой величины (аналогично длине в метрах, напряжению в вольтах и т.д.).

Наш повседневный опыт говорит о том, что, принимая информацию к сведению, мы постоянно устраняем некоторую неопределенность. Это очень напоминает *эксперименты со случайными событиями*. Проводя такие эксперименты, мы наблюдаем случайные события и это снижает *неопределенность* системы.

Переходим к самой сущности теории информации. Прежде всего дадим определение простейшего источника, а затем введем понятие количества информации, как измеряемой величины для того, чтобы с ее помощью охарактеризовать источник.

Дискретный источник без памяти

Простейший *дискретный источник без памяти* X в каждый фиксированный момент времени выдает некоторый символ x_i из конечного *алфавита* $X = \{x_1, x_2, \ldots, x_N\}$ с вероятностью $P(x_i) = p_i$. Выборки символов производятся независимо друг от друга.

В качестве простейшего примера можно привести двоичный источник без памяти с алфавитом $X = \{x_1 = 0, x_2 = 1\}$ и вероятностями $0 \leq p_1 \leq 1$ и $p_2 = 1 - p_1$. Выбор очередной цифры производится независимо от прежних и последующих выборок. Изображение простейшего источника приведено на рис. 2.1. Рассмотрим вначале одиночные события. Повседневный опыт подсказывает, что часто происходящие события, так же как и их вероятности, дают нам мало информации. Возьмем, например, сообщение «собака укусила человека». Это привычное известие не привлекло бы к себе никакого внимания, в то время, как сообщение «человек укусил собаку» все газеты напечатали бы крупным шрифтом. Из этого можно сделать вывод: частые, ожидаемые события несут мало информации и, наоборот, редкие, т.е. неожиданные события, обладают высоким информационным содержанием. Следовательно, информация и вероятность находятся в обратно пропорциональной зависимости. Исходя из этого, введем понятие количества информации, как измеряемой величины на основании следующих трех аксиом [1].

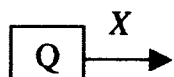


Рис. 2.1. Простейший источник источник информации алфавита X.

Аксиомы для определения количества информации [1]

1. Информация одиночного события $x_i \in X$, происходящего с вероятностью p_i, имеет положительное значение

$$I(p_i) \geq 0. \tag{2.1}$$

2. Совместная информация двух независимых событий (x_i, x_j) с совместной вероятностью $P(x_i, x_j) = p_{ij} = p_i \cdot p_j$, равна сумме их информаций

$$I(p_{i,j}) = I(p_i) + I(p_j). \tag{2.2}$$

3. Информация является непрерывной функцией от вероятности события.

Аксиомы 1 и 2 утверждают, что информация нескольких событий не может взаимно уничтожаться. Аксиома 2 вводит понятие совместной информации событий. Из аксиомы 3 следует, что небольшое изменение вероятности события приводит к небольшому изменению ее

информации. Аксиома 2 определяет информацию двух независимых событий. Из (2.2) следует, что информация события определяется как логарифмическая функция ее вероятности. Следовательно, информацию можно определить следующим образом:

Информация события, происходящего с вероятностью p, равна

$$I(p) = -\log_2(p) \text{ с } [I] = \text{ бит}. \tag{2.3}$$

В данной формуле используется двоичный логарифм. Возможны следующие обозначения двоичного логарифма: $\log_2(x) = \mathrm{ld}\ (x) = \mathrm{lb}(x)$, где под ld подразумевается термин дуальный логарифм, а под lb – бинарный.[1] Иногда используют натуральный логарифм с единицей измерения *нат*, но можно использовать любую единицу измерения информации. Можно также переходить от одной единице к другой, применяя формулу пересчета:

$$\log_a(x) = \log_b(x)/\log_b(a) = \log_b(x) \cdot \log_a(b).$$

Размерность *бит* используется в информационной технике при двоичной системы исчисления. Как будет показано в следующих разделах, двоичная система очень удобна при описании процесса принятия решения, когда на любой вопрос существует только два ответа: «да» или «нет». В [10] приведена наглядная интерпретация понятия «бит».

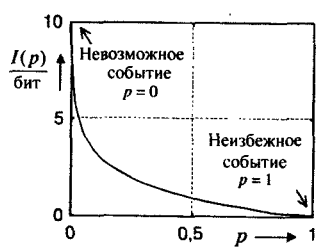


Рис. 2.2. Информация символа $I(p)$ с вероятностью появления p.

На рис. 2.2 показано поведение информации как функции вероятности. Информация постоянно происходящего события равна нулю.

[1] В отечественной математической литературе для обозначения двоичного и натурального логарифма принято использовать $\log_2$ и ln. – *Прим. перев.*

С ростом неопределенности информация также растет и для *невозможного события* стремится к бесконечности. Таким образом, информация соответствует всем приведенным ранее рассуждениям и удовлетворяет аксиомам 1 – 3. С точки зрения теории вероятности, определение информации можно рассматривать как некоторое отображение событий. Аналогичное отображение имеют стохастические переменные. В следующих разделах это будет поясняться на примерах.

2.2. Энтропия и избыточность

После того, как информация отдельного события определена, рассмотрим *источник событий*. Для его описания будем использовать информацию, которую несут содержащиеся в нем события. По аналогии с термодинамикой введем понятие энтропии. В термодинамике энтропия является мерой неупорядоченности системы. В теории информации энтропия определена как мера неопределенности источника. Используя информацию отдельных событий, выразим энтропию источника следующим образом:

Энтропия простейшего источника без памяти X с алфавитом $X = \{x_1, x_2, \ldots, x_N\}$ и соответствующим вероятностям $p_1, p_2, \ldots, p_N$ равна

$$H(x) = \sum_{i}^{N} -p_i \log_2(p_i) \text{ бит.} \tag{2.4}$$

Представим себе игру, в которой некоторое событие источника должно быть предсказано. Если источник отдает предпочтение определенному событию, смело ставьте на него и, в основном, вы будете выигрывать. Если все события равновероятны, то ставьте на любое событие: если неопределенность источника максимальна, шансы на выигрыш минимальны.

Пример: Оценка энтропии.

Поясним эту связь на примере простейшего дискретного источника без памяти из табл. 2.1. Информация источника представляет собой результат эксперимента со случайными событиями a, b, c, d. Пусть в результате повторения этого эксперимента мы получаем последовательность

$$\{a, b, a, d, a, a, c, d, b, a, a, b, \ldots\}. \tag{2.5}$$

Таблица 2.1. Дискретный источник без памяти с символами алфавита $X = \{a, b, c, d\}$ с вероятностью p_i и информацией $I(p_i)$.

Символ	a	b	c	d
p_i	1/2	1/4	1/8	1/8
$I(p_i)$	1 бит	2 бит	3 бит	3 бит

Подставив на место каждого события его информацию, получим типичную функцию стохастического процесса

$$\{I[n]/\text{бит}\} = \{1, 2, 1, 3, 1, 1, 3, 3, 2, 1, 1, 2, \ldots\}. \tag{2.6}$$

Предположим *эргодичность* (постоянство поведения) такого процесса во времени. Такую эргодичность мы, например, предполагаем при бросании монетки или игрального кубика. С ростом числа испытаний N среднее значение информации источника

$$\overline{I} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} I[n] \tag{2.7}$$

стремится к *математическому ожиданию*

$$E(I) = \sum_{i=1}^{4} -p_i \log_2(p_i) \text{ бит}. \tag{2.8}$$

Таким образом, учитывая сходимость ряда (2.7) к математическому ожиданию, получаем практическую меру для информации источника. В рассмотренном примере математическое ожидание $E(I)$ равно 1,75 бит. Из первых 12 испытаний мы также получаем оценку для $I(n)$ 1,75 бит.

Проводя аналогичные рассуждения, Шеннон [1] заложил в определение энтропии три следующие аксиомы.

Аксиоматическое определение энтропии

1. Энтропия $H(X) = f(p_1, p_2, \ldots, p_N)$ является непрерывной функцией вероятностей $p_1, p_2, \ldots, p_N$.

2. Для источников с одинаковой вероятностью событий $p_i = \frac{1}{N}$ энтропия увеличивается с ростом числа событий N.

3. Разложение процедуры выбора событий на несколько этапов не изменяет энтропию (процедуру выбора можно свести к последовательным двоичным решениям).

Пример: Разложение процедуры выбора.

Данный пример поясняет аксиому 3. Рассмотрим три события a, b и c, которые происходят с соответственными вероятностями $1/2, 1/3$ и $1/6$. Для того, чтобы выбрать одно из этих трех, мы можем поставить два независимых вопроса (рис. 2.3).

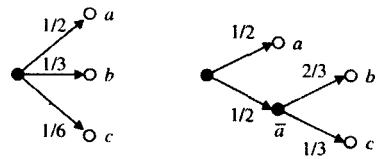


Рис. 2.3. Разложение процесса выбора символов.

На эти вопросы могут быть даны только два ответа: или «да» или «нет». Согласно аксиоме (3), к энтропии предъявляется следующее требование:

$$H\left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right) = H_1\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{1}{2}H_2\left(\frac{2}{3}, \frac{1}{2}\right), \tag{2.9}$$

причем, второй вопрос задается с вероятностью $1/2$. Мы покажем в общем виде (рис. 2.4), что определение энтропии (2.8) удовлетворяет требованию аксиомы (H3).

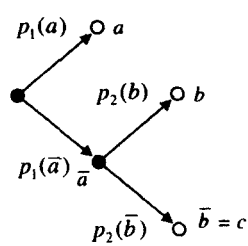


Рис. 2.4. Разложение процедуры принятия решения.

Для *разложенной* энтропии получаем

$$\begin{aligned}\frac{H(X)}{\text{бит}} = &-p_1(a)\log_2(p_1(a)) - p_1(\bar{a})\log_2(p_1(\bar{a})) + \\ &+ p_1(\bar{a})[-p_2(b)\log_2(p_2(b)) - p_2(\bar{b})\log_2(p_2(\bar{b}))],\end{aligned} \tag{2.10}$$

где вероятности определяются следующим образом

$$p_1(\bar{a}) = p(b) + p(c), \tag{2.11}$$

$$p_2(b) + p_2(\overline{b}) = 1, \tag{2.12}$$

$$\frac{p_2(b)}{p_2(\overline{b})} = \frac{p(b)}{p(c)}. \tag{2.13}$$

Замечание. *Последнее равенство подтверждается постановкой эксперимента со случайными событиями. Пусть событие a произошло 300 раз, событие b – 200, а событие c – 100 раз. Частота каждого события в данном примере равна его вероятности. Если мы отбросим событие a, то останется 300 выборок событий b и c, частоты выборок этих событий удвоятся, но их отношение не изменится.*

Из (2.11)- (2.13) следует, что вероятности на втором шаге можно выразить как

$$p_2(b) = \frac{p(b)}{p(b) + p(c)} = \frac{p(b)}{p(\overline{a})} \tag{2.14}$$

$$p_2(\overline{b}) = \frac{p(c)}{p(b) + p(c)} = \frac{p(c)}{p(\overline{a})}. \tag{2.15}$$

Подставляя полученные выражения в формулу для энтропии, имеем

$$\begin{aligned}\frac{H(X)}{\text{бит}} = &-p_1(a)\log_2(p_1(a)) - p_1(\overline{a})\log_2(p_1(\overline{a})) + \\ &+ p_1(\overline{a})\left[-\frac{p(b)}{p_1(\overline{a})}\log_2\left(\frac{p(b)}{p_1(\overline{a})}\right) - \frac{p(c)}{p_1(\overline{a})}\log_2\left(\frac{p(c)}{p_1(\overline{a})}\right)\right],\end{aligned} \tag{2.16}$$

что после упрощений соответствует энтропии без разложения процесса выбора событий

$$\frac{H(X)}{\text{бит}} = -p_1(a)\log_2(p_1(a)) - p(b)\log_2(p(b)) - p(c)\log_2(p(c)). \tag{2.17}$$

В рассмотренном примере было использовано свойство *логарифмической функции*, переводящее произведение в сумму. Определение (2.4) является единственным, при котором аксиома 3 имеет силу. Заметим также, что рассмотренное разложение процесса выбора событий может быть сведено к последовательности бинарных решений «да» и «нет». Максимальной неопределенности соответствует максимальная энтропия источника. Сформулируем следующую теорему:

Теорема 2.2.1. Энтропия простейшего дискретного источника без памяти максимальна, если все события, в нем содержащиеся, имеют одинаковую вероятность. В этом случае энтропия просто равна логарифму числа событий

$$H_0 = \log_2 N \text{ бит}. \tag{2.18}$$

Замечание. *Последующее доказательство теоремы является типичным в теории информации. В таких доказательствах используются оценки и предельные переходы, которые ранее были известны, но не находили практического применения.*

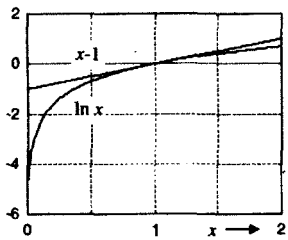


Рис. 2.5. Верхняя оценка логарифмической функции.

Доказательство. Для доказательства рассмотрим два дискретных источника без памяти P и Q, каждый из которых содержит N событий с вероятностями p_i и q_i соответственно. Далее воспользуемся известной верхней оценкой логарифмической функции (рис. 2.5)

$$\ln x \le x - 1. \tag{2.19}$$

Используя эту оценку, получаем

$$\ln q_i \underbrace{- \ln p_i}_{I(p_i) \text{ нат}} = \ln \frac{q_i}{p_i} \le \frac{q_i}{p_i} - 1. \tag{2.20}$$

Умножив обе части неравенства на p_i и просуммировав по всем событиям $1 \le i \le N$, имеем

$$\sum_{i=1}^{N} p_i [\ln q_i - \ln p_i] \le \sum_{i=1}^{N} p_i \left[\frac{q_i}{p_i} - 1 \right]. \tag{2.21}$$

После упрощения получаем

$$\frac{H(P)}{\text{нат}} + \sum_{i=1}^{N} p_i \ln q_i \le \underbrace{\sum_{i=1}^{N} q_i}_{1} - \underbrace{\sum_{i=1}^{N} p_i}_{1} = 0 \tag{2.22}$$

и, следовательно

$$\frac{H(P)}{\text{нат}} \le -\sum_{i=1}^{N} p_i \ln q_i. \tag{2.23}$$

Предположим, что источник Q содержит только равновероятные события. Тогда

$$\frac{H(P)}{\text{нат}} \le -\sum_{i=1}^{N} p_i \ln \left[\frac{1}{N}\right] = \ln N \underbrace{\sum_{i=1}^{N} p_i}_{1} = \ln N. \tag{2.24}$$

Так как в процессе доказательства на источник P не было наложено никаких ограничений, то данное неравенство имеет место для любого дискретного источника без памяти, содержащего N событий

$$H(X) \le \log_2 N \text{ бит}. \tag{2.25}$$

Максимум достигается, когда все события имеют одинаковые вероятности. ■

Любой источник, содержащий N событий, не все из которых имеют одинаковую вероятность, обладает энтропией, меньшей $\log_2 N$. Рассмотрим источник емкостью $H_0 = \log_2 N$ как резервуар для хранения информации, который никогда не переполняется.

Разность между максимальной емкостью H_0 и энтропией источника X, содержащего N событий, называется *избыточностью* источника

$$R = H_0 - H(X). \tag{2.26}$$

Относительная избыточность определяется как

$$r = \frac{R}{H_0} = 1 - \frac{H(X)}{H_0}. \tag{2.27}$$

Пример: Энтропия дискретного источника без памяти, содержащего 6 событий.

Таблица 2.2. Дискретный источник без памяти с символами x_i алфавита $X = \{a, b, c, d, e, f\}$ с вероятностью p_i и информацией $I(p_i)$.

x_i	a	b	c	d	e	f
p_i	0,05	0,15	0,05	0,4	0,2	0,15
$I(p_i)$	4,32 бит	2,74 бит	4,32 бит	1,32 бит	2,32 бит	2,74 бит

Для того, чтобы конкретизировать проведенные выше рассуждения, рассмотрим численный пример. В таблице 2.2 задан дискретный источник без памяти X с соответствующими алфавитом и вероятностями событий. Следуя (2.3), подсчитаем информацию каждого события и дополним таблицу значениями.

Энтропия источника равна

$$H(X) = 2,25 \text{ бит}. \tag{2.28}$$

Значение H_0 для событий равно

$$H_0 = \log_2 6 \text{ бит} = 2,585 \text{ бит}, \tag{2.29}$$

тогда избыточность

$$R = (\log_2 6 - 2,25) \text{ бит} = 0,335 \text{ бит} \tag{2.30}$$

и, соответственно, относительная избыточность составляет

$$r = 1 - \frac{2,25}{\log_2 6} = 0,130 \cong 13\%. \tag{2.31}$$

Особое значение имеют дискретные двоичные источники без памяти, так как в большинстве случаев с их помощью можно описать процесс передачи данных.

Пусть задан двоичный источник без памяти с алфавитом $X = \{0, 1\}$ и с вероятностями для символов «0» и «1» - $p_0 = p$ и $p_1 = 1 - p_0$ соответственно. Выбор символов производится независимо. Его энтропия, называемая также функцией Шеннона, зависит только от вероятности p.

Энтропия двоичного источника (функция Шеннона)

$$\frac{H_b(p)}{\text{бит}} = -p \log_2 p - (1-p) \log_2 (1-p). \tag{2.32}$$

На рис. 2.6 показано поведение функции Шеннона. Энтропия двоичного источника всюду положительна и симметрична относительно $p = 1/2$ и имеет максимум при одинаковой вероятности символов «0» и «1». Максимальная энтропия равная 1 бит соответствует двоичному решению, т.е. на вопрос о значении символа это ответ: либо «да» либо «нет».

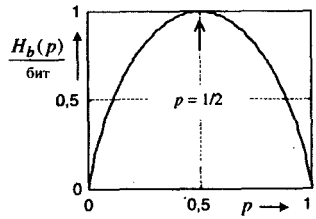


Рис. 2.6. Энтропия двоичного источника.

ГЛАВА 3

КОДИРОВАНИЕ ДЛЯ ДИСКРЕТНЫХ ИСТОЧНИКОВ БЕЗ ПАМЯТИ

3.1. Теорема кодирования источников I

В разделе 2.2 было введено понятие энтропии как средней информации дискретного источника без памяти. Более того, было показано, что любое событие, содержащееся в источнике, может быть разложено на последовательности двоичных решений с исходами «да» или «нет» без потери информации. Таким образом, каждому событию, содержащемуся в алфавите, может быть приписана некоторая последовательность двоичных символов «0» или «1» (в дальнейшем такую последовательность будем называть кодовым словом события). При этом не происходит потери информации, так как каждое событие может быть непосредственно восстановлено по соответствующему кодовому слову. Такой процесс называется кодированием источника, а совокупность кодовых слов всех событий – кодом источника.

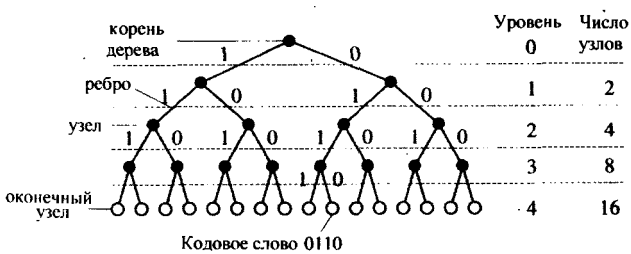


Рис. 3.1. Кодовое дерево.

Возникает вопрос: какое среднее число бит надо затратить при кодировании источника? Попутно возникают вопросы: сколько мест на плате займут переключательные элементы при реализации кода источника и каковы длительности кодовых слов при передачи информации?

Ответы на эти вопросы дает теорема Шеннона о кодировании источников. Перед тем, как приступить к рассмотрению этой теоремы, возьмем в качестве примера двоичного кодирования кодовую конструкцию, изображенную на рис. 3.1.

Начиная от корня кодового дерева число ребер на каждом уровне удваивается, причем, слева располагаются ребра, соответствующие единицам, а справа – нулям кодовых слов. Кодовые слова образуются при прохождении соответствующих ребер, ведущих от корня к оконечным узлам. В качестве примера будем рассматривать кодовое слово 0110.

Если мы пройдем N уровней, то получим 2^N оконечных узлов, которые соответствуют кодовым словам с длиной $\log_2 2^N$ бит $= N$ бит и можно закодировать 2^N событий. Предположим далее, что этим оконечным узлам соответствуют 2^N равновероятных событий из всего алфавита источника. Тогда, на кодирование каждого события затрачивается ровно N бит, что равно энтропии источника. Связь между средней длиной кодового слова и энтропией источника обобщает теорема кодирования источников.

Теорема 3.1.1. Теорема кодирования источников I.
Для любого дискретного источника без памяти X с конечным алфавитом и энтропией $H(x)$ существует D- ичный префиксный код, в котором средняя длина кодового слова $\overline{n}$ удовлетворяет неравенству

$$\frac{H(x)}{\log D} \leq \overline{n} \leq \frac{H(x)}{\log D} + 1. \tag{3.1}$$

Термин *префиксный код* означает, что никакое начало кодового слова не может быть другим кодовым словом. Это значит, что поток событий может быть закодирован без специального разделения этих событий. В случае $D = 2$ используется двоичный код. Если энтропия задана в битах, то средняя длина $\overline{n}$ также выражается в битах, что прямо указывает на использование двоичного кода.

Теорема кодирования источников указывает на то, что средняя длина кодового слова не может быть меньше энтропии. Однако, как будет показано в разделах 5.2 и 5.5, при блоковом кодировании источников, средняя длина кодового слова может приближаться к энтропии как угодно близко. Это обстоятельство еще раз подчеркивает важность понятия энтропии. Энтропия - есть мера минимальных средних затрат. Примером практической реализации неравенства (3.1) является код Хаффмана, рассматриваемый в следующем разделе.

Замечание. *Доказательство теоремы 3.1.1 сводится к использованию неравенства Крафта и не несет в себе каких-либо конструктивных соображений для построения оптимальных, с точки зрения затрат, кодов. Доказательство довольно объемно, но дает хорошее представление о методах теории информации. Тем не менее, оно может быть пропущено без ущерба для понимания последующих разделов.*

Доказательство.

Шаг 1. *Неравенство Крафта.*

Для существования однозначно декодируемого D-ичного кода, содержащего K кодовых слов с длинами $n_1, n_2, \ldots, n_k$, необходимо и достаточно, чтобы выполнялось неравенство Крафта

$$\sum_{k=1}^{K} D^{-n_k} \leq 1. \tag{3.2}$$

Для доказательства этого утверждения воспользуемся кодовой конструкцией, изображенной на рис. 3.2.

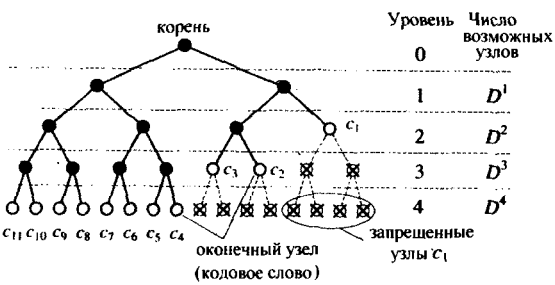


Рис. 3.2. Кодовое дерево для $D = 2$ и $n = 4$.

Мы видим 3 характерных свойства этой конструкции:

1. Существует D^i *узлов* i-го порядка (i-го уровня);
2. Каждый узел i-го порядка порождает точно D^{n-i} n-го уровня;
3. Кодовая конструкция соответствует *префиксному коду*, т.е. никакое кодовое слово не является началом другого кодового слова, так как кодовые слова однозначно определяются конечными узлами.

Для простоты упорядочим длины кодовых слов

$$n_1 \le n_2 \le \cdots n_k = n. \tag{3.3}$$

Теперь начнем отсчет. Кодовое слово c_1 длины n_1 запрещает в точности D^{n-n_1} возможных конечных узлов на последнем n-ом уровне. Так как кодовые слова префиксного кода не сливаются, в совокупности получим

$$\sum_{k=1}^{K} D^{n-n_k} \tag{3.4}$$

запрещенных узлов на n-ом уровне. Общее число возможных узлов на n-ом уровне равно D^n , следовательно

$$\sum_{k=1}^{K} D^{n-n_k} \le D^n. \tag{3.5}$$

Разделив обе части неравенства на D^n, получим неравенство Крафта.

Шаг 2. *Утверждение Мак-Миллана.*

Каждый однозначно декодируемый код удовлетворяет неравенству Крафта.

При выводе неравенства Крафта были использованы особенности префиксных кодов. Но это условие не является необходимым. Как будет показано далее, необходимым условием является однозначная декодируемость кода. Возведем сумму в неравенстве Крафта в степень L

$$\left[\sum_{k=1}^{K} D^{-n_k}\right]^L = \sum_{k_1=1}^{K}\sum_{k_2=1}^{K}\cdots\sum_{k_L=1}^{K} D^{-[n_{k_1}+n_{k_2}+\cdots+n_{k_L}]}. \tag{3.6}$$

Обозначив через A_i число комбинаций, содержащих L кодовых слов с суммарной длиной i, запишем (3.6) в компактной форме

$$\left[\sum_{k=1}^{K} D^{-n_k}\right]^L = \sum_{i=1}^{L_{n,\max}} A_i D^{-i}, \tag{3.7}$$

где $L_{n,\max}$ максимальная длина сообщения, содержащего L кодовых слов.

Если код является однозначно декодируемый, то все последовательности из L кодовых слов суммарной длины i различны. Так как имеется всего D^i возможных последовательностей, то

$$A_i \le D^i \tag{3.8}$$

и, таким образом,

$$\left[\sum_{k=1}^{K} D^{-n_k}\right]^L \leq \sum_{i=1}^{L_{n,\max}} 1 = L_{n,\max}. \tag{3.9}$$

Извлекая L-ичный корень, получим оценку сверху для суммы в неравенстве Крафта

$$\sum_{k=1}^{K} D^{-n_k} \leq (L_{n,\max})^{1/2} \tag{3.10}$$

для всех натуральных L.

Обсудим значение числа L. Это число независимых кодовых слов, которые выбираются из $\{1, 2, ..., K\}$ и используются для построения всех возможных последовательностей длины, не превышающей $L_{n,\max}$. Поэтому, при $L \to \infty$, мы приходим к неравенству Крафта

$$\sum_{k=1}^{K} D^{-n_k} \leq \lim_{L\to\infty} (L_{n,\max})^{1/L} = 1. \tag{3.11}$$

Приведенные выше рассуждения справедливы для каждого однозначно декодируемого кода. Поэтому, каждый однозначно декодируемый код удовлетворяет неравенству Крафта.

Шаг 3.

Запишем левую часть неравенства (3.1) в виде

$$H(x) - \overline{n} \log D \leq 0. \tag{3.12}$$

Используя вероятность событий p_k и соответствующие длины кодовых слов n_k , имеем

$$\begin{aligned} H(x) - \overline{n} \log D &= \sum_{k=1}^{K} p_k \log \frac{1}{p_k} - \sum_{k=1}^{K} p_k n_k \log D = \\ &= \sum_{k=1}^{K} p_k \log \frac{D^{-n_k}}{p_k}. \end{aligned} \tag{3.13}$$

Как и при доказательстве утверждения (2.1), оценим сверху ло-

гарифмическую функцию при помощи (2.19) и получим

$$\sum_{k=1}^{K} p_k \log \frac{D^{-n_k}}{p_k} \le \log e \left(\sum_{k=1}^{K} p_k \left[\frac{D^{-n_k}}{p_k} - 1 \right] \right) = \\ = \log e \Bigg(\underbrace{\sum_{k=1}^{k} D^{-n_k}}_{\le 1} - \underbrace{\sum_{k=1}^{k} p_k}_{1} \Bigg) \le 0. \tag{3.14}$$

Здесь было использовано неравенство Крафта и условие $\sum\limits_{k=1}^{K} p_k = 1$. Таким образом, левая часть неравенства (3.1) теоремы о кодировании источников доказана.

Шаг 4.

Правая часть неравенства (3.1)

$$\overline{n} \le \frac{H(X))}{\log D} + 1. \tag{3.15}$$

При доказательстве используем неравенство Крафта, условие существования кода с соответствующими длинами кодовых слов n_k и условие $\sum_{k=1}^{K} p_k = 1$.

Неравенство Крафта можно записать в виде

$$\sum_{k=1}^{K} D^{-n_k} \le \sum_{k=1}^{K} p_k. \tag{3.16}$$

Пока имеет силу неравенство Крафта, мы свободны в выборе длин кодовых слов $n_k \in \mathcal{N}$. Выберем для каждого слагаемого такое наименьшее n_k, при котором

$$D^{-n_k} \le p_k. \tag{3.17}$$

Неравенство Крафта при таком выборе будет выполняться, следовательно, используя конструкцию рис. 3.2, мы можем построить такой префиксный код. Так как n_k наименьшее целое, при котором имеет место (3.17), то для $n_k - 1$ справедливо

$$p_k < D^{-(n_k - 1)}. \tag{3.18}$$

Остальная часть доказательства лишь формальна.

Используя свойства логарифмической функции, получаем

$$p_k \log p_k < p_k \log D^{-(n_k-1)} = p_k(-n_k+1)\log D. \qquad (3.19)$$

Суммируя по всем K, имеем

$$\underbrace{\sum_{k=1}^{K} p_k \log p_k}_{-H(X)} < \underbrace{\log D \sum_{k=1}^{K} p_k(-n_k)}_{-\bar{n}} + \log D \underbrace{\sum_{k=1}^{K} p_k}_{1}. \qquad (3.20)$$

Разделив обе части неравенства на $\log D$ и переставляя члены с домножением на -1 (что меняет знак неравенства), получаем искомое доказательство. ■

3.2. Коды Хаффмана

Коды Хаффмана[1] принадлежат к семейству кодов с переменной длиной кодовых слов. Самый известный код переменной длины – азбука Морзе[2] (табл. 3.1). Основная идея азбуки Морзе – передавать часто встречающиеся буквы кодовыми словами короткой длины и, наоборот, редко встречающиеся буквы – длинными кодовыми словами для того, чтобы минимизировать средние затраты. Такой способ кодирования называют так же кодированием с минимальной избыточностью или энтропийным кодированием.

Так, например, в азбуке Морзе часто встречающаяся буква «E» кодируется одним знаком «·», а редкая «X» – четырьмя знаками «— · · —».

В 1952 г. Хаффман показал, что предложенное им кодирование с переменной длиной кодовых слов является оптимальным префиксным кодом для дискретных источников без памяти. То есть, средняя длина слова кода Хаффмана минимальна и он так же является *кодом без запятой*. Термин «код без запятой» означает, что при установленной синхронизации возможна непрерывная передача потока сообщений с однозначным декодированием без специального разделения кодовых слов.

В *префиксном коде* никакое кодовое слово не является префиксом другого слова.

[1] Давид Хаффман (1925–1999) – американский ученый.
[2] Самуэль Морзэ (1791–1872) – американский художник и изобретатель.

Таблица 3.1. Буквы, символы азбуки Морзе и их относительная частота в немецком литературном тексте.

Буква	Символ азбуки Морзе	Относительная частота	Буква	Символ азбуки Морзе	Относительная частота
A	•■	0,0651	N	■•	0,0992
B	■•••	0,0257	0	■■■	0,0229
C	■•■•	0,0284	P	•■■•	0,0094
D	■••	0,0541	Q	■■•■	0,0007
E	•	0,1669	R	•■•	0,0654
F	••■•	0,0204	S	•••	0,0678
G	■■•	0,0365	T	■	0,0674
H	••••	0,0406	U	••■	0,0370
I	••	0,0782	V	•••■	0,0107
J	•■■■	0,0019	W	•■■	0,0140
K	■•■	0,0188	X	■••■	0,0002
L	•■••	0,0283	Y	■•■■	0,0003
M	■■	0,0301	Z	■■••	0,0100

Замечание. *Коды Хаффмана играют важнейшую роль в кодировании изображений. Они являются основной частью стандартов JPEG, MPEG и H.261 [19]. Кроме этого, они так же используются при оцифровке аудио сигналов. В литературе, в качестве примеров энтропийного кодирования, упоминаются коды Шеннона и Фано, но они не во всех случаях являются оптимальными и мы пропустим их описание.*

Кодирование Хаффмана производится за три шага. Мы наглядно поясним этот процесс на маленьком примере.

Кодирование Хаффмана.

1. **Упорядочение.** Расположить знаки в порядке убывания их вероятностей.

2. **Редукция.** Объединить два знака с наименьшими вероятно-

Таблица 3.2. Вероятности и энтропия двух символов.

x_i	a	b	c	d	e	f
p_i	0,05	0,15	0,05	0,4	0,2	0,15
$\frac{I(p_i)}{\text{бит}}$	4,32 бит	2,74 бит	4,32 бит	1,32 бит	2,32 бит	2,74 бит
$\frac{H(X)}{\text{бит}}$	$\approx 2{,}25$					

стями в один составной знак. Переупорядочить список знаков в соответствии с шагом 1. Продолжать этот процесс до тех пор, пока все знаки не будут объединены.

3. **Кодирование.** Начать с последнего объединения. Приписать первой компоненте составного знака символ «0», а второй - символ «1». Продолжать этот процесс до тех пор, пока все простые знаки не будут закодированы.

В случае, когда несколько знаков имеют одинаковые вероятности, объединяются те два из них, которые до этого имели наименьшее число объединений. Этим достигается выравнивание длин кодовых слов, что облегчает передачу информации.

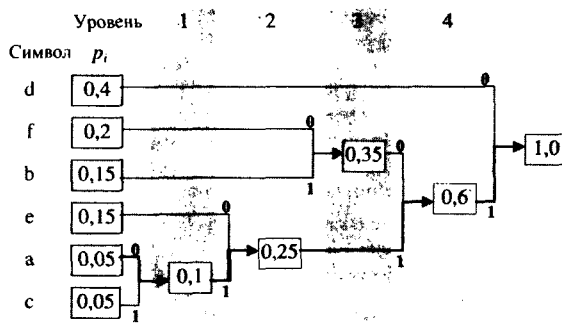


Рис. 3.3. Кодирование Хаффмана $n = 4$.

Пример:

Кодирование Хаффмана наглядно показано на примере источника, заданного табл. 3.2. При этом мы немного упростим кодирование,

Таблица 3.3. Кодирование кодом Хаффмана к рис. 3.3.

x_i	d	f	b	e	a	c
p_i	0,4	0,2	0,15	0,15	0,05	0,05
Кодовое слово	0	100	101	110	1110	1111
Длина кодового слова	1	3	3	3	4	4

отказавшись от наглядных переупорядочений знаков на втором шаге, т.к. в таком маленьком примере это можно сделать сделать «в уме». В соответствии с первым шагом, расположим все знаки в порядке убывания их вероятностей (рис.3.3).

На втором шаге объединим символы «a» и «c», обладающие наименьшими вероятностями, в составной символ «ac». Далее объединим «e» с «ac» в составной символ «eac» с вероятностью 0.25. Теперь наименьшими вероятностями уже обладают символы «f» и «b». Объединив их, получим символ «fb». Следующая редукция даст составной символ «fbeac» с вероятностью 0,6. И, наконец, объединив все символы, получим составной символ «dfbeac», вероятность которого равна 1.

На третьем шаге будем идти справа налево, приписывая верхним компонентам объединений символ «0», а нижним – «1». Полученные кодовые слова всех простых знаков с их длинами приведены в табл. 3.3.

Данный код обладает минимальной средней длиной кодового слова.

Средняя длина кодового слова определяется длинами всех слов n_i, «взвешенными» с соответствующими вероятностями p_i

$$\overline{n} = \sum_{i}^{N} p_i n_i. \tag{3.21}$$

В рассмотренном выше примере средняя длина кодового слова $\overline{n} = 2,3$ бит близка к энтропии $H(X) = 2,25$ бит. Важнейшей величиной, называемой эффективностью кода или фактором сжатия, является отношение средней длины к энтропии. В нашем примере эффективность кода равна $\eta = 0,976$.

Эффективность кода или *фактор сжатия*

$$\eta = \frac{H(x)}{\overline{n}}. \tag{3.22}$$

Из примера отчетливо видно, что чем больше разница между вероятностями символов, тем больше выигрыш кода Хаффмана по сравнению с простым блоковым кодированием.

Теорема Шеннона о кодировании источников показывает, насколько эффективным может быть такое кодирование. Но теория информации также указывает на то обстоятельство, что при кодировании могут появляться кодовые слова очень большой длины. Это обстоятельство может препятствовать практическому использованию теоремы кодирования источников.

Реализация декодера кода Хаффмана следует непосредственно из рис. 3.3. На рис. 3.4 представлено дерево, называемое кодовым деревом декодера.

Декодирование каждого нового слова начинается с исходного узла (корня) кодового дерева. Если мы принимаем «0», то идем по ребру, соответствующему нулю (по верхнему ребру). В нашем примере при этом мы достигаем оконечного узла d. Следовательно, был передан символ d и мы начинаем декодирование нового символа с корня.

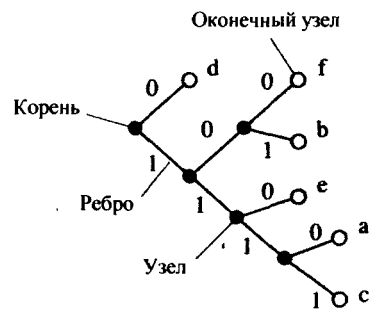


Рис. 3.4. Кодовая конструкция для $D = 2$ и $n = 4$.

Если мы принимаем «1», то идем по нижнему ребру и попадаем в узел, из которого выходят два ребра. Следующий принятый бит указывает, какое из этих двух ребер нужно выбрать. Мы продолжаем эту процедуру до тех пор, пока не достигнем оконечного узла. В этом случае мы принимаем решение о переданном символе и опять переходим к корню кодового дерева.

При всей простоте построения и декодирования, коды Хаффмана обладают тремя недостатками:

- Различные длины кодовых слов приводят к неравномерным за-

держкам декодирования.

- Сжатие данных снижает избыточность и поэтому повышает предрасположенность к распространению ошибок. В случае кодирования Хаффмана это означает, что один, ошибочно распознанный бит, может привести к тому, что все последующие символы будут декодированы неверно.

- Кодирование Хаффмана предполагает знание вероятностей событий (знаков), или, по крайней мере, подходящих оценок этих вероятностей. На практике очень часто вероятности событий неизвестны, а их оценки весьма затруднены.

Именно поэтому для сжатия больших массивов данных часто используют *универсальный алгоритм кодирования*, известный как алгоритм Лемпеля-Зива. Описание этого алгоритма приведено в разделе 6.3. Универсальный алгоритм сжатия не требует априорного знания статистики источника.

ГЛАВА 4

ЭНТРОПИЯ СВЯЗАННЫХ ИСТОЧНИКОВ

До сих пор в своих рассуждениях мы исходили из предположения независимости последовательных событий. Однако, стоит лишь только открыть немецкий орфографический словарь, мы сразу же обнаружим зависимость между рядом стоящими буквами, например: «qu», «ch», «ck», «tz» и «sch». Читая немецкий текст, мы видим, что после «q» за редким исключением следует «u». В этом случае «u», как почти неизбежное событие, практически не несет в себе никакой информации. Поэтому, при определении информации подобного рода источников, мы должны принимать во внимание взаимную связь между событиями.

4.1. Взаимная и условная информация

При аксиоматическом построении теории информации использовалось такое понятие, как информация пары событий. Напомним и обобщим эти рассуждения. Рассмотрим два дискретных источника X и Y. Объединим их события в пары событий (x_i, y_i). Мы получим простейшую модель связанных источников (рис.4.1).

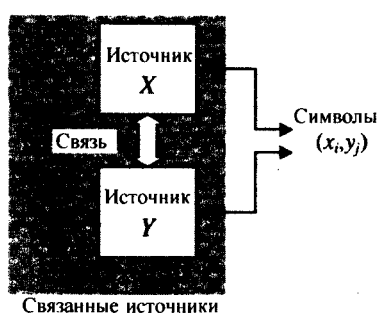


Рис. 4.1. Модель двух связанных источников.

Если оба источника каким-то образом связаны между собой, то следует ожидать, что событие одного источника позволяет делать некоторое предположение о событии другого. В терминах теории информации это означает, что неопределенность второго источника снижается, т.е. источники обмениваются взаимной информацией.

Введем условную вероятность $p(x/y)$ – вероятность события x при условии, что произошло событие y. Выразим совместную вероятность $p(x_i, y_i)$ двух событий x_i и y_i через их априорные и условные вероятности

$$p(x_i, y_i) = p(x_i/y_i)p(y_i) = p(y_i/x_i)p(x_i). \tag{4.1}$$

Используя логарифмическую функцию, сразу же получаем информации событий (x_i, y_i), (x_i) и (y)

$$\begin{aligned} \underbrace{\log_2 p(x_i, y_i)}_{-I(x_i,y_i) \text{ бит}} &= \log_2 p(x_i/y_i) + \underbrace{\log_2(p_i)}_{-I(y_i) \text{ бит}} = \\ &= \log_2 p(y_i/x_i) + \underbrace{\log_2 p(x_i)}_{-I(x_i) \text{ бит}}, \end{aligned} \tag{4.2}$$

то есть

$$I(x_i, y_i) = I(y_i) - \mathrm{ld}(x_i/y_i) \text{ бит} = I(x_i) - \log_2 p(y_i/x_i) \text{ бит}. \tag{4.3}$$

Прибавляя и одновременно вычитая $I(x_i)$ в первой части (4.3) и, соответственно, $I(y_i)$ во второй, получаем

$$\begin{aligned} I(x_i, y_i) &= (x_i) + I(y_i) - \log_2 \frac{p(x_i/y_i)}{p(x_i)} \text{ бит} = \\ &= I(x_i) + I(y_i) - \log_2 \frac{p(y_i/x_i)}{p(y_i)}. \end{aligned} \tag{4.4}$$

Таким образом, информация пары событий (x_i, y_i) определяется суммой информаций этих событий за вычетом некоторой неотрицательной величины, которая снижает неопределенность, т.е. сама в свою очередь является информацией. Поэтому, назовем ее взаимной информацией пары событий.

Взаимная информация пары событий определяется как

$$\begin{aligned} I(x_i; y_i) &= \log_2 \frac{p(x_i/y_i)}{p(x_i)} = \log_2 \frac{p(y_i/x_i)}{p(y_i)} = \\ &= \log_2 \left(\frac{\text{апостериорная вероятность}}{\text{априорная вероятность}} \right). \end{aligned} \tag{4.5}$$

Обратите внимание на то, что взаимная информация $I(x_i; y_i)$ всегда положительна. Важным свойством также является симметрия взаимной информации относительно источников, т.к.

$$\frac{p(x_i/y_i)p(y_i)}{p(x_i)p(y_i)} = \frac{p(y_i/x_i)}{p(y_i)}. \tag{4.6}$$

Симметрия относительно источников в (4.5) позволяет сделать вывод, что обмен информацией между источниками является взаимным, а не односторонним.

Для того, чтобы лучше представлять себе смысл взаимной информации, рассмотрим два граничных случая.

1. Источники независимы. Тогда для пары независимых событий имеем

$$p(x_i, y_i) = p(x_i)p(y_i), \tag{4.7}$$

то есть источники не обмениваются информацией

$$I(x_i; y_i) = 0. \tag{4.8}$$

2. Источники жестко связаны, то есть событие одного источника однозначно определяет событие другого

$$p(x_i/y_i) = p(y_i/x_i) = 1. \tag{4.9}$$

В этом случае происходит полный обмен информацией

$$I(x_i; y_i) = I(x_i) = I(y_i). \tag{4.10}$$

Из (4.4) следует, что информацию пары событий (x_i, y_i) можно интерпретировать, как разность между информацией пары независимых событий $I(x_i) + I(y_i)$ и заранее предсказанной взаимной информацией $I(x_i; y_i)$, обусловленной связанностью источников X и Y

$$I(x_i, y_i) = I(x_i) + I(y_i) - I(x_i, y_i). \tag{4.11}$$

Рассмотрим еще раз (4.3) и введем понятие условной информации.

Условная информация (апостериорная неопределенность)

$$I(x_i/y_i) = -\log_2 p(x_i/y_i) \text{ бит}. \tag{4.12}$$

Из (4.3) следует

$$I(x_i; y_i) = I(y_i) + I(x_i/y_i) = I(x_i) + I(y_i/x_i), \tag{4.13}$$

то есть информацию пары событий можно определить как сумму информации события y_i и информации события x_i при условии, что событие y_i уже известно, или, наоборот, как сумму информации события x_i и информации события y_i при условии, что событие x_i уже известно.

4.2. Совместная и условная энтропия

После рассмотрения отдельных пар событий в предыдущем разделе, перейдем к средним оценкам источника.

На рис. 4.2 показана исходная ситуация.

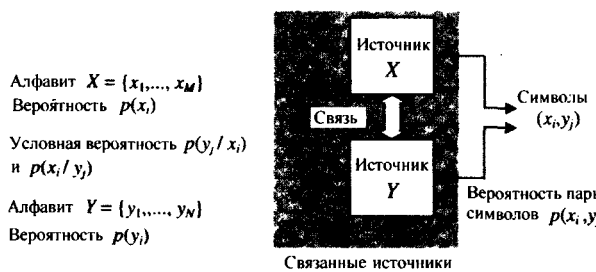


Рис. 4.2. Два связанных дискретных источника.

Совместная энтропия двух источников определяется как математическое ожидание информации всех пар событий.

Совместная энтропия двух дискретных источников без памяти X и Y

$$\frac{H(X,Y)}{\text{бит}} = -\sum_X \sum_Y p(x,y) \log_2 p(x,y). \qquad (4.14)$$

Замечание. *Здесь подразумевается, что рассматриваются все пары совместных событий, то есть*

$$\sum_X \sum_Y p(x,y) = \sum_{i=1}^{M} \sum_{j=1}^{N} p(x_i, y_j) = 1.$$

Усредняя условные информации всех пар событий, получим условную энтропию.

Таблица 4.1. Оценка совместной вероятности пар символов $p(x_i, y_j)$ и вероятность отдельных символов $p(x_i)$ и $p(y_j)$.

	y_1	y_2	y_3	y_4	$p(x_i)$
x_1	0,10	0,05	0,05	0	0,20
x_2	0,05	0,15	0,15	0	0,35
x_3	0	0,10	0,10	0,10	0,30
x_4	0	0,05	0,05	0,05	0,15
$p(y_j)$	0,15	0,35	0,35	0,15	1

Условная энтропия двух дискретных источников без памяти X и Y

$$\frac{H(Y/X)}{\text{бит}} = -\sum_X \sum_Y p(x,y)\log_2 p(y/x)$$
$$\frac{H(X/Y)}{\text{бит}} = -\sum_X \sum_Y p(x,y)\log_2 p(x/y). \tag{4.15}$$

Заменяя в (4.14) $\log_2 p(x,y)$ на $\log_2(p(x/y)p(y))$ и на $\log_2(p(y/x)p(y))$, получим

$$H(X,Y) = H(Y) + H(X/Y) = H(X) + H(Y/X). \tag{4.16}$$

Таким образом, совместная энтропия может быть представлена в виде суммы энтропии одного источника и некоторой части энтропии другого источника. Для независимых источников энтропия второго источника входит в сумму целиком, т.к. $H(X/Y) = H(X)$ и $H(Y/X) = H(Y)$. Для связанных источников всегда $H(X/Y) < H(X)$ и $H(Y/X) < H(Y)$. Поэтому, в общем случае, всегда имеет место

$$H(X,Y) \le H(Y) + H(X). \tag{4.17}$$

Пример: Связанные источники.

Сейчас самое время подробно разобрать числовой пример, наглядно поясняющий приведенные выше определения и формулы. Для этой цели была подобрана задача, методика решения которой может непосредственно использоваться на практике.

Пусть мы имеем выборку 100000 пар совместных событий (x_i, y_i) дискретных источников X и Y и алфавит каждого источника содержит четыре события. Пусть пара (x_1, y_1) встретилась 10000 раз.

Тогда оценка вероятности пары (x_1, y_1) равна $10000/10000 = 0,1$. Оценки остальных пар событий также получены подсчетами их относительной частоты и сведены в таблицу 4.1. Будем считаем, что полученные оценки близки к вероятностям пар событий и в дальнейшем будем говорить уже о вероятностях. Вероятности событий x_i, y_i получены суммированием строк и столбцов. Контрольная сумма $\sum_{i=1}^{4} x_i = \sum_{i=1}^{4} y_i = 1$ приведена в правом нижнем углу.

Теперь, когда нам известны все вероятности, необходимые для подсчета энтропий, определим:

1. Энтропии источников X и Y;
2. Совместную энтропию источников;
3. Обе условные энтропии;

   Для контроля мы также вычислим:

4. Условные вероятности $P(y_g/x_i)$;
5. Определим условную энтропию $H(Y/X)$.

Замечание. *Для простоты проведем расчеты с точностью до 4 знаков после запятой.*

Решение.

1.

$$\frac{H(X)}{\text{бит}} = \sum_{i=1}^{4} -p(x_i)\log_2 p(x_i) =$$

$$= -2[0,15 \cdot \log_2(0,15) + 0,35 \cdot \log_2(0,35)] = 1,9261, \qquad (4.18)$$

$$\frac{H(Y)}{\text{бит}} = \sum_{j=1}^{4} -p(y_j)\log_2 p(y_i) = 1,8813;$$

2.

$$\frac{H(X,Y)}{\text{бит}} = \sum_{i=1}^{4}\sum_{j=1}^{4} -p(x_i, y_j) = 3,4464; \qquad (4.19)$$

3. Без длинных вычислений из (4.16) получаем

$$H(X/Y) = H(X,Y) - H(Y) = 3,4464 - 1,8813 = 1,5651 \text{ бит},$$

$$H(Y/X) = H(X,Y) - H(X) = 3,4464 - 1,19261 = 1,5203 \text{ бит};$$

Таблица 4.2. Условная вероятность $p(y_i/x_j)$.

	y_1	y_2	y_3	y_4	$\sum_{j=1}^{4} p(y_j/x_i)$
x_1	1/2	1/4	1/4	0	1
x_2	1/7	3/7	3/7	0	1
x_3	0	1/3	1/3	1/3	1
x_4	0	1/3	1/3	1/3	1

4. В таблице 4.2 приведены условные вероятности, подсчитанные исходя из таблицы 4.1. Заметим, что при этом мы получили так называемую стохастическую матрицу. Сумма условных вероятностей для каждой строки равна 1.

5.
$$\frac{H(Y/X)}{\text{бит}} = -\sum_{i=1}^{4}\sum_{j=1}^{4} p(x_i, y_j)\log_2 p(y_i/x_i) = 1,5496. \qquad (4.20)$$

4.3. Выводы

Все приведенные в предыдущих разделах рассуждения в математической форме сведены в табл. 4.3. Напомним, что основной идеей *теории информации является представление информации источника* как меры неопределенности. Эта неопределенность раскрывается посредством экспериментов со случайными событиями из алфавита этого источника. Такой подход поясняют три столбца таблицы.

Так как информация исходит из случайности событий, в первом столбце вводится понятие вероятности событий и совместной вероятности пары событий как основополагающих величин. Для пары событий вводится также понятие условной вероятности. Во втором столбце дается определение информации события и пары событий, а также условной и взаимной информации. И, наконец, в третьем столбце, вводится понятие энтропии как меры неопределенности источника.

Энтропия источника, совместная и условная энтропии двух источников трактуются как математические ожидания соответствующих информаций событий. Условная вероятность – это вероятность одного события при условии, что другое событие уже произошло, по-

Таблица 4.3. Дискретные источники без памяти X и Y с символами $x \in X = \{x_1, x_2, \ldots, x_M\}$ и $y \in Y = \{y_1, y_2, \ldots, y_N\}$.

	Информации	Энтропия
Вероятность отдельного символа (априорная вероятность) $p(x)$	Информация отдельного символа $I(x) = -\log_2 p(x)$ бит	Энтропия $H(X) = -\sum_X p(x)\log_2 p(x)$ бит $H(Y) = -\sum_Y p(y)\log_2 p(y)$ бит
Совместная вероятность двух символов $p(x,y)$	Информация пары символов $I(x,y) = -\log_2 p(x,y)$ бит	$H(X,Y) = -\sum_X \sum_Y p(x,y)\log_2 p(x,y)$ бит
Условная вероятность (апостериорная вероятность) $p(x/y) = \frac{p(x,y)}{p(y)}$ $p(y/x) = \frac{p(x,y)}{p(x)}$	Условная информация $I(x/y) = -\log_2 p(x/y)$ бит $I(y/x) = -\log_2 p(y/x)$ бит	$H(X/Y) = -\sum_X \sum_Y p(x,y)\log_2 p(x/y)$ бит $H(Y/X) = -\sum_X \sum_Y p(x,y)\log_2 p(y/x)$ бит
	Взаимная информация $I(x;y) = \log_2 \frac{\text{апостериорная вероятность}}{\text{априорная информация}}$ бит $= \log_2 \frac{p(x/y)}{p(x)}$ бит $= \frac{p(x/y)}{p(x)}$ бит	

этому, понятия условной информации и условной энтропии вполне естественно выводятся из условной вероятности.

Взаимная информация не имеет аналога в теории вероятности. Это совершенно новое понятие теории информации, играющее центральную роль в информационной технике. Взаимная информация связывает понятие канала с возможностью передачи информации по нему, т.е. с его пропускной способностью. Это понятие будет подробно рассмотрено в 7 главе этой книги.

ГЛАВА 5

СТАЦИОНАРНЫЕ ДИСКРЕТНЫЕ ИСТОЧНИКИ С ПАМЯТЬЮ

5.1. Энтропия

Сигналы *аналоговых источников информации* ограничены по полосе, поэтому коррелированы во времени. Примером может служить аналоговый речевой сигнал в телефонной линии. После оцифровки, аналоговый источник превращается в дискретный и, например, после квантования сигнала на 256 уровней, мы получаем последовательнось 8-ми битовых двоичных целых чисел от 0 до 255. Как видно из рис. 5.1, значение двух соседних чисел близки друг к другу, т.к. телефонный сигнал передается в узкой полосе частот. Из-за временной связи соседних отсчетов, то есть памяти отсчета, его неопределенность (информация) снижается по сравнению с аналоговым источником без памяти, поэтому основной задачей методов сжатия, особенно при передаче видеосигналов, является снижение избыточности.

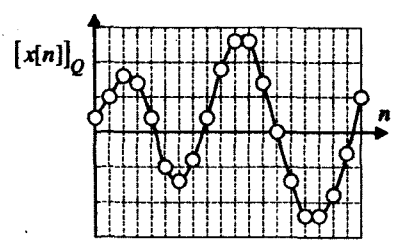


Рис. 5.1. Непрерывный сигнал.

Возникает вопрос о том, каким образом определить энтропию *дискретного источника с памятью*. Начнем с постановки задачи.

Определение 5.1.1. *Дискретный источник* X можно представить как *дискретный во времени стохастический процесс*, реализацией

которого является последовательность событий x_n, принадлежащих алфавиту источника $X = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$.

Замечание. *Во избежание путаницы, мы обозначили содержимое алфавита греческими буквами. В этом случае на месте переменной x_n n-го события может быть поставленно любое число из алфавита X.*

Определение 5.1.2. Дискретный источник является *стационарным*, если совместные вероятности последовательностей событий не зависят от выбора начальной точки отсчета времени.

Замечание. *Независимость наблюдений от точки отсчета означает, что мы можем начинать выборку с любого момента времени, то есть статистика не зависит от времени начала наблюдений.*

Определение 5.1.3. Стационарный дискретный источник математически полностью описан, если известны все совместные вероятности $p(x_{n_1}, x_{n_2}, \ldots, x_{n_M})$ для любой выборки $n_1, n_2, \ldots, n_M$, где $M \to \infty$.

Определение энтропии стационарного дискретного источника с памятью следует из двух подходов, приводящих к одинаковому результату. При первом подходе мы используем понятие совместной энтропии, при втором - условной энтропии. В обоих случаях мы ищем ответ на следующий вопрос: «Если память источника распространяется на несколько последовательных событий, то какую дополнительную информацию несет отдельное событие в том случае, если блок предшествующих событий уже известен?»

Подход 1. Совместная энтропия.

Совместная энтропия двух источников x_1 и x_2 с одинаковыми алфавитами и одинаковыми распределениями вероятностей событий определяется как

$$H(x_1, x_2) = -\sum_{i=1}^{M}\sum_{j=1}^{M} p(x_i, x_j)\log_2 p(x_i, x_j). \qquad (5.1)$$

Распространим это определение на L последовательных источников X_i и найдем энтропию источника X_L как

$$H_L(X) = \frac{1}{L}H(X_1, X_2, \ldots, X_L) = -\frac{1}{L}\sum_{X_L} p(x)\log_2 p(x), \qquad (5.2)$$

где вектор $X = (x_1, x_2, \ldots, x_L)$ и суммирование производится по всем возможным компонентам вектора X. Устремляя L к бесконечности, мы полностью охватим память источника и получим предельное значение $H_L(X)$ (если оно существует), равное

$$H_\infty(X) = \lim_{L\to\infty} H(X_L). \tag{5.3}$$

Подход 2. Условная энтропия.

Условная энтропия L-того события в случае, если $L-1$ предшествующих событий уже известны, определяется как

$$H_\infty(X) = \lim_{L\to\infty} H(X_L/X_1, X_2, \ldots, X_{L-1}). \tag{5.4}$$

Хотя в левых частях равенств (5.3) и (5.4) мы уже использовали одинаковое обозначение энтропии отдельного события, этот факт предстоит доказать. Проведем это доказательство за 4 шага.

Теорема 5.1.1. Для стационарного дискретного источника с $H_L(x) < \infty$ имеет место:

1. $H(X_L|X_1, X_2, \ldots, X_{L-1})$ не возрастает с ростом длины блока L;
2. $H_L(X) \geq H(X_L|X_1, X_2, \ldots, X_{L-1})$;
3. $H_L(X)$ не возрастает с ростом длины блока L;
4. Энтропия стационарного дискретного источника $H_L(X)$ $\lim\limits_{L\to\infty} H_L(X) = \lim\limits_{L\to\infty} H(X_L|X_1, X_2, \ldots, X_{L-1}) = H_\infty(X)$.

Доказательство.

1. Из определения энтропии, как меры неопределенности источника, непосредственно следует, что возрастание числа ограничений не может повлечь за собой рост неопределенности, а следовательно и энтропии.

2. Из «правила цепочки» для совместной энтропии следует

$$\begin{aligned} H_L(X) = \frac{1}{L}[H(X_1) + H(X_2|X_1) + \cdots + \\ + H(X_L|X_1, X_2, \ldots, X_{L-1})]. \end{aligned} \tag{5.5}$$

Замечание. *«Правило цепочки» для совместной энтропии следует из «правила цепочки» для вероятностей. Простейший пример «правила цепочки» для вероятностей* $p(x,y) = p(x/y)p(y)$ *и* $p(x,y,z) = p(x/yz)p(y/z)p(z)$. *Так как логарифмическая функция отображает произведение в сумму, получаем «правило цепочки» для совместной энтропии.*

Так как энтропия всегда неотрицательна и имеет место неравенство

$$H(X_1) \geq H(X_2|X_1) \geq \cdots \geq, H(X_L|X_1, X_2, \ldots, X_{L-1}), \tag{5.6}$$

откуда следует нижняя оценка 2.

3. Из (5.5) прежде всего следует разложение

$$H_L(X) = \frac{L-1}{L} H_{L-1}(X) + \frac{1}{L} H(X_L|X_1, X_2, \ldots, X_{L-1}). \tag{5.7}$$

Используя уже известное соотношение (5.5), получаем неравенство

$$L \cdot H_L(X) \leq (L-1) \cdot H_{L-1}(X) + H_L(X). \tag{5.8}$$

После подстановки получаем утверждение 3.

$$H_L(X) \leq H_{L-1}(X). \tag{5.9}$$

4. Утверждения 1., 2. и ограничение $H_1(X) < \infty$ устанавливают существование предела. Используя далее «правило цепочки», получаем

$$\begin{aligned} H_{L+j}(X) = \frac{1}{L+j}[&H(X_1, X_2, \ldots, X_{L-1}) + \\ &+ H(X_L|X_1, X_2, \ldots, X_{L-1}) + H(X_{L+1}|X_1, X_2, \ldots, X_L) + \\ &+ \cdots + H(X_{L+j}|X_1, X_2, \ldots, X_{L+j-1})]. \end{aligned} \tag{5.10}$$

Согласно утверждению 1., условная энтропия в правой части равенства не возрастает, поэтому справедлива оценка

$$\begin{aligned} H_{L+j}(X) \leq \frac{1}{L+j} &H(X_1, X_2, \ldots, X_{L-1}) + \\ &+ \frac{j+1}{L+j} H(X_L/|X_1, X_2, \ldots, X_{L-1}). \end{aligned} \tag{5.11}$$

Устремляя j к бесконечности, получим

$$\begin{aligned} \lim_{j\to\infty} H_{L+j}(X) \leq \lim_{j\to\infty} &\frac{1}{L+j} H(X_1, X_2, \ldots, X_{L-1}) + \\ &+ \frac{j+1}{L+j} H(X_L|X_1, X_2, \ldots, X_{L-1}), \end{aligned} \tag{5.12}$$

что дает для каждого натурального L,

$$H_\infty \leq H(X_L|X_1, X_2, \ldots, X_{L-1}), \tag{5.13}$$

но, т.к. для любого натурального L выполняется так же и 2., то 5.13 превращается в равенство при $L \to \infty$. ■

5.2. Теорема кодирования источников 2

Теперь мы можем дополнить теорию информации еще одной теоремой. Оказывается, что объединяя события источника в блоки длины L и кодируя эти блоки, средняя длина кодового слова на событие может достигнуть энтропию источника $H_\infty(x)$ при $L\to\infty$ как угодно близко. При этом память источника полностью учитывается.

Теорема 5.2.1. *Теорема кодирования* стационарного дискретного источника с энтропией $H_L(X)$.

Для блока длины L существует D-ичный префиксный код, в котором средняя длина кодового слова на одно событие $\overline{n}$ удовлетворяет неравенству

$$\frac{H_L(X)}{\log D} \le \overline{n} \le \frac{H_L(X)}{\log D} + \frac{1}{L}. \tag{5.14}$$

Теорема (5.14) не нуждается в специальном доказательстве. Если мы рассматриваем блоки длины L как новые независимые события с энтропией, равной $L \cdot H_L(x)$ и применяем теорему кодирования источников 1, то мы имеем

$$\frac{LH_L(X)}{\log D} \le L\overline{n} \le \frac{LH_L(X)}{\log D} + 1. \tag{5.15}$$

Разделив все члены неравенства на L, получаем среднюю длину кодового слова на событие

$$\frac{H_L(X)}{\log D} \le \overline{n} \le \frac{H_L(X)}{\log D} + \frac{1}{L}. \tag{5.16}$$

При $L \to \infty$, $H_L(x) \to H_\infty(x) = H(x)$ мы имеем

$$\frac{LH_L(X)}{\log D} \le \overline{n}L \le \frac{LH_L(X)}{\log D} + 1, \tag{5.17}$$

то есть

$$\frac{H_\infty(X)}{\log D} \le \overline{n} \le \frac{H_\infty(X)}{\log D} + \delta. \tag{5.18}$$

Таким образом, для любого сколь угодно малого δ, существует метод кодирования блоков, содержащих $L > 1/\delta$ событий, при котором для средней длины кодового слова на событие $\overline{n}$ выполняется неравенство (5.18). Теорема кодирования источников 2 показывает, что увеличивая длину блока L, мы можем как угодно близко подойти к энтропии $H(x) = H_\infty(x)$. Однако, на практике существуют некоторые ограничения. Для того, чтобы блок из L событий мог быть

продекодирован, он должен быть полностью принят, что может привести к недопустимым задержкам декодирования и недопустимому объему буфера памяти.

5.3. Конечные цепи Маркова

В этом и последующих параграфах будет рассматриваться специальная форма дискретных источников с памятью – марковские источники. Их описание сходно с марковскими цепями, которые нашли разнообразное применение в других областях науки и техники. Так, на основе марковских цепей строятся модели распознавания речи, модели передачи по телефонным коммутируемым каналам. Цепи Маркова[1] используются при исследовании моделей сетей связи (каналы Гильберта-Элиота) и в теории управления транспортными потоками. Значение цепей Маркова основывается не только на их полном математическом описании, но также на том факте, что с их помощью можно составить математическую модель многих процессов, наиболее близкую к практике.

5.3.1. Дискретные во времени цепи Маркова

В этом разделе шаг за шагом вводится понятие конечных дискретных во времени марковских цепей. Мы наглядно поясним это понятие на простейшем примере «случайных блужданий».

Пример: Случайные блуждания студента.

Нас интересует вопрос о том, доберется ли пьяный студент от дверей пивной до дверей студенческого общежития. Поставим вопрос по другому (рис. 5.2): какова вероятность того, что случайные блуждания на 7 временном шаге приведут студента в пространственное состояние S_1? или

P («случайные блуждения» приведут к состоянию S_1 в момент времени $n = 7$).

Ситуация, изображенная на рис. 5.2 уже содержит в себе важнейшие признаки цепи Маркова. Под марковской цепью понимается дискретный во времени и по состоянию марковский процесс $S(n)$. Его реализацией является множество путей, ведущих из состояний S_1 в состояние S_i.

[1] А. А. Марков (1856 1922) – выдающийся русский математик. *Прим. перев.*

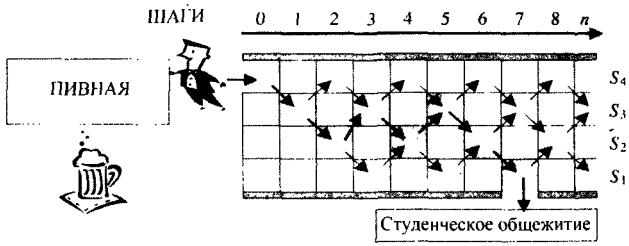


Рис. 5.2. Случайные блуждания студента.

Исходным пунктом для описания марковской цепи является *множество состояний*

$$\mathbf{S} = \{S_1, S_2, \ldots, S_N\}, \tag{5.19}$$

где N натуральные и *стохастический вектор* распределения вероятностей состояний в момент времени n

$$\mathbf{p}_n = (p_n(1), p_n(2), \ldots, p_n(N)). \tag{5.20}$$

Для того, чтобы полностью определить цепь Маркова, нам остается задать метод подсчета вероятностей $p_n(i)$ для любого момента времени n. Из определения вероятности имеем

$$0 \le p_n(i) \le 1 \ \forall i = 1, 2, \ldots, N \text{ и } \sum_{i=1}^{N} p_n(i) = 1. \tag{5.21}$$

Особое значение имеет распределение вероятностей в начале наблюдения, т.е. начальные условия

$$\mathbf{p}_0 = (p_0(1), p_0(2), \ldots, p_0(N)). \tag{5.22}$$

Смена состояний описывается *переходными вероятностями*

$$\pi_{n_2,n_1}(j/i) = P(s[n_1] = s_i \cap s[n_2] = S_j). \tag{5.23}$$

Эти переходные вероятности обладают следующими свойствами:

1. Одно из состояний в момент времени n всегда достигается при любом S_i в момент n_1

$$\sum_{j=1}^{N} \pi_{n_2,n_1}(j/i) = 1. \tag{5.24}$$

2. Граничное распределение, т.е. вероятности j-го состояния в момент времени n_2

$$\sum_{j=1}^{N} \pi_{n_2,n_1}(j/i) = p_{n_2}(j). \tag{5.25}$$

3. Рекурсивная формула для переходных вероятностей (частный случай уравнения Колмогорова – Чэпмена.)

$$\pi_{n_3,n_1}(j/i) = \sum_{l=1}^{N} \pi_{n_3,n_2}(j/l)\pi_{n_2,n_1}(l/i). \tag{5.26}$$

Дискретное равенство Колмогорова - Чэпмена получается трансформацией «правила цепочки» для вероятностей

$$p(x_1, x_2) = p(x_2/x_1)p(x_1), \tag{5.27}$$

$$p(x_1, x_2, x_3) = p(x_3/x_1, x_2)p(x_2/x_1)p(x_1). \tag{5.28}$$

Умножая обе части равенства на $1/p(x_1)$, получим

$$p(x_2, x_3/x_3) = p(x_3/x_1, x_2)p(x_2/x_1). \tag{5.29}$$

Суммируя по всем x_2, приходим к дискретной форме *уравнения Колмогорова – Чэпмена*

$$p(x_3/x_1) = \sum_{X_2} p(x_3/x_1, x_2)p(x_2/x_1). \tag{5.30}$$

При преобразовании (5.30) в (5.26), мы предполагаем

$$p(x_3/x_1, x_2) = p(x_3/x_2). \tag{5.31}$$

Последнее равенство характеризует марковский процесс.

Определение 5.3.1. *Марковским процессом* называется процесс, в котором прошлое не оказывает влияния на будущее, если настоящее известно.

Замечание. *В рассмотренном примере известное прошлое это стохастическая переменная X_1, известное настоящее - X_2, неизвестное будующее - X_3.*

Применение цепей Маркова сильно упрощается, когда они гомогенны, стационарны и регулярны. Рассмотрим эти три понятия.

Определение 5.3.2. *Цепь Маркова гомогенна*, если переходные вероятности между состояниями не зависят от выбора временной точки отсчета.

Таким образом, переходные вероятности зависят только от разности времен отсчетов l.

$$\pi(j/i) = p(s[n] = s_i \cup s[n+l] = s_j)\ n = 0, 1, 2, 3, \ldots\ . \tag{5.32}$$

В этом случае для $l = n_2 - n_1$ и $m = n_3 - n_2$ равенство Колмогорова – Чэпмена (5.26) можно записать в виде

$$\pi_{l+m}(j/i) = \sum_r \pi_m(j/r)\pi_l(r/i). \tag{5.33}$$

Это равенство может быть представлено в виде суммы покомпонентных произведений векторов-строк на векторы-столбцы. Записав переходные вероятности в виде матрицы, получим уравнение (5.33) в матричной форме

$$\begin{pmatrix} \pi_{l+m}(1/1) & \pi_{l+m}(2/1) & \cdots & \pi_{l+m}(N/1) \\ \pi_{l+m}(1/2) & \pi_{l+m}(2/2) & \cdots & \pi_{l+m}(N/2) \\ \vdots & \vdots & \ddots & \vdots \\ \pi_{l+m}(1/N) & \pi_{l+m}(2/N) & \cdots & \pi_{l+m}(N/N) \end{pmatrix} = \tag{5.34}$$

$$\begin{pmatrix} \pi_l(1/1) & \pi_l(2/1) & \cdots & \pi_l(N/1) \\ \pi_l(1/2) & \pi_l(2/2) & \cdots & \pi_l(N/2) \\ \vdots & \vdots & \ddots & \vdots \\ \pi_l(1/N) & \pi_l(2/N) & \cdots & \pi_l(N/N) \end{pmatrix} \cdot \begin{pmatrix} \pi_m(1/1) & \pi_m(2/1) & \cdots & \pi_m(N/1) \\ \pi_m(1/2) & \pi_m(2/2) & \cdots & \pi_m(N/2) \\ \vdots & \vdots & \ddots & \vdots \\ \pi_m(1/N) & \pi_m(2/N) & \cdots & \pi_m(N/N) \end{pmatrix}.$$

Этот процесс может быть начат с первого шага с помощью *матрицы переходных вероятностей*

$$\mathbf{\Pi} = \begin{pmatrix} \pi(1/1) & \pi(2/1) & \cdots & \pi(N/1) \\ \pi(1/2) & \pi(2/2) & \cdots & \pi(N/2) \\ \vdots & \vdots & \ddots & \vdots \\ \pi(1/N) & \pi(2/N) & \cdots & \pi(N/N) \end{pmatrix}. \tag{5.35}$$

Заметим, что матрица Π – *стохастическая матрица* с суммой вероятностей строк равной 1.

Многократно применяя такое матричное преобразование к исходному распределению состояний p_0, мы можем получить распределение вероятностей p_n в любой момент времени n

$$\mathbf{p}_n = \mathbf{p}_0\mathbf{\Pi}^n. \tag{5.36}$$

Теорема 5.3.1. *Гомогенная цепь Маркова* полностью характеризуется *матрицей переходных вероятностей* и исходным распределением состояний.

Приведенные выше рассуждения можно наглядно обобщить при помощи графа состояний (рис. 5.3). Здесь узлы соответствуют состояниям, а пути между ними — переходам между состояниями. Каждому пути приписан вес, равный переходной вероятности. Таким образом, граф состояний даст полную информацию о матрице переходных вероятностей.

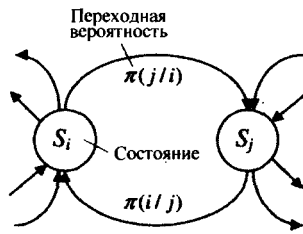


Рис. 5.3. Граф состояний гомогенной цепи Маркова.

Пример: Случайные блуждания студента (продолжение).

Случайные блуждания (см. рис. 5.2) представлены на рис. 5.4 в виде графа состояний.

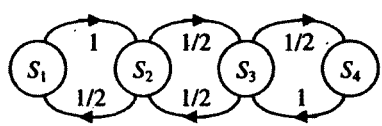


Рис. 5.4. Граф состояний для случайных блужданий студента.

Графу состояний соответствует матрица переходных вероятностей

$$\mathbf{\Pi} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{5.37}$$

Из рис. 5.2 следует, что исходное распределение состояний

$$\mathbf{p}_0 = (0,\ 0,\ 0,\ 1). \tag{5.38}$$

Вероятность того, что в результате случайных блужданий студент на 7-ом временном шаге окажется у дверей общежития, определяется первой компонентой вектора состояний на 7 шаге P_7. Из (5.36) следует

$$\mathbf{p}_7 = \mathbf{p}_0\mathbf{\Pi}^7 \tag{5.39}$$

Расчеты с помощью компьютера дают

$$\mathbf{\Pi}^7 = \begin{pmatrix} 0 & 0,619 & 0 & 0,3281 \\ 0,3359 & 0 & 0,6641 & 0 \\ 0 & 0,6641 & 0 & 0,3359 \\ 0,3281 & 0 & 0,6719 & 0 \end{pmatrix}. \tag{5.40}$$

Отсюда

$$\mathbf{p}_7 = (0,\ 0,\ 0,\ 1) \cdot \mathbf{\Pi}^7 = (0,3281,\ 0,\ 0,6719,\ 0) \tag{5.41}$$

и искомая вероятность равна

$$p_7(1) = 0,3281. \tag{5.42}$$

Важнейшим частным случаем марковской цепи является случай, когда распределение состояний не зависит от времени наблюдения.

Определение 5.3.3. *Гомогенная цепь Маркова стационарна* , если распределение состояний постоянно во времени. В этом случае начальное распределение является *собственным вектором* матрицы переходных вероятностей , т.е

$$\mathbf{p}_0\mathbf{\Pi} = \mathbf{p}_0. \tag{5.43}$$

Замечание. *Если вектор распределения состояний является стохастическим вектором с суммой компонентов, равной 1, то сумма компонент собственного вектора также равна 1.*

Для того, чтобы цепь Маркова была стационарной, должно выполняться (5.43). Пусть $\mathbf{p}_0 = (p_1,\ p_2,\ p_3,\ p_4)$, тогда

$$\mathbf{p}_0\mathbf{\Pi} = (p_1,\ p_2,\ p_3,\ p_4) \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \tag{5.44}$$

$$= (p_2/2,\ p_1 + p_3/2,\ p_2/2 + p_4,\ p_3/2),$$

$$p_1 + p_2 + p_3 + p_4 = 1. \tag{5.45}$$

Используя условия (5.43) и (5.45), находим стационарное распределение состояний

$$\mathbf{p}_0 = (1/6,\ 1/3,\ 1/3,\ 1/6). \tag{5.46}$$

Из рекурсивного соотношения (5.36) возникают следующие важнейшие вопросы: Что происходит по «истечении долгого времени», т.е. при $n \to \infty$? Устанавливается ли стационарное распределение состояний? Имеется ли нечто подобное стационарному распределению, например, два устойчивых распределения состояний?

Определение 5.3.4. *Гомогенная цепь Маркова* называется *регулярной*, если:

- *Предельная матрица*

$$\mathbf{\Pi}_\infty = \lim_{n\to\infty} \mathbf{\Pi}^n \tag{5.47}$$

существует, причем, все N строк предельной матрицы представляют собой *предельное распределение* $\mathbf{p}_\infty$;

- Предельное распределение является единственным *стационарным распределением* вероятностей состояний любой регулярной цепи Маркова;

- Цепь Маркова всегда *регулярна*, если существует некоторое натуральное n, при котором все компоненты некоторого столбца матрицы $\mathbf{\Pi}^n$ отличны от нуля.

Последнее из утверждений определения 5.3.4 равносильно следующему: цепь Маркова является регулярной, если на некотором шаге n существует по меньшей мере одно состояние, которое может быть достигнуто из любого начального состояния.

Пример: Случайные блуждания (продолжение).

Рассмотрим пример случайных блужданий студента и выясним, является ли соответствующая этим блужданиям цепь Маркова регулярной.

Матрица переходных вероятностей (5.37) имеет два граничных состояния в зависимости от того, является ли число временных шагов n четным или нечетным

$$\lim_{N\to\infty} \mathbf{\Pi}^{2N} = \begin{pmatrix} 1/3 & 0 & 2/3 & 0 \\ 0 & 2/3 & 0 & 1/3 \\ 1/3 & 0 & 2/3 & 0 \\ 0 & 2/3 & 0 & 1/3 \end{pmatrix}, \tag{5.48}$$

$$\lim_{N\to\infty} \mathbf{\Pi}^{2N+1} = \begin{pmatrix} 0 & 2/3 & 0 & 1/3 \\ 1/3 & 0 & 2/3 & 0 \\ 0 & 2/3 & 0 & 1/3 \\ 1/3 & 0 & 2/3 & 0 \end{pmatrix}, \tag{5.49}$$

поэтому, в данном случае цепь Маркова не является регулярной.

С другой стороны, не выполняется и последнее условие из определения 5.3.4, так как на каждом шаге все четные состояния переходят в нечетные и наоборот (см. рис. 5.2).

Замечание. *Если мы выберем начальное распределение, например равное* $\mathbf{p}_0$ *из (5.46), то на любом временном шаге любое из состояний достижимо.*

Пример: Марковская цепь с тремя состояниями.

Пусть марковская цепь задана графом состояний (рис. 5.5)

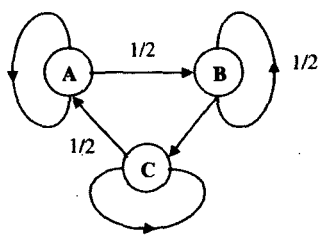


Рис. 5.5. Граф состояний.

1. Постройте матрицу переходных вероятностей.
2. Покажите, что цепь Маркова стационарна. При этом исходите из равномерного начального распределения состояний.
3. Покажите, что цепь Маркова регулярна.
4. Постройте предельную переходную матрицу.

Решение.

1. Матрица переходных вероятностей

$$\mathbf{\Pi} = \begin{pmatrix} 1/2 & 1/2 & 0 \\ 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \end{pmatrix}; \tag{5.50}$$

2. Стационарность.

Так как начальное распределения состояний равномерно, то

$$\mathbf{p}_0 = 1/3(1,\ 1,\ 1), \tag{5.51}$$

при этом выполняется условие (5.4)

$$\mathbf{p}_0\mathbf{\Pi} = 1/3(1,\ 1,\ 1)\begin{pmatrix} 1/2 & 1/2 & 0 \\ 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \end{pmatrix} = 1/3(1,\ 1,\ 1) = p_0; \tag{5.52}$$

3. Цепь Маркова регулярна, так как

$$\mathbf{\Pi}^2 = \left(\frac{1}{2}\right)\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} = 1/4\begin{pmatrix} 1 & 2 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 1 \end{pmatrix} \tag{5.53}$$

и для $\mathbf{\Pi}^2$ выполняется последнее из условий определения 5.3.4;

4. Предельная переходная матрица.

Так как цепь Маркова регулярна, воспользуемся определением 5.3.4, согласно которому $p_\infty = p_0$ и из 5.51 имеем

$$\mathbf{\Pi}_\infty = \begin{pmatrix} \mathbf{p}_0 \\ \mathbf{p}_0 \\ \mathbf{p}_0 \end{pmatrix} = 1/3\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{5.54}$$

5.3.2. Конечные дискретные марковские источники с памятью r

Марковские цепи можно с успехом использовать для моделирования конечных дискретных источников с памятью. Предполагая, что стохастические параметры источников с памятью могут быть подсчитаны как средние по времени величины (то есть источники обладают свойством эргодичности), наметим пути дальнейших рассуждений.

Пусть задана произвольная последовательность $\{x[n]\} = \{a, b, a, c, b, b, a, d, a, d, b, b, a, c, \ldots\}$ источника с алфавитом $X = \{a, b, c, d\}$. Мы уже ранее определили частоты событий, как оценки для вероятностей событий $p(a)$, $p(b)$, $p(c)$ и $p(d)$ и нашли энтропию источника, считая события независимыми. Если источник обладает памятью, то его энтропия может быть только меньше, то есть ранее мы находили оценку сверху.

Возникает вопрос, каким образом можно включить в анализ память источника.

Для этого необходимо учитывать зависимость между событиями. Оценим условные вероятности $p(a/a)$, $p(b/a)$, $p(c/a)$, $p(d/a)$, $p(a/b)$, ... и $p(d/d)$ двух последовательных событий путем подсчета частот парных событий. После этого источник может быть разложен на четыре подисточника, соответствующих первым символам в парных событиях.

На рис. 5.6 этот первый шаг рассуждений наглядно продемонстрирован. Здесь символ a определяет один из четырех подисточников. Событиям, происходящим за символом a (путям на графе), приписываются веса, равные вероятностям, например $p^{(a)}(b) = p(b/a)$. Таким образом, каждый такой подисточник уже может рассматриваться как некоторый самостоятельный источник без памяти. Энтропия такого источника может быть вычислена известными методами. Исходный источник с памятью представляет собой стохастическую совокупность четырех подисточников без памяти, а его энтропия определяется средними значениями энтропий этих подисточников. Мы можем продолжить рассуждения, рассматривая все более длинные состояния подисточников (например, векторы a, a или a, b, c, d) до тех пор, пока вся память исходного источника не будет охвачена.

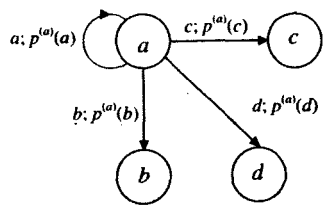


Рис. 5.6. Представление источника в виде цепи Маркова (первый шаг).

Эти эвристические рассуждения обобщены в следующем определении.

Определение 5.3.5. Конечный дискретный *марковский источник* с *памятью* r полностью определяется следующими условиями:

1. Задано непустое множество состояний $\mathbf{S} = \{S_1, S_2, \ldots, S_N\}$, причем, S содержит векторы длины r;

2. Каждое состояние S_i соответствует дискретному источнику без памяти с алфавитом $X_i = \{x_1, x_2, \ldots, x_M\}$ и вероятностями j-ых символов алфавита $p^{(i)}(j)$;

3. Состояние $S[n] = (x[n-r], x[n-r+1], \ldots, x[n-1])$ из $r-1$ последовательных символов и очередной символ $x[n]$ образуют новое состояние $S[n+1] = (x[n-r+1], x[n-r+2], \ldots, x[n])$;

4. Задано начальное распределение состояний $\mathbf{p}_0 = (p_0(1), p_0(2), \ldots, p_0(N))$.

Мы видим, что память r охватывает r последовательных символов, так как на вероятность очередного символа оказывает влияние в точности r предыдущих символов. Поясним это более подробно на примере.

Пример: Марковский источник с памятью $r = 2$.

Рассмотрим двоичный источник с алфавитом $X = \{0, 1\}$. Комбинации двух символов дают четыре состояния

$$\mathbf{S} = \{S_1 \cong (0,0), S_2 \cong (1,1), S_3 \cong (0,1), S_4 \cong (1,0)\}. \tag{5.55}$$

Переходные вероятности между состояниями задаются величинами $\mathbf{p}_{S_i}(P(x[n]=0), P(x[n]=1))$:

$$\begin{aligned} \mathbf{p}_{S_1} = (0,1), \mathbf{p}_{S_2} = (1/2, 1/2), \\ \mathbf{p}_{S_3} = (2/3, 1/3), \mathbf{p}_{S_4} = (3/4, 1/4). \end{aligned} \tag{5.56}$$

Если задать еще и начальное распределение состояний

$$\mathbf{p}_0 = (p_0(1), p_0(2), p_0(3), p_0(4)), \tag{5.57}$$

то все требования из 5.3.5 выполнены и конечный марковский источник определен. Условия (5.55) и (5.56) являются достаточными для построения графа состояний, который изображен на рис. 5.7.

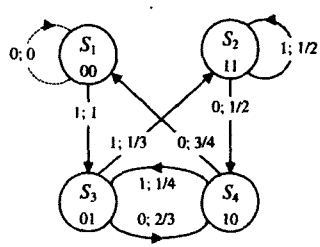


Рис. 5.7. Граф состояний марковского источника с памятью r.

Проанализируем матрицу переходных вероятностей и исследуем ее на регулярность. Матрица переходных вероятностей строится по графу состояний рис. 5.7 и имеет вид

$$\mathbf{\Pi} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 1/3 & 0 & 2/3 \\ 3/4 & 0 & 1/4 & 0 \end{pmatrix}. \tag{5.58}$$

Регулярность проверяется с помощью предельной матрицы. Согласно определению 5.3.4

$$\mathbf{\Pi}_\infty = \frac{1}{41}\begin{pmatrix} 9 & 8 & 12 & 12 \\ 9 & 8 & 12 & 12 \\ 9 & 8 & 12 & 12 \\ 9 & 8 & 12 & 12 \end{pmatrix}. \tag{5.59}$$

Замечание. *Предельная матрица была найдена с помощью программной системы MatLab (http://www.mathworks.com).*

Все строки предельной матрицы равны, следовательно она является регулярной. Соответствующее предельное распределение имеет вид

$$\mathbf{p}_\infty = \frac{1}{41}(9, 8, 12, 12). \tag{5.60}$$

Принципы пошаговой аппроксимации источника с памятью обобщает следующее утверждение.

Теорема 5.3.2. Стационарный марковский источник с памятью r может быть аппроксимирован стационарным марковским источником с памятью l, где $0 \le l < r$.

Если величина r заранее известна, то на первом шаге аппроксимации рассматривается источник без памяти.

Модель источника без памяти полностью описывается распределением вероятностей символов. Средняя вероятность символов – это вероятность, которую оценивает наблюдатель, не зная, в каком состоянии находится источник, поэтому, она определяется стационарным распределением вероятностей состояний $\mathbf{p}_\infty$ и вероятностями символов $a_1, \ldots, a_M$ в состояниях $S_1, \ldots, S_N$

$$(p(a_1), \cdots, p(a_M)) = \mathbf{p}_\infty \cdot \begin{pmatrix} p_{s_1}(a_1) & p_{s_1}(a_2) & \cdots & p_{s_1}(a_M) \\ p_{s_2}(a_1) & p_{s_2}(a_2) & \cdots & p_{s_2}(a_M) \\ \vdots & \vdots & \ddots & \vdots \\ p_{s_N}(a_1) & p_{s_N}(a_2) & \cdots & p_{s_N}(a_M) \end{pmatrix}. \tag{5.61}$$

Пример: Марковский источник с памятью $r = 2$ (продолжение).

- Источник без памяти ($l = 0$). В числовом примере для $a_1 = 0$ и $a_2 = 1$ получаем

$$(p(0), p(1)) = \frac{1}{41}(9, 8, 12, 12) \cdot \begin{pmatrix} 0 & 1 \\ 1/2 & 1/2 \\ 2/3 & 1/3 \\ 3/4 & 1/4 \end{pmatrix} = \frac{1}{41}(21, 20). \qquad (5.62)$$

- Стационый марковский источник с памятью $l = 1$. В этом случае модель источника имеет два состояния. Соответствующий граф состояний изображен на рис. 5.8.

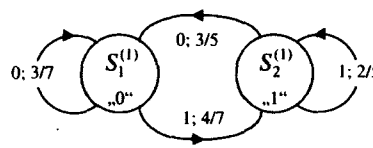


Рис. 5.8. Граф состояний аппроксимирующего марковского источника.

Определим соответствующие графу вероятности. Вероятности состояний равны

$$P(S_1^{(1)}) = p(0) = \frac{21}{41} \text{ и } P(S_2^{(1)}) = p(1) = \frac{20}{41}. \qquad (5.63)$$

Совместные вероятности пар символов можно непосредственно оценить по первоначальному источнику. Они будут равны вероятностям состояний (5.57)

$$\begin{aligned} p(0,0) &= p_\infty(1) = \frac{9}{41} \quad p(1,1) = p_\infty(2) = \frac{8}{41} \\ p(0,1) &= p_\infty(3) = \frac{12}{41} \quad p(1,0) = p_\infty(4) = \frac{12}{41}. \end{aligned} \qquad (5.64)$$

Теперь можно определить переходные вероятности для аппроксимирующего источника. В соответствии с их определением

$$\pi^{(1)}(j/i) = \frac{p(i,j)}{p(i)}, \qquad (5.65)$$

получаем матрицу переходных вероятностей

$$\mathbf{\Pi}^{(1)} = \begin{pmatrix} 3/7 & 4/7 \\ 3/5 & 2/5 \end{pmatrix}. \tag{5.66}$$

Регулярность проверяем путем нахождения предельной матрицы

$$\mathbf{\Pi}_{\infty}^{(1)} = \begin{pmatrix} 0,5122 & 0,4878 \\ 0,5122 & 0,4878 \end{pmatrix}. \tag{5.67}$$

Так как строки предельной матрицы равны, мы имеем регулярную марковскую цепь с предельным распределением

$$\mathbf{p}_{\infty}^{(1)} \approx (0,5122, 0,4878). \tag{5.68}$$

Так как состояния соответствуют символам 0 и 1, должно выполняться

$$\mathbf{p}_{\infty}^{(1)} \approx (21/41, 20/41). \tag{5.69}$$

5.4. Энтропия стационарного марковского источника

Энтропия стационарного марковского источника вычисляется, исходя из того, что каждое состояние источника является подисточником без памяти, обладающим определенной энтропией. Таким образом, энтропия первоначального источника равна математическому ожиданию энтропий подисточников.

Теорема 5.4.1. *Стационарный марковский источник* с алфавитом из M символов, имеющий N состояний, т.е N подисточников, энтропия каждого из которых равна

$$H(X|S_i) = -\sum_{m=1}^{M} p_{s_i}(x_m) \log_2(p_{s_i}(x_m)) \text{ бит}, \tag{5.70}$$

обладает *энтропией*, равной математическому ожиданию энтропий подисточников

$$H_{\infty}(x) = \sum_{i=1}^{N} p_{\infty} H(X/S_i). \tag{5.71}$$

В дальнейшем будет показано, что эвристический подход (5.71) соответствует общему свойству энтропии стационарного дискретного источника в утверждении 5.1.1.

Замечание. *Доказательство утверждения достаточно сложно, поэтому его можно опустить без ущерба для понимания последующих разделов.*

Доказательство. Энтропия марковского источника [10].

Доказательство проводится в три шага.

Шаг 1.

На первом шаге покажем, что при известном состоянии $Z_0 = S_j$ условная энтропия марковского источника определяется как

$$H(X_l|X_{l-1},\ldots,X_0,Z_0=S_j)=\sum_{i=1}^{N}\pi_l(i/j)H(X/Z=S_i), \qquad (5.72)$$

где через X_l обозначен l-ый подисточник, а через Z_l – состояние на шаге l.

Для того, чтобы доказать (5.72), рассмотрим и преобразуем лежащие в основе энтропий условные вероятности $P(x_l/x_{l-1},\ldots,x_0,Z_0=S_j)$. В качестве дальнейшего ограничения введем состояние Z_l

$$P(x_l|x_{l-1},\ldots,x_0,Z_0=S_j)=P(x_l|Z_l,x_{l-1},\ldots,x_0;Z_0=S_j). \qquad (5.73)$$

Так как Z_l полностью определяется начальным состоянием Z_0 и символами $x_0,\ldots,x_{l-1}$, дополнительное ограничение не влияет на условную вероятность, т.е. равенство справедливо. Второе преобразование следует из предполагаемого свойства марковского источника, согласно которому l-ый символ зависит от l-го состояния

$$P(x_l|Z_l,x_{l-1},\ldots,x_0,Z_0=S_j)=P(x_l|Z_l). \qquad (5.74)$$

Для того, чтобы найти условную энтропию (5.72), требуется умножить условные вероятности, лежащие в основе энтропии, на соответствующие вероятности и найти математическое ожидание

$$\frac{H(X_l|X_{l-1},\ldots,X_0,Z_0=S_j)}{\text{бит}}=$$
$$=\sum_{X_0,\ldots,X_l,Z_l}P(x_l,x_{l-1},\ldots,x_0,Z_l|Z_0=S_j)$$
$$\times\log_2 P(x_l|x_{l-1},\ldots,x_0,Z_0=S_j). \qquad (5.75)$$

Величины Z_l не оказывают влияния на совместные вероятности, так как распределения суммы по всем состояниям Z_l является гранич-

ным. Учитывая правую часть (5.74), окончательно получаем

$$\frac{H(X_l|X_{l-1},\ldots,X_0,Z_0=S_j)}{\text{бит}} = \\ = \sum_{X_l,Z_l} P(x_l,Z_l|Z_0=S_j)\log_2 P(x_l|Z_l). \quad (5.76)$$

Вероятность символа x_l на l-ом временном шаге при заданных Z_0 и S_j равна вероятности S_j в S_i при условии, что в состоянии S_i задан символ x_l

$$P(x_l,Z_l|Z_0=S_j) = P(x_l|S_i)\pi_l(i/j), \quad (5.77)$$

поэтому, для (5.76) мы можем написать

$$\sum_{X_l,Z_l} P(x_l,Z_l|Z_0=S_j)\log_2 P(x_l/Z_l) = \\ = \sum_{X_l,S_i} \pi_l(i/j)P(x_l|S_i)\log_2 P(x_l|S_i). \quad (5.78)$$

Сумма по всем символам X_l при заданном состоянии S_i равна энтропии i-го подисточника

$$\sum_{S_i}\pi(i/j)\sum_{X_l}P(x_l|S_i) = \log P(x_l|S_i) = \sum_{i=1}^{N}\pi(i/j)H(X|S_i) \quad (5.79)$$

и утверждение (5.72) доказано.

Шаг 2.

До сих пор мы исходили из фиксированного начального состояния. Для того, чтобы определить энтропию марковского источника через математическое ожидание, будем считать начальное состояние случайным

$$H(X_l|X_{l-1},\ldots,X_0,Z_0) = \sum_{j=1}^{N}\sum_{i=1}^{N} p_{Z_0}(j)\pi_l(i/j)H(X|S_i). \quad (5.80)$$

Сумма по j включает в себя все переходы в i-ое состояние, что дает в результате вероятность i-го состояния на l-ом шаге. Учитывая стационарность источника, получаем

$$\sum_{j=1}^{N} P_{Z_0}(j)\pi_l(i/j) = p_\infty(i). \quad (5.81)$$

Окончательно имеем

$$H(X_l|X_{l-1},\ldots,X_0,Z_0)=\sum_{i=1}^{N}p_\infty(i)H(X|S_i). \tag{5.82}$$

Заметим, что условная энтропия стационарного марковского источника не зависит от числа символов l. Первые части равенств (5.82) и (5.71) одинаковы. Остается доказать равенство левых частей (5.82) и (5.71) при $l\to\infty$.

Шаг 3.

Рассмотрим выражение для энтропии при известном начальном состоянии и преобразуем его с помощью «правила цепочки»

$$\frac{1}{L}H(X_L,\ldots,X_0|Z_0)=\frac{1}{L}[H(X_0|Z_0)+H(X_1|X_0,Z_0)+\cdots+ \\ +H(X_L|X_{L-1},\ldots,X_0,Z_0)]. \tag{5.83}$$

Так как марковский источник стационарен, все слагаемые правой части равны и определяются в соответствии с (5.82). Если имеется L слагаемых, то для любого натурального L получаем

$$\lim_{L\to\infty}\frac{1}{L}H(X_L,\ldots,X_0|Z_0)=\sum_{i=1}^{N}p_\infty(i)H(X|S_i). \tag{5.84}$$

Условие в левой части еще не нарушено. Используя универсальное соотношение

$$H(X)=I(X;Y)+H(X/Y), \tag{5.85}$$

получим

$$H(X_L,\ldots,X_0|Z_0)=H(X_L,\ldots,X_0)-I(X_L,\ldots,X_0;Z_0). \tag{5.86}$$

Из (5.84) следует

$$\lim_{L\to\infty}[H(X_L,\ldots,X_0)-I(X_L,\ldots,X_0;Z_0)]=\sum_{i=1}^{N}p_\infty(i)H(X|S_i). \tag{5.87}$$

С другой стороны, используя универсальное соотношение

$$I(X;Y)=H(Y)-H(Y/X), \tag{5.88}$$

получим

$$I(X_L,\ldots,X_0;Z_0)=H(Z_0)-H(Z_0/X_L,\ldots,X_0). \tag{5.89}$$

Для энтропии имеет место оценка

$$0 \le H(Z_0|X_L,\dots,X_0) \le H(Z_0) \le \log N, \tag{5.90}$$

следовательно

$$0 \le I(X_L,\dots,X_0;Z_0) \le H(Z_0) \le \log N \tag{5.91}$$

и в предельном случае получим

$$\lim_{L\to\infty} \frac{1}{L} I(X_L,\dots,X_0;Z_0) \le 0. \tag{5.92}$$

Используя (5.92), из (5.87) для энтропии стационарного марковского источника окончательно получаем (5.71). ■

В качестве примера рассмотрим опять марковский источник с $r = 2$ и его апроксимации из предыдущих разделов.

1. Источник без памяти.

Прежде всего определим энтропию источника без учета памяти. Из распределения вероятностей двух символов получим энтропию, равную

$$\frac{H^0(X)}{\text{бит}} = -\frac{21}{41}\log_2\frac{21}{41} - \frac{20}{41}\log_2\frac{20}{41} \approx 0,999. \tag{5.93}$$

Энтропия близка к единице, так как символы почти равновероятны.

2. Марковский источник с памятью $r = 1$

Рассмотрим апроксимацию первоначального источника источником с $r = 1$, который состоит из двух подисточников. С учетом вероятностей (см. рис. 5.8), получаем

$$\frac{H_1^{(1)}(X/Z = S_1^{(1)})}{\text{бит}} = -\pi^{(1)}(1/1)\log_2\pi^{(1)}(1/1) - \tag{5.94}$$

$$-\pi^{(1)}(2/1)\log_2\pi^{(1)}(2/1) = -3/7\log_2 3/7 - 4/7\log_2 4/7 \approx 0,985$$

и

$$\frac{H_2^{(1)}(X/Z = S_2^{(1)})}{\text{бит}} = -\pi^{(1)}(1/2)\log_2\pi^{(1)}(1/2) - \tag{5.95}$$

$$-\pi^{(1)}(2/2)\log_2\pi^{(1)}(2/2) = -3/5\log_2 3/5 - 2/5\log_2 2/5 \approx 0,971.$$

Таким образом, энтропия марковского источника с $r = 1$ равна

$$\frac{H^{(1)}(X)}{\text{бит}} = P(S_1^{(1)}H_1^{(1)}(X) + P(S_2^{(1)}))H_2^{(1)}(X) =$$

$$= \frac{21}{41}0,985 + \frac{20}{41}0,971 \approx 0,979. \tag{5.96}$$

По сравнению с (1), энтропия немного уменьшилась.

3. Марковский источник с памятью $r = 2$.

В этом случае мы должны принимать во внимание 4 подисточника (см. рис. 5.7). Для состояния S_1 имеем

$$H_1(X/Z = S_1) = 0 \text{ бит}. \tag{5.97}$$

Так как подисточник 2 постоянно вырабатывает символ «1», энтропия в состоянии S_2 равна

$$H_2(X/Z = S_2) = 1 \text{ бит}. \tag{5.98}$$

Подисточники 3 и 4 обладают энтропией соответственно

$$\frac{H_3(X/Z = S_3)}{\text{бит}} = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} \approx 0,918 \tag{5.99}$$

и

$$\frac{H_4(X/Z = S_4)}{\text{бит}} = -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} \approx 0,811. \tag{5.100}$$

В результате получаем энтропию источника с $r = 2$, равную

$$\begin{aligned} H_\infty(X) &= \sum_{i=1}^{4} P(S)_i H_i(X) = \\ &= \frac{1}{41}(8 + 12 \cdot 0,918 + 12 \cdot 0,811) \text{ бит} \approx 0,701 \text{ бит}. \end{aligned} \tag{5.101}$$

Итак, мы видим, что по сравнению с марковским источником с памятью $r = 1$, энтропия уменьшилась.

5.5. Кодирование стационарных марковских источников

Прежде всего напомним теорему кодирования источников, сформулированную в теореме 5.2.1. В этой теореме рассматриваются блоки, содержащие L символов. Теорема утверждает, что в случае, когда L стремится к бесконечности, для блоков длины L существует префиксный код, в котором средняя длина кодового слова на один символ как угодно близка к совместной энтропии $H_L(X)$.

В случае марковского источника с памятью r, совместная энтропия для блока длины $L = r$ является предельным случаем и теорема кодирования может быть сформулированна следующим образом:

Теорема 5.5.1. *Теорема кодирования* стационарных марковских источников с памятью r и энтропией $H_\infty(x)$.

Для блока длины $L \geq r$ существует D-ичный префиксный код, в котором средняя длина кодового слова $\overline{n}$ удовлетворяет неравенству

$$\frac{H_\infty(x)}{\log D} \leq \overline{n} \leq \frac{H_\infty(x)}{\log D} + \frac{1}{L}. \tag{5.102}$$

Из разложения марковского источника на подисточники без памяти непосредственно вытекает стратегия оптимального кодирования.

Если начальные состояния известны, то при кодировании источников все последующие состояния однозначно определены. При этом в передатчике и приемнике для каждого множества подисточников возможно провести кодирование и декодирование Хаффмана, учитывая распределение вероятностей символов и состояний.

Практическая реализация кодов Хаффмана показывает, что для достижения существенного кодового выигрыша, необходимо кодировать блоки достаточно большой длины. Кроме этого, базовые состояния всегда должны определяться r символами для того, чтобы переход к блокам большей длины был относительно несложным.

Предлагаемая простая стратегия полностью учитывает память источника и, следовательно, в предельном случае, позволяет получить оптимальный префиксный код.

Кодирование стационарного марковского источника X с памятью r и энтропией, равной $H_\infty(X)$.

1. Объединить в блоки $l = r + 1$ символов источника.
2. Провести *кодирование Хаффмана* для блоков.
3. Если средняя длина кодового слова на символ существенно отличается от энтропии $H_\infty(X)$, то увеличить длину блока за счет последующих символов. Провести кодирование Хаффмана для блоков большей длины. Продолжить этот процесс до удовлетворительного приближения средней длины кодового слова к энтропии.

Пример: Кодирование марковского источника с памятью $r = 2$ (продолжение).

Проверим эффективность предложенного алгоритма на численном примере из предыдущего раздела.

В соответствии с памятью источника $r = 2$, объединим в блоки каждые три символа источника и проведем кодирование Хаффмана.

Необходимые вероятности состояний для блоков определяются стационарным распределением вероятностей (5.58) и матрицей переходных вероятностей (5.58) или графом состояний (рис. 5.7). Вероятность блока 001, например, равна

$$P_{001} = P(S_1)\pi(3/1) = p_\infty(1)\pi(3/1) = 9/41 \cdot 1 = \frac{9}{41}. \qquad (5.103)$$

Можно так же заметить, что блок 000 никогда не появляется и, следовательно, не должен кодироваться.

Из таблицы 5.1. находим, что средняя длина кодового слова на символ равна 0,911. Поэтому эффективность кодирования относительно мала

$$\eta = \frac{H_\infty(x)}{\overline{n}} = \frac{0,701}{0,911} \approx 0,77. \qquad (5.104)$$

Путем построения блоков большей длины, эффективность кодирования может быть существенно повышена. Повторим кодирование Хаффмана для блоков, длина которых равна 4 (табл. 5.2). Из таблицы 5.1. следует, что средняя длина кодового слова на символ равна 0,744. Эффективность кодирования уже равна

$$\eta = \frac{H_\infty(x)}{\overline{n}} \approx \frac{0,701}{0,744} \approx 0,94 \qquad (5.105)$$

и мы, в основном, реализовали большую часть возможностей кодирования, однако, дальнейшим увеличением длины блока можно увеличить выигрыш от кодирования.

Пример: Марковский источник первого порядка.

На рис. 5.9 задан граф состояний марковского источника первого порядка.

1. Дополните недостающие вероятности рис. 5.9 и найдите стационарное распределение вероятностей состояний.

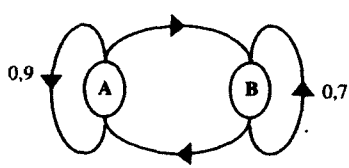


Рис. 5.9. Граф состояний марковского источника первого порядка.

Таблица 5.1. Кодирование Хаффмана для марковского источника с памятью $r = 2$ и длиной блока 3.

Символы	p_i	Кодовая конструкция	Кодовое слово	n_i	$n_i \cdot p_i$
001	9/41		00	2	18/41
100	9/41		10	2	18/41
010	8/41		010	3	24/41
110	4/41		110	3	12/41
111	4/41		111	3	12/41
011	4/41		0110	4	16/41
101	3/41		0111	4	12/41
000	0		-	-	-

$$\frac{\bar{n}}{\text{бит}} = \frac{1}{3} \cdot \frac{112}{41} \approx 0{,}911$$

2. Найдите энтропию источника.

3. Проведите кодирование Хаффмана для блоков, состоящих из трех двоичных символов.

4. Какой эффективностью обладает кодирование?

Решение.

1. Переходные вероятности рис. 5.9 равны

$$\begin{aligned} \pi(A/A) &= 0{,}9 \quad \pi(B/A) = 0{,}1 \\ \pi(A/B) &= 0{,}3 \quad \pi(B/B) = 0{,}7. \end{aligned} \tag{5.106}$$

Стационарное значение вероятности состояния A определяется, исходя из следующих равенств

$$\begin{aligned} P(A) &= 1 - P(B) \\ P(A) &= 0{,}9 \cdot P(A) + 0{,}3 \cdot P(B), \end{aligned} \tag{5.107}$$

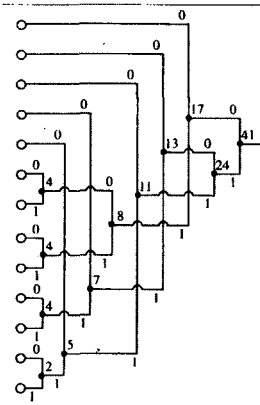
поэтому,

$$\begin{aligned} P(A) &= 0{,}9 \cdot P(A) + 0{,}3 \cdot (1 - P(A)) \\ P(A) &\cdot [1 - 0{,}9 + 0{,}3] = 0{,}3 \\ P(A) &= 3/4. \end{aligned} \tag{5.108}$$

Из этого следует, что

$$P(B) = 1/4. \tag{5.109}$$

Таблица 5.2. Кодирование Хаффмана для марковского источника с памятью $r = 2$ и длиной блока 4.

Символы	P_i	Кодовая конструкция	Кодовое слово	n_i	$n_i P_i$
1001	9/41		00	2	18/41
0010	6/41		100	3	18/41
0100	6/41		110	3	18/41
0011	3/41		1010	4	12/41
1100	3/41		1110	4	12/41
0101	2/41		0100	4	8/41
1110	2/41		0101	4	8/41
1111	2/41		0110	4	8/41
0110	2/41		0111	4	8/41
0111	2/41		10110	5	10/41
1010	2/41		10111	5	10/41
1101	1/41		11110	5	5/41
1011	1/41		11111	5	5/41
1000	0		-	-	-
0000	0		-	-	-
0001	0		-	-	-
			$\frac{\bar{n}}{\text{бит}} = \frac{1}{4} \cdot \frac{122}{41} \approx 0{,}744$		

2. Энтропия источника определяется как

$$\begin{aligned} \frac{H(A)}{\text{бит}} &= -0,9\log_2 0,9 - 0,1\log_2 0,1 = 0,469 \\ \frac{H(B)}{\text{бит}} &= -0,7\log_2 0,7 - 0,3\log_2 0,3 = 0,811 \\ H_\infty(X) &= P(A)\cdot H(A) + P(B)\cdot H(B) = 0,572 \text{ бит.} \end{aligned} \tag{5.110}$$

3. Вероятности блоков, состоящих из трех двоичных символов, равны

$$\begin{aligned} P(AAA) &= P(A)\cdot\pi(A/A)\cdot\pi(A/A) = 0,6075 \\ P(AAB) &= P(A)\cdot\pi(A/A)\cdot\pi(B/A) = 0,0675 \\ P(ABA) &= P(A)\cdot\pi(B/A)\cdot\pi(A/B) = 0,0225 \\ P(ABB) &= P(A)\cdot\pi(B/A)\cdot\pi(B/B) = 0,0525 \\ P(BAA) &= P(B)\cdot\pi(A/B)\cdot\pi(A/A) = 0,1225 \\ P(BAB) &= P(B)\cdot\pi(A/B)\cdot\pi(B/A) = 0,0525 \\ P(BBA) &= P(B)\cdot\pi(B/B)\cdot\pi(A/B) = 0,0075 \\ P(BBB) &= P(B)\cdot\pi(B/B)\cdot\pi(B/B) = 0,0675. \end{aligned} \tag{5.111}$$

Кодирование Хаффмана представлено на рис. 5.10.

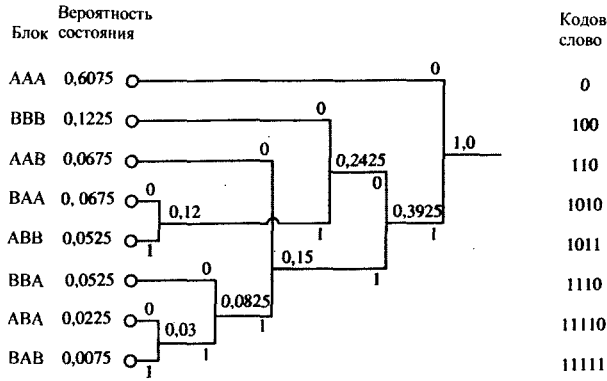


Рис. 5.10. Кодирование Хаффмана.

4. Средняя длина кодового слова определяется как

$$\begin{aligned}\overline{n} = 1/3(0,6075 + 3(0,1225 + 0,0675)+ \\ + 4(0,0675 + 0,0525 + 0,0525)+ \\ + 5(0,0225 + 0,0075)) = 0,6725,\end{aligned} \tag{5.112}$$

следовательно, эффективность кодирования равна

$$\eta = \frac{H_\infty(x)}{\overline{n}} = \frac{0,572}{0,6725} \approx 0,85. \tag{5.113}$$

5.6. Выводы

В приведенных ниже таблицах читатель может найти важнейшие утверждения и связь марковских цепей и марковских источников.

Таблица 5.3. Марковские цепи.

- Марковским процессом называется стохастический процесс, в котором настоящее известно, а будущее не зависит от прошлого.

- Дискретный по времени и состояниям марковский процесс называется марковской цепью. Его реализацией является последовательность состояний

$$S_i \in \mathbf{S} = \{S_1, S_2, \ldots, S_N\}.$$

- Цепь Маркова является гомогенной, если переходные вероятности между состояниями не зависят от выбора временной точки отсчета, следовательно,

$$\pi(j/i) = P(S_j/S_i) \text{ для } j, i = 0, 1, 2, 3, \ldots, N.$$

- Гомогенная марковская цепь полностью определяется матрицей переходных вероятностей $\mathbf{\Pi}$

$$\mathbf{\Pi} = (\pi(j/i))_{N\times N} = \begin{pmatrix} \pi(1/1) & \pi(2/1) & \cdots & \pi(N/1) \\ \pi(1/2) & \pi(2/2) & \cdots & \pi(N/2) \\ \vdots & \vdots & \ddots & \vdots \\ \pi(1/N) & \pi(2/N) & \cdots & \pi(N/N) \end{pmatrix}$$

и начальным распределением

$$\mathbf{p}_0 = (p_0(S_1), p_0(S_2), \ldots, p_0(S_N)).$$

- Эквивалентом матрицы переходных вероятностей является граф состояний с узлами, путями и весом путей, соответствующим состояниям, переходам между состояниями и переходным вероятностям.

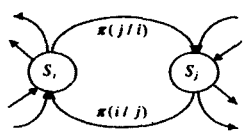


Рис. 5.11.

- Распределение состояний гомогенной марковской цепи на n -ом шаге определяется как

$$\mathbf{p}_n = \mathbf{p}_0\mathbf{\Pi}^n.$$

Таблица 5.4. Марковские цепи. (продолжение)

- Гомогенная цепь Маркова стационарна, если распределение состояний постоянно во времени. В этом случае начальное распределение является собственным вектором переходной матрицы, т.е

$$\mathbf{p}_0 = \mathbf{p}_0\mathbf{\Pi}.$$

- Гомогенная цепь Маркова регулярна, если существует предел

$$\mathbf{\Pi}_\infty = \lim_{n\to\infty} \mathbf{\Pi}^n,$$

причем, все строки $\mathbf{\Pi}_\infty$ равны предельному распределению $\mathbf{p}_\infty$.

- Предельное распределение $\mathbf{p}_\infty$ является единственным стационарным распределением регулярной цепи Маркова.

Таблица 5.5. Стационарные Марковские источники.

Конечный дискретный марковский источник с памятью r полностью определяется следующими условиями:

- Задано непустое множество состояний $\mathbf{S} = \{S_1, S_2, \ldots, S_N\}$, причем, S содержит все векторы длины r;
- Каждое состояние $s_i \in \mathbf{S}$ соответствует дискретному источнику без памяти с алфавитом $X_i = \{x_1, x_2, \ldots, x_M\}$ и вероятностями j-ых символов алфавита $p^{(i)}(j)$;
- Состояние $S[n] = (x[n-r], x[n-r+1], \ldots, x[n-1])$ из r последовательных символов и очередной символ $x[n]$ образуют новое состояние $s[n+1] = (x[n-r+1], x[n-r+2], \ldots, x[n])$;
- Задано начальное распределение состояний $\mathbf{p}_0 = (p_0(1), p_0(2), \ldots, p_0(N))$.

Энтропия стационарного марковского источника с памятью r определяется как математическое ожидание условных энтропий подисточников

$$H_\infty(X) = \sum_{i=1}^{N} p_\infty(i) H(X/S_i),$$

при этом условная энтропия i-ого подисточника, соответствующего состоянию i равна

$$H(X|S_i) = -\sum_{m=1}^{M} p_{S_i}(x_m) \log_2(P_{S_i}(x_m)) \text{ бит.}$$

ГЛАВА 6

СЖАТИЕ ДАННЫХ

6.1. Введение

Задачей *сжатия данных* является минимизация технических затрат на хранение или передачу информации путем оптимального кодирования источников. При этом различают два понятия:

- *Несущественная информация* – это информация, которой можно пренебречь при передаче. Примером может служить традиционная телефония. В телефонных каналах передача информации осуществляется в полосе 3,4 кГц. Все остальные спектральные составляющие отбрасываются, при этом существенная часть передаваемой информации теряется. Ясно, что первоначальный речевой сигнал не может быть полностью восстановлен на приемном конце. В этом случае говорят о кодировании с потерями.

- Под *избыточностью* понимают неоднократное повторение в сообщении необходимой для приемника информации. Избыточность может быть устранена без потери информации. Примером является кодирование Хаффмана. Такое кодирование называют кодированием без потерь.

Важнейшими примерами сжатия данных являются цифровое радиовещание (Digital Audio Broadcasting, DAB) и цифровое телевидение (Digital Video Broadcasting, DVB)[19]. Обе системы работают на основе кодирования аудио и видео сигналов, использующего стандарты MPEG (Motion Pictures Experts Group). Кодирование аудиосигнала основано на психологической модели восприятия речи, которая использует скрытые спектральные и временные эффекты, при этом в сигнальном блоке происходит удаление невоспринимаемой на слух части аудиосигнала (несущественной информации). Аналогичные эффекты используются также при кодировании видеосигнала (в частности, психологический эффект движения). Кодирование изображений позволяет достичь еще большей степени сжатия.

Степень сжатия определяется затратами для передачи или хранения информации без сжатия k_0 и затратами с использованием некоторого метода сжатия k_m

$$G_k = \frac{k_0 - k_m}{k_0}. \tag{6.1}$$

Степень сжатия зависит от используемого алгоритма и свойств источника. Приведем некоторые численные примеры степеней сжатия, достигаемых на практике :

-до 80 % для текстовых данных (в формате редактора Word 97 с помощью программы сжатия ZIP);

-87,5 % при переходе от PCM-телефонии со скоростью 64 кбит/сек к передаче информации по рекомендации ITU G.725 со скоростью 8 кбит/сек;

-90 % при кодировании информации стереофоничских аудио компакт дисков со скоростью $2 \cdot 16$ бит $\cdot 44$ кГц $= 1408$ кбит/сек методом, использующим стандарт сжатия MPEG (Advanced Audio Coding) со скоростью 112 кбит/сек и почти равнозначным качеством речи.

Следующим примером является энтропия немецкого литературного текста. Результаты частотного анализа представлены на рис. 6.1. Если рассматривать буквы изолированно, то получим энтропию, приблизительно равную 4,7 бит/букву. Объединяя буквы в блоки, мы используем уже такие очевидные связи, как слоги, слова и т.д., поэтому, для блоков очень большой длины асимптотически достижимая граница равна 1,6 бит/букву.

Рис. 6.1. Энтропия немецкого литературного языка как функция длины блока

Алгоритмы сжатия данных можно разделить на три группы:

1. Статические алгоритмы, например, *кодирование Хаффмана.*

   Сжатие немецкого литературного текста методом Хаффмана, по сравнению с сжатием информации, состоящей из произволь-

ных стандартных символов ASCII, позволяет достичь приблизительно 50 % выигрыша.

2. *Адаптивные алгоритмы*, например, модифицированное кодирование Хаффмана. Здесь распределение вероятностей символов вначале полагается равномерным, а потом меняется во времени по мере накопления статистики.

3. *Динамические алгоритмы*, например кодирование, используемое в рекомендации ITU V42. bis.

Основная проблема *энтропийного кодирования* заключается в том, что оно предполагает знание распределения вероятностей символов. Очень часто статистика символов заранее неизвестна и эффективному кодированию должен предшествовать частотный анализ. Здесь на помощь приходят *универсальные алгоритмы.*

- Универсальные алгоритмы сжатия, являющиеся по своей сути адаптивными, не нуждаются в априорной статистике. Такое эффективное кодирование начинается сразу же после поступления информации на вход кодера.

- Кроме этого, существуют «быстрые» алгоритмы с относительно простой технической сложностью.

- Каждый из предполагаемых алгоритмов помогает достичь высокой степени сжатия.

В качестве примера методов сжатия, рассмотрим два важнейших алгоритма: арифметическое кодирование, при котором производится динамический частотный анализ и универсальный алгоритм Лемпеля-Зива. Алгоритм Ле́мпеля-Зива LZ77 был предложен в 1977 году и модифицирован в 1984 г. Он используется в рекомендации ITU V.42.bis и называется LZW алгоритмом.

6.2. Арифметическое кодирование

При *арифметическом кодировании* мы исходим из того факта, что при нормализованном распределении сумма вероятностей символов (и соответствующих им относительных частот) источника всегда равна единице. Если относительные частоты символов неизвестны передатчику и приемнику:

Таблица 6.1. Буквы и их относительные частоты.

Буква	E	S	G	L	R
Относительная частота	0,5	0,2	0,1	0,1	0,1

– они могут определяться, например, путем текущих статистических изменений передаваемой информации в фиксированные моменты времени;

– приемник и передатчик совместно, исходя из относительных частот, устанавливают жесткие правила кодирования.

Особенностью арифметического кодирования является то, что для отображения последовательности символов в потоки натуральных чисел на интервале [0,1] используются относительные частоты.

Результатом такого отображения является сжатие символов (посимвольное сжатие) в соответствии с их вероятностями. Поясним идею арифметического кодирования с помощью следующего примера.

Рассмотрим арифметическое кодирование последовательности букв «GELEEESSER». Относительные частоты букв в этом потоке приведены в таблице 6.1.

Процедура кодирования представлена на рис. 6.2.

Первой букве «G», в соответствии с ее относительной частотой, соответствует интервал $[0.7, 0.8[$. Согласно алгоритму, каждая цепочка букв, начинающихся с G, всегда будет отображаться в число, принадлежащее этому интервалу. Таким образом, в рассматриваемом примере первая десятичная цифра после запятой уже определена.

Кодирование последующих букв производится аналогично с тем важным отличием, что теперь делению каждый раз будет подвергаться интервал, выбранный на предыдущем шаге. Из рис. 6.2 следует, что букве «E» на втором шаге соответствует интервал $[0.7, 0.75[$.

Таблица 6.3, в которой алгоритм кодирования прослежен по шагам, показывает, что последовательность «GELEEESER» отображается в число 740387 (0 и запятая не нуждаются в отображении). Отметим, что:

1. Часто встречающимся буквам ставятся в соответствие большие интервалы. На их отображение затрачивается меньше десятичных цифр, чем на отображение редко встречающихся букв.

2. Длинные сообщения отображаются в «длинные» числа. Представление этих чисел в двоичной форме, необходимое для передачи сообщений, приводит к появлению кодовых слов большой длины.

Практическая реализация скользящего алгоритма арифметического кодирования требует высокой точности, которая ограничивается длиной кодовых слов. Для сокращения необходимой длины регистра, при реализации арифметического кодирования используется целочисленная арифметика с выдачей уже готовых промежуточных результатов.

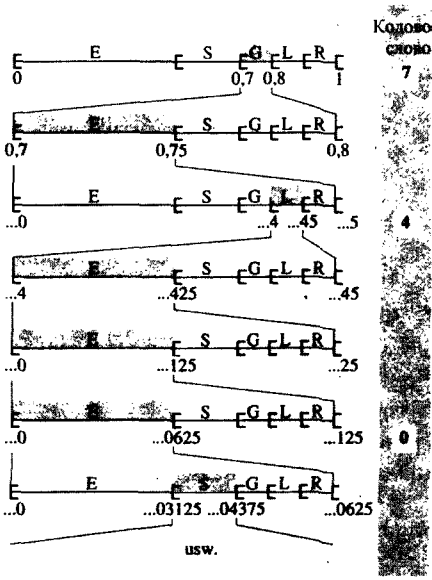


Рис. 6.2. Арифметическое кодирование.

Алгоритм представлен в таблице 6.2. Механизм его действия раскрывается с помощью таблицы 6.3. В нашем примере для реализации кодирования достаточно регистра для хранения шести текущих десятичных цифр.

В соответствии с таблицей 6.2, на первом шаге происходит инициализация переменных LO и HI. Для первой кодируемой буквы «G» ширина интервала равна $B = 1'000'000$. Нижняя и верхняя границы равны соответственно $LO = 0 + 1'000'000 \cdot 0,7 = 700'000$ и $HI = 0 + 1'000'000 \cdot 0,8 - 1 = 799'999$. Первая десятичная цифра уже определена и равна 7, а содержимое регистров LO и HI сдвигается на одну позицию влево. Заметим, что в регистре LO освободившееся место занимает 0, а в регистре HI – 9.

Таблица 6.2. Алгоритм арифметического кодирования.

Начальные значения					
Регистр	LO=000'000			HI=999'999	
Символ	E	S	G	L	R
LS(·)	0	0,5	0,7	0,8	0,9
HS(·)	0,5	0,7	0,8	0,9	1

Алгоритм	
Ширина интервала	B=HI-LO+1
Нижняя граница	LO=LO+B·LS
Верхняя граница	HI=LO+B·HS-1

Для второй буквы «E» ширина интервала равна $B = 1'000'000$, поэтому, получаем $LO = 0 + 1'000'000 = 000'000$ и $HI = 0 + 1'000'000 \cdot 0,5 - 1 = 499'999$. Кодирование последующих букв проводится аналогично. В завершении работы алгоритма выдается некоторое число из последнего интервала. Для этого мы округляем верхнюю границу интервала HI, отбрасывая младшие разряды до тех пор, пока это возможно. В результате получаем число с минимальным количеством цифр, принадлежащее последнему интервалу.

6.3. Кодирование Лемпеля – Зива

Алгоритм кодирования Лемпеля – Зива LZ77 основан на принципе динамических словарей. Мы представим вкратце эту концепцию и наглядно поясним ее на простейших примерах [19].

В основе алгоритма лежат четыре основные идеи:

1. Каждая очередная закодированная последовательность символов добавляется к ранее закодированным символам таким образом, что вместе с ними она образует разложение всей переданной и принятой информации на несовпадающие между собой фразы (Парсинг).

2. Такое разложение хранится в памяти и используется в дальнейшем в качестве словаря.

3. Кодирование осуществляется при помощи указателей на фразы из уже сформированного словаря фраз.

Таблица 6.3. Арифметическое кодирование фразы «GELEEESSER».

Символ	LO	HI	Выход	Символ	LO	HI	Выход
Старт	000'000	999'999		E	000'000	062'499	
G	700'000	799'999		Выход/сдвиг	000'000	624'999	**0**
Выход/сдвиг	000'000	999'999	**7**	S	312'500	437'499	
E	000'000	499'999		S	375'000	399'999	
L	400'000	449'999		Выход/сдвиг	750'000	999'999	**3**
Выход/сдвиг	000'000	499'999	**4**	E	750'000	874'999	
E	000'000	249'999		R	862'500	874'999	**87**
E	000'000	124'999		Конец			

4. Кодирование является динамической процедурой, ориентированной на блоки. Сам процесс кодирования может быть дополнен скользящими окнами, содержащими текущий словарь фраз и Look-ahead буфером.

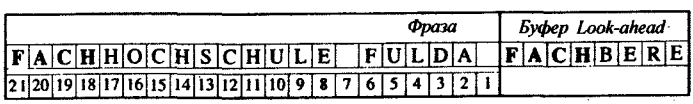


Рис. 6.3. Скользящее окно алгоритма LZ77, соответствующее фразе FACH.

В процессе кодирования обрабатываемый текст отображается в последовательность указателей или флагов. Структура закодированного текста показана на рис. 6.4. В примере, показанном на рис. 6.3, цепочка букв «FACH» заменяется последовательностью $[21, 4, B]$.

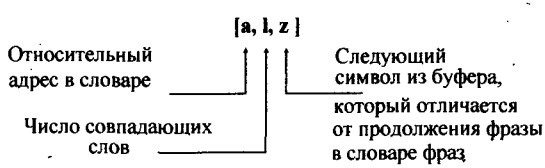


Рис. 6.4. Структура указателей.

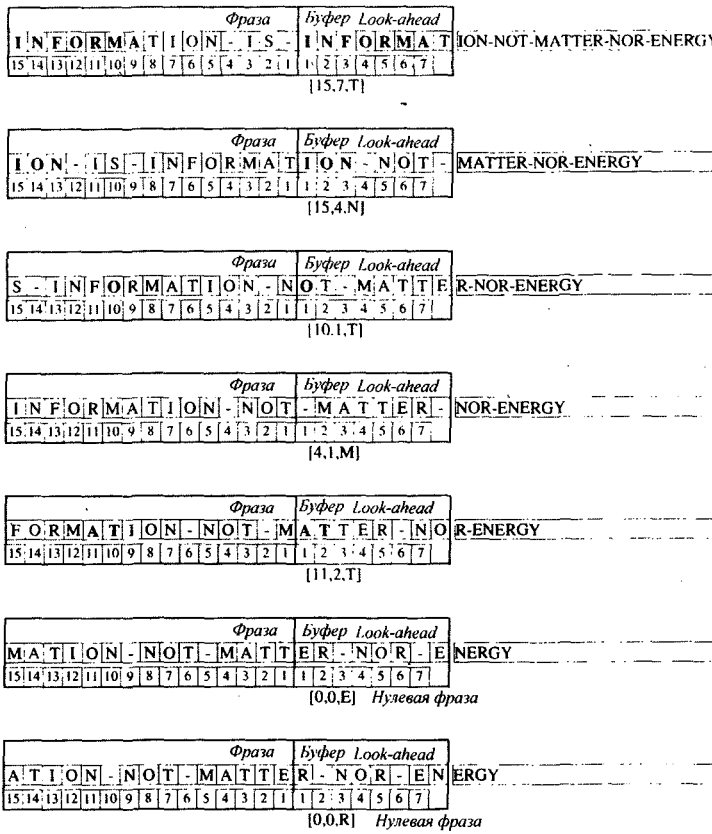


Рис. 6.5. Пример кодирования по алгоритму LZ77 с нулевой фразой и повторением символов.

На рис. 6.5 представлен некоторый частный случай и приведено его алгоритмическое решение. На шестом шаге кодирования очередным является символ «E», не содержащийся в словаре фраз. В этом случае он кодируется посредством, так называемой, «нулевой фразы». «Нулевая фраза» имеет вид [0, 0, символ] и при декодировании распознается по двум стоящим рядом нулям.

Еще одним интересным случаем является повторение символов, так называемое Character Rans. В этом случае, для замены символа используются уже два флага. Первый «0» служит признаком по-

явления некоторой особенности текста. Последующие «1» и символ указывают на то, что символ повторяется. Во втором флаге указывается число повторений и последующий символ.

Затраты на кодирование определяются длиной окна, содержащего словарь фраз ω_p, длиной Look ahead буфера ω_L и затратами на двоичное представление указателя

$$\frac{K_E}{\text{бит}} = \log_2 \omega_p + \log_2 \omega_L + 8. \tag{6.2}$$

Кодирование Лемпеля – Зива приводит к сжатию данных в том случае, если затраты на кодирование, т.е. длина указателя в двоичном исчислении в среднем оказывается меньше, чем при непосредственном кодировании, например, кодом ASCII, что соответствует 8 битам на один символ.

В типичном случае $\omega_p = 2^{12} = 4096$ и $\omega_L = 2^4 = 16$ и затраты на двоичное представление указателя составляют 24 бита. Для фразы, состоящей из четырех букв, которая уже содержится в словаре фраз, экономия, по сравнению с прямым кодированием кодом ASCII (32 бита), составляет 25 %.

Для кодирования Лемпеля – Зива установлено, что:

- Часто появляющиеся цепочки символов кодируются очень эффективно;
- Редко появляющиеся символы и последовательности символов с течением времени удаляются из словаря фраз;
- Повторяющиеся символы также кодируются эффективно;
- На кодирование нулевых фраз затрачивается относительно большое число бит;
- Методы теории информации позволяют доказать, что кодирование методом Лемпеля – Зива асимптотически оптимально. Это означает, что для очень длинного текста избыточность исчезает, то есть среднее число бит, необходимое для кодирования одного символа, стремится к энтропии текста;
- Практически достижимая степень сжатия для длинных текстов составляет 50 – 60%.

ГЛАВА 7

ДИСКРЕТНЫЕ КАНАЛЫ БЕЗ ПАМЯТИ И ПЕРЕДАЧА ИНФОРМАЦИИ

7.1. Введение

В главе 4 были рассмотрены два связанных источника информации. Были введены такие ключевые понятия как совместные, взаимные и условные информации пар событий (символов) для связанных источников. На их основе мы пришли к фундаментальным понятиям информации – совместной, взаимной и условной энтропии. (См. табл. 4.3). Там же было отмечено, что совместная и условная энтропии имеют аналоги в теории вероятностей и определяются как математические ожидания совместных и условных информаций всех пар событий двух источников.

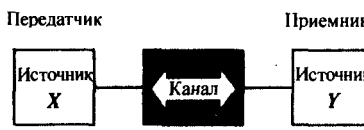


Рис. 7.1. Модель передачи.

Мы продолжим эти рассуждения, уделив основное внимание взаимной энтропии. Для описания каналов передачи информации используем концепцию двух связанных источников. Оказывается, что с помощью понятий, введенных в главе 4, удается полностью описать процесс передачи информации по каналам без памяти. В результате, мы оценим возможность передачи информации по каналу, т.е. пропускную способность канала.

В Шенноновской модели канала связи информация одного источника (передатчика) передается по каналу приемнику и выдается потребителю. Для потребителя имеет значение только выход приемника, т.е. приемник сам является источником информации, поэтому,

модель связанных источников полностью применима к цепочке Передатчик – Канал – Приемник (рис. 7.1). Если происходит передача информации, то символы одного источника должны оказывать влияние на символы другого источника. В качестве примера рассмотрим двоичные симметричные каналы без памяти.

7.2. Двоичный симметричный канал

Двоичный симметричный канал (ДСК) является простейшим примером взаимодействия двух дискретных источников без памяти. Он является дискретной двоичной моделью передачи информации по каналу с аддитивным белым гауссовским шумом (АБГШ).

Замечание. *При проверке эффективности алгоритмов помехоустойчивого кодирования, для расчетов и моделирования каналов связи методом Монте-Карло успешно применяются дискретные модели каналов.*

Двоичный симметричный канал описывается с помощью диаграммы переходов (рис. 7.2). На диаграмме представлены возможные переходы двоичных символов $(0,1)$ источника X в двоичные символы источника Y. Каждому переходу приписана переходная вероятность.

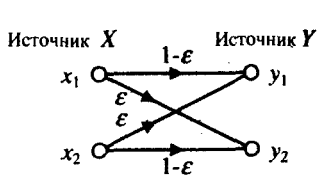


Рис. 7.2. Диаграмма передачи данных по двоичному симметричному каналу.

Из рис. 7.2 видно, что ошибочным переходам соответствует вероятность ε, поэтому, обычно говорят, что при передаче двоичной информации по ДСК, ошибка происходит с вероятностью ε. Эквивалентом диаграммы переходов является матрица канала. Она содержит переходные вероятности и является стохастической матрицей, у которой сумма всех элементов каждой строки равна единице.

Матрица канала с входным алфавитом, состоящим из M символов x_i и выходным алфавитом, состоящим из N символов y_j, содер-

жит все переходные вероятности $P(y_j/x_i)$ и имеет вид

$$\mathbf{P}_{Y/X} = \begin{pmatrix} p(y_1/x_1) & p(y_2/x_1) & \cdots & p(y_N/x_1) \\ p(y_1/x_2) & p(y_2/x_2) & \cdots & p(y_N/x_2) \\ \vdots & \vdots & \ddots & \vdots \\ p(y_1/x_M) & p(y_2/x_M) & \cdots & p(y_N/x_M) \end{pmatrix}. \tag{7.1}$$

В случае ДСК имеем

$$\mathbf{P}^{\text{ДСК}}_{Y/X} = \begin{pmatrix} 1-\varepsilon & \varepsilon \\ \varepsilon & 1-\varepsilon \end{pmatrix}. \tag{7.2}$$

Из симметрии переходов следует, что равномерное распределение символов на входе канала влечет за собой равномерное распределение выходных символов.

Выпишем условные и взаимные информации всех возможных пар событий, предполагая равномерное распределение входных символов. Для ДСК имеем

$$\begin{aligned} I(y_1/x_1) &= -\log_2 p(y_1/x_1) \text{ бит} = -\log_2(1-\varepsilon) \text{ бит} \\ I(y_2/x_1) &= -\log_2 p(y_2/x_1) \text{ бит} = -\log_2 \varepsilon \text{ бит} \\ I(y_1/x_2) &= -\log_2 p(y_1/x_2) \text{ бит} = -\log_2 \varepsilon \text{ бит} \\ I(y_2/x_2) &= -\log_2 p(y_2/x_2) \text{ бит} = -\log_2(1-\varepsilon) \text{ бит}. \end{aligned} \tag{7.3}$$

Отсюда следует

$$\begin{aligned} I(x_1;y_1) &= \log_2 \frac{p(y_1/x_1)}{p(y_1)} \text{ бит} = \log_2 \frac{(1-\varepsilon)}{1/2} = (1+\log_2(1-\varepsilon))\text{бит} \\ I(x_1;y_2) &= \log_2 \frac{p(y_2/x_1)}{p(y_1)} \text{ бит} = \log_2 \frac{\varepsilon}{1/2} = (1+\log_2 \varepsilon) \text{ бит} \\ I(x_2;y_1) &= \log_2 \frac{p(y_1/x_2)}{p(y_1)} \text{ бит} = \log_2 \frac{\varepsilon}{1/2} = (1+\log_2 \varepsilon) \text{ бит} \\ I(x_2;y_2) &= \log_2 \frac{p(y_2/x_2)}{p(y_1)} \text{ бит} = \log_2 \frac{(1-\varepsilon)}{1/2} = (1+\log_2(1-\varepsilon))\text{бит}. \end{aligned} \tag{7.4}$$

Рассмотрим три особых случая

1. $\varepsilon = 0$ (передача без ошибок)

   $I(x_1;y_1) = I(x_2;y_2) = 1$ бит.

   Других взаимных информаций не существует, так как пары взаимных символов (x_1, y_2) и (x_2, y_1) никогда не могут появится. Информация передается от источника X к источнику Y без потерь.

2. $\varepsilon = 1/2$. Для всех пар символов (x_i, y_j) имеем

$$I(x_i; y_j) = \log_2 \frac{1/2}{1/2} \text{ бит} = 0.$$

Источники X и Y независимы. Передачи информации не происходит.

3. $\varepsilon = 1$. В этом случае или какие-то вероятности перепутаны, или мы где-то полностью заблуждаемся. Обнаружив этот факт и проинвертировав принятые символы y_i, мы придем к первому случаю.

В заключении рассмотрим поведение условной $I(y_i/x_i) = I(y_1/x_1)$ и взаимной $I(y_i; x_i) = I(y_1; x_1)$ информаций в ДСК как функций вероятности ошибки ε.

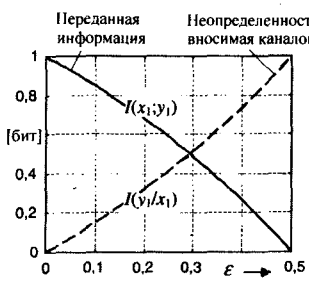


Рис. 7.3. Условная $I(y_1/x_1)$ и взаимная информация $I(y_1; x_1)$.

Условную информацию $I(y_i/x_i)$ можно рассматривать как неопределенность, вносимую каналом, а взаимную информацию $I(y_i; x_i)$ как информацию, передаваемую по каналу. При передаче одного двоичного символа $\varepsilon = 0$, информация передается без потерь, поэтому $I(y_i/x_i) = 0$, а $I(y_i; x_i) = 1$ бит. С ростом вероятности ошибки ε, неопределенность, вносимая каналом, возрастает, а передаваемая информация, наоборот, убывает. При $\varepsilon = 0,5$, передача информации отсутствует, поэтому $I(y_i/x_i) = 1$, а $I(y_i; x_i) = 0$. Сумма же условной $I(y_i/x_i)$ и взаимной $I(y_i; x_i)$ информации не зависит от ε и всегда равна одному биту.

7.3. Передача информации

После рассмотрения отдельных пар событий в предыдущем разделе, вернемся опять к модели передачи информации. На рис. 7.4 показана исходная ситуация.

Описание канала с помощью переходных вероятностей сводится, в конечном счете, к совместным вероятностям пар событий. С этой точки зрения, оба источника в модели передачи информации равнозначны, поэтому подразделять источники на передатчик и приемник, имея в виду направление передачи информации, здесь и в дальнейшем не всегда имеет смысл.

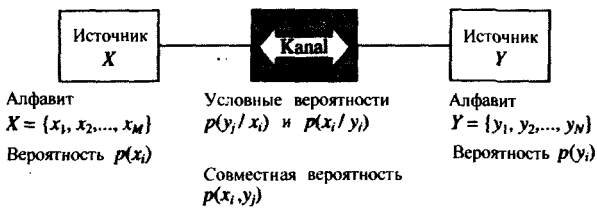


Рис. 7.4. Два дискретных источника без памяти, связанные каналом.

В главе 4 (см. таб. 4.3) совместная энтропия двух источников определена как математическое ожидание информации всех возможных пар событий

$$\frac{H(X,Y)}{\text{бит}} = -\sum_X \sum_Y p(x,y) \log_2 p(x,y). \tag{7.5}$$

Точно так же определяется условная энтропия

$$\begin{aligned} \frac{H(Y/X)}{\text{бит}} &= -\sum_X \sum_Y p(x,y) \log_2 p(y/x) \\ \frac{H(X/Y)}{\text{бит}} &= -\sum_X \sum_Y p(x,y) \log_2 p(x/y). \end{aligned} \tag{7.6}$$

Из этого следует

$$H(X,Y) = H(Y) + H(X/Y) = H(X) + H(Y/X) \tag{7.7}$$

и

$$H(X,Y) \leq H(Y) + H(X), \tag{7.8}$$

причем, знак равенства имеет место только для независимых источников.

В случае двух источников, связанных каналом, совместная неопределенность снижается, так как событие одного источника позволяет заранее предполагать событие другого источника. С точки зрения теории информации, снижение неопределенности означает обмен информацией между источниками. Рассуждая аналогично, приходим к выводу, что среднее значение информации, передаваемой по каналу, определяется как математическое ожидание взаимных информаций всех пар событий.

Среднее значение информации, которым обмениваются два дискретных источника без памяти X и Y, равно

$$\begin{aligned}\frac{I(X;Y)}{\text{бит}} &= \sum_X \sum_Y p(x,y)\log_2\frac{p(y/x)}{p(y)} = \\ &= \sum_X \sum_Y p(x,y)\log_2\frac{p(x/y)}{p(x)}.\end{aligned} \tag{7.9}$$

Замечание. *Обратите внимание на знак «минус» в левой части равенства и знак «плюс» перед правой частью.*

Из определения передаваемой информации следует

$$\begin{aligned}\frac{I(X;Y)}{\text{бит}} &= \underbrace{\sum_X \sum_Y p(x,y)\log_2 p(x/y)}_{-H(X/Y)\text{ бит}} - \\ &\quad \underbrace{-\sum_X \log_2 p(x) \underbrace{\sum_Y p(x,y)}_{p(x)}}_{H(X)\text{ бит}},\end{aligned} \tag{7.10}$$

и, поэтому,

$$I(X;Y) = H(X) - H(X/Y) = H(Y) - H(Y/X). \tag{7.11}$$

В этом месте опять возникает вопрос о сущности аксиоматического определения энтропии. В качестве «пробного камня» докажем справедливость следующего утверждения.

Теорема 7.3.1. Передаваемая информация $I(X;Y)$ всегда неотрицательна, причем, она равна нулю только для независимых источников X и Y

$$I(X;Y) \geq 0. \tag{7.12}$$

Доказательство.

При доказательстве будем исходить из определения $I(X;Y)$ и используем три приема. Во-первых, воспользуемся оценкой функции натурального логарифма (2.19). Во-вторых, без ограничения общности будем рассматривать только такие пары символов, вероятность которых отлична от нуля. В третьих, в аргументе логарифмической функции из (2.19) поменяем местами числитель и знаменатель, что эквивалентно умножению логарифмической функции на минус 1, поэтому, нам достаточно доказать справедливость неравенства

$$\frac{-I(X;Y)}{\text{нат}} = \sum_X \sum_Y p(x,y) \ln \frac{p(x)}{p(x/y)} \leq 0. \tag{7.13}$$

Так как, в силу сделанного нами ограничения, суммы берутся только по парам (x,y), для которых $p(x,y) \neq 0$, аргумент логарифмической функции $\frac{p(x)}{p(x/y)}$ всегда имеет отличное от нуля конечное положительное значение, поэтому, используем оценку (2.19)

$$\begin{aligned} \frac{-I(X;Y)}{\text{нат}} &= \sum_X \sum_Y p(x,y) \left[\frac{p(x)}{p(x/y)} - 1 \right] = \\ &= \sum_X \sum_Y \underbrace{\frac{p(x,y)p(x)}{p(x/y)}}_{p(x)p(y)} - \underbrace{\sum_X \sum_Y p(x,y)}_{1} = \\ &= \sum_X \sum_Y p(x)p(y) - 1 \leq 0. \end{aligned} \tag{7.14}$$

■

Если бы (7.12) не выполнялось, передача информации не снижала бы энтропии (т.е. определенность источника не повышалась бы).

То, что переданная информация всегда неотрицательна и всегда справедливы равенства (7.11) и (7.7), лишний раз подтверждает справедливость следующих утверждений:

Любое ограничение не может повышать неопределенность источника

$$H(X) \geq H(X/Y). \tag{7.15}$$

Совместная энтропия достигает своего максимума, когда источники независимы

$$H(X,Y) \leq H(X) + H(Y). \tag{7.16}$$

Найденные зависимости можно наглядно пояснить при помощи диаграммы потоков информации (рис.7.5). Диаграмма помогает уяснить смысл условных энтропий $H(X/Y)$ и $H(Y/X)$.

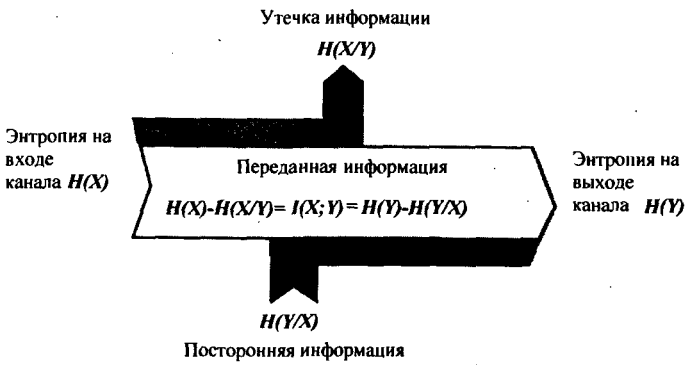


Рис. 7.5. Диаграмма информационных потоков.

$H(X/Y)$ – определяет среднюю меру неопределенности посланного символа источника X в том случае, когда символы приемника источника Y известны, т. е. оставшуюся неопределенность приемника. Величину $H(X/Y)$ часто называют также «утечкой» информации, так как энтропию $H(X)$ можно интерпретировать как собственную информацию источника X, $H(X) = I(X;X)$. В бесшумном канале $H(X/Y) = 0$ информация передается от источника X к источнику Y и обратно без потерь (без «утечки»). Если канал полностью зашумлен, то $H(X/Y) = H(X)$ и никакой передачи информации не происходит («утекает» вся информация).

$H(Y/X)$ – определяет среднюю неопределенность принятого символа при известных посланных символах, поэтому ее называют «посторонней» шумовой информацией.

Передачу информации по зашумленному каналу можно рассматривать как серию случайных экспериментов, которые способствуют снижению неопределенности. С точки зрения теории информации, канал является источником шумов.

Пример: Передача информации по двоичному симметричному каналу (ДСК).

Поясним физический смысл величины $I(x;y)$ на примере ДСК (рис. 7.2). Для двоичного симметичного канала имеем

$$\frac{I(X;Y)}{\text{бит}} = \sum_{i=1}^{2}\sum_{j=1}^{2} p(x_i,y_j)\log_2\frac{p(y_j/x_i)}{p(y_j)}. \qquad (7.17)$$

Как видим, $I(X;Y)$ зависит только от двух параметров – вероятности ошибки в канале и вероятности появления символа x_1 на выходе канала $p(x_1)$. При этом выполняются следующие выражения

$$\begin{aligned} p(x_1) &= p \text{ и } p(x_2) = 1-p \\ P_{Y/X} &= \begin{pmatrix} p(y_1/x_1)\; p(y_2/x_1) \\ p(y_1/x_2)\; p(y_2/x_2) \end{pmatrix} = \begin{pmatrix} 1-\varepsilon & \varepsilon \\ \varepsilon & 1-\varepsilon \end{pmatrix} \\ p(y_1) &= (1-\varepsilon)p + \varepsilon(1-p) \text{ и } p(y_2) = 1-p(y_1) \\ p(x_i,y_i) &= p(y_j/x_i)p(x_i). \end{aligned} \qquad (7.18)$$

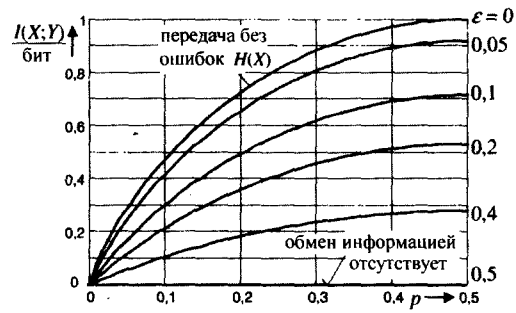


Рис. 7.6. Передача информации по двоичному симметричному каналу с вероятностью ошибки ε для различных значений вероятности символа на входе канала p.

Результаты вычислений для $I(X;Y)$ при различных p и ε представлены на рис. 7.6 в виде семейства кривых $I(X;Y) = f(p)$ при $\varepsilon = \{0.05, 0.1, 0.2, 0.4, 0.5\}$. В канале без шума $\varepsilon = 0$ передача информации происходит без искажений и информация $I(X;Y)$, в этом случае, равна энтропии $H(x)$ на входе канала. С увеличением уровня шума, вероятность ошибки ε повышается, а количество переданной

информации снижается, причем, относительно малый уровень шума $\varepsilon = 0,05$ приводит к заметному снижению $I(X;Y)$. В полностью зашумленном канале $\varepsilon = 0,5$ передача информации невозможна.

Интересно отметить, что при фиксированных значениях ε, информация $I(X;Y)$ существенно зависит от вероятности p на входе канала. При $p = 1/2$ через канал передается максимальное количество информации. В разделе 7.5, в котором будет введено новое понятие – пропускная способность канала, это свойство $I(X;Y)$ будет рассмотрено подробно.

Пример: Связанные источники.

Мы хотим дополнительно пояснить смысл энтропии на числовом примере. Для этого мы предлагаем такую конструкцию связанных источников, в которой все интересующие нас величины могут быть достаточно просто подсчитаны.

В таблице 7.1 задан дискретный источник без памяти Z с символами из алфавита $\{0,1,2,3\}$ и соответствующими вероятностями символов. Каждый символ z_i кодируется двоичным кодом с первым битом x_i и вторым битом y_i. Мы будем интерпретировать эти биты как символы двух связанных источников X и Y.

Таблица 7.1. Источник Z и его двоичное кодирование.

i	z_i	$pz(z_i)$	x	y
1	0	1/2	0	0
2	1	1/4	0	1
3	2	1/8	1	0
4	3	1/8	1	1

Выполните следующие задания:

1. Опишите источники X и Y;
2. Установите связь между источниками X и Y в форме модели канала, в которой источник X является входом канала, а источник Y – его выходом;
3. Приведите для задания 2 диаграмму информационных потоков и найдите для этой диаграммы числовые значения энтропий;
4. Найдите энтропию источника Z;

5. Выполните задания 2 и 3, считая источник Y входом канала.

Решение.

1. Начнем с описания источников X и Y. Оба источника являются дискретными источниками без памяти. Используя таблицу 7.1, найдем распределение вероятностей символов 0 и 1 для каждого из них

$$p_X(0) = p_Z(0) + p_Z(1) = \frac{3}{4} \text{ и} \tag{7.19}$$
$$p_X(1) = p_Z(2) + p_Z(3) = \frac{1}{4};$$
$$p_Y(0) = p_Z(0) + p_Z(2) = \frac{5}{8} \quad \text{и} \tag{7.20}$$
$$p_Y(1) = p_Z(1) + p_Z(3) = \frac{3}{8}.$$

Согласно (2.34), энтропии источников равны

$$\frac{H(X)}{\text{бит}} = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) \approx 0,8113, \tag{7.21}$$

$$\frac{H(Y)}{\text{бит}} = -\frac{5}{8}\log_2\left(\frac{5}{8}\right) - \frac{3}{8}\log_2\left(\frac{3}{8}\right) \approx 0,9544. \tag{7.22}$$

2. Модель канала представляет собой двоичный канал с символами x_1 и x_2 на входе и символами y_1 и y_2 на выходе. Канал может быть задан матрицей переходных вероятностей (7.1), содержащей вероятности $p(y_j/x_i)$. Из (7.19), (7.20) и таблицы 7.1 следует, что

$$\begin{aligned}
P_{Y/X}(0/0) &= \frac{P_{X,Y}(0,0)}{P_X(0)} = \frac{P_Z(0)}{P_X(0)} = \frac{1/2}{3/4} = \frac{2}{3} \\
P_{Y/X}(0/1) &= \frac{P_{X,Y}(1,0)}{P_X(1)} = \frac{P_Z(2)}{P_X(1)} = \frac{1/8}{1/4} = \frac{1}{2} \\
P_{Y/X}(1/0) &= \frac{P_{X,Y}(0,1)}{P_X(0)} = \frac{P_Z(1)}{P_X(0)} = \frac{1/4}{3/4} = \frac{1}{3} \\
P_{Y/X}(1/1) &= \frac{P_{X,Y}(1,1)}{P_X(1)} = \frac{P_Z(3)}{P_X(1)} = \frac{1/8}{1/4} = \frac{1}{2}.
\end{aligned} \tag{7.23}$$

В результате получим матрицу переходных вероятностей канала

$$\mathbf{P}_{Y/X} = \begin{pmatrix} 2/3 & 1/3 \\ 1/2 & 1/2 \end{pmatrix}. \tag{7.24}$$

Замечание. *Как и следовало ожидать, матрица является стохастической, так как сумма вероятностей в каждой ее строке равна единице.*

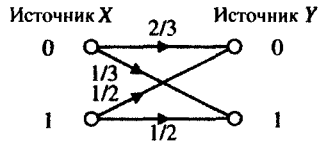


Рис. 7.7. Двоичный канал.

Диаграмма канала с вероятностями переходов приведена на рис. 7.7. Можно заметить ее сходство с диаграммой (рис. 7.2). Однако, в нашем примере, уже нельзя говорить об ошибках в канале.

3. Для построения диаграммы информационных потоков необходимо знание величин $H(Y/X)$, $H(X/Y)$ и $I(X;Y)$. По известным переходным вероятностям можно вычислить $H(Y/X)$

$$\frac{H(Y/X)}{\text{бит}} = - p_Z(0)\log_2 p_{Y/X}(0/0) - p_Z(1)\log_2 p_{Y/X}(1/0) - p_Z(2)\log_2 p_{Y/X}(0/1) - p_Z(3)\log_2 p_{Y/X}(1/1). \quad (7.25)$$

Подставляя числовые значения, находим

$$\frac{H(Y/X)}{\text{бит}} = -\frac{1}{2}\log_2\left(\frac{2}{3}\right) - \frac{1}{4}\log_2\left(\frac{1}{3}\right) - \frac{1}{8}\log_2\left(\frac{1}{2}\right) - \frac{1}{8}\log_2\left(\frac{1}{2}\right) \approx 0,9387. \quad (7.26)$$

Величины $I(X;Y)$ и $H(X/Y)$ можно найти из (7.11).

Диаграмма информационных потоков представлена на рис. 7.8.

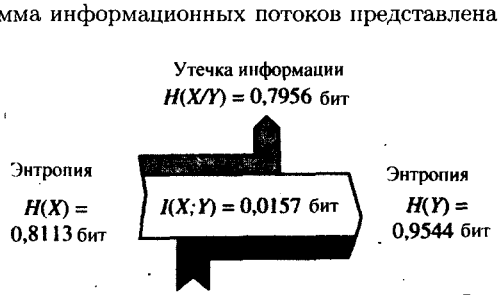


Рис. 7.8. Диаграмма информационных потоков связанных источников.

4. Энтропия источника Z равна

$$\frac{H(Z)}{\text{бит}} = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) - \frac{2}{8}\log_2\left(\frac{1}{8}\right) = 1,75. \quad (7.27)$$

Полученным результатам можно дать следующую интерпретацию. Энтропия источника Z равна совместной энтропии двоичных источников X и Y. Энтропия источника $H(X)$ равна 0,8113, остальные 0,9387 бит вносит условная энтропия $H(Y/X)$ согласно (7.7).

5. По аналогии с заданием 2, находим

$$\begin{aligned}
P_{X/Y}(0/0) &= \frac{P_{X,Y}(0,0)}{P_Y(0)} = \frac{P_Z(0)}{P_Y(0)} = \frac{1/2}{5/8} = \frac{4}{5} \\
P_{X/Y}(0/1) &= \frac{P_{X,Y}(0,1)}{P_Y(1)} = \frac{P_Z(1)}{P_Y(1)} = \frac{1/4}{3/8} = \frac{2}{3} \\
P_{X/Y}(1/0) &= \frac{P_{X,Y}(1,0)}{P_Y(0)} = \frac{P_Z(2)}{P_Y(0)} = \frac{1/8}{5/8} = \frac{1}{5} \\
P_{X/Y}(1/1) &= \frac{P_{X,Y}(1,1)}{P_Y(1)} = \frac{P_Z(3)}{P_Y(1)} = \frac{1/8}{3/8} = \frac{1}{3}
\end{aligned} \quad (7.28)$$

и получаем матрицу канала

$$\mathbf{P}_{Y/X} = \begin{pmatrix} 4/5 & 1/5 \\ 2/3 & 1/3 \end{pmatrix}. \quad (7.29)$$

Диаграмма канала и диаграмма информационных потоков показаны на рис. 7.9 и рис. 7.10.

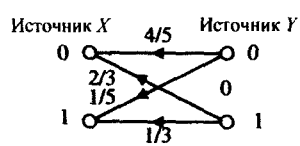


Рис. 7.9. Двоичный канал.

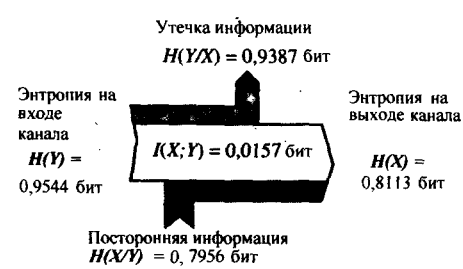


Рис. 7.10. Диаграмма информационных потоков.

7.4. Выводы

Все определения и величины, рассмотренные в предыдущих разделах, обобщены в таблицах 7.2 и 7.3. Следует обратить особое внимание на переходы от теории вероятностей к теории информации.

При этих переходах выходные символы дискретных источников без памяти рассматриваются как исходы случайных экспериментов. В соответствии с вероятностями этих исходов, каждому символу приписывается некоторая информационная мера, равная логарифму вероятности его появления. Заметим, что вероятности символов можно рассматривать как стохастические переменные.

При переходе от вероятностей символов к их информационному содержанию, вводится новая величина, не имеющая аналога в теории вероятностей – взаимная информация, которая возникает при анализе пар совместных событий (x, y). Взаимная информация определяется как логарифм отношения апостериорной вероятности символа y $p(y/x)$ к его априорной вероятности $p(y)$ и служит информационной мерой связности двух событий.

Для описания источников используются средние значения информаций символов, генерируемых источником. Таким образом, вводится понятие энтропии как математического ожидания количества информации отдельных событий (символов) или пар событий. При этом, особую роль играет взаимная информация. Ее математическое ожидание $I(X;Y)$ является мерой передаваемой информации и характеризует связь между двумя источниками X и Y, т.е. описывает среднее количество информации, которой обмениваются между собой источники по каналу связи. Основополагающее значение величины $I(X;Y)$ будет подробно раскрыто в следующих разделах.

Таблица 7.2. Дискретные источники без памяти X и Y с символами $x \in X = \{x_1, x_2, \ldots, x_M\}$ и $y \in Y = \{y_1, y_2, \ldots, y_N\}$.

Теория вероятностей	Теория информации
Вероятность отдельного символа (априорная вероятность) $p(x)$	Информация отдельного символа $I(x) = -\log_2 p(x)$ бит (7.30)
Совместная вероятность двух символов $p(x,y)$	Информация пары символов $I(x,y) = p(x,y)$ бит (7.31)
Условная вероятность (апостериорная вероятность) $p(x/y) = \dfrac{p(x,y)}{p(y)}$ (7.32) $p(y/x) = \dfrac{p(x,y)}{p(x)}$	Условная информация $I(x/y) = -\log_2 p(x/y)$ бит (7.33) $I(y/x) = -\log_2 p(y/x)$ бит
	Взаимная информация $I(x;y) =$ $= \log_2 \dfrac{\text{апостериорная инф.}}{\text{априорная информ.}}$ бит $=$ $= \log_2 \dfrac{p(x/y)}{p(x)}$ бит $= \dfrac{p(x/y)}{p(x)}$ бит (7.34)
Обозначения $p(x) = P_X(x_i)$ для $x_i \in X$ $p(x/y) = P_{X/Y}(x_i/y_j)$ для $x_i \in X$ и $y_j \in Y$ $\sum\limits_X p(x) = \sum\limits_{i=1}^{M} P_X(x_i)$ $\sum\limits_Y p(y) = \sum\limits_{j=1}^{M} P_Y(y_i)$	Некоторые важные соотношения, используемые для расчетов $\sum\limits_X p(x) = 1$; $\sum\limits_Y p(y) = 1$ $\sum\limits_X \sum\limits_Y p(x,y) = 1$ $\sum\limits_Y p(x,y) = p(x)$; $\sum\limits_X p(x,y) = p(y)$ $\sum\limits_Y p(x/y) = 1$; $\sum\limits_X p(x/y) = 1$

Таблица 7.3. Описание «в среднем» дискретных источников без памяти с символами $x \in X = \{x_1, x_2, \ldots, x_M\}$ и $y \in Y = \{y_1, y_2, \ldots, y_N\}$.

Энтропия	$H(X) = -\sum_X p(x)\log_2 p(x)$ бит (7.35) $H(Y) = -\sum_Y p(y)\log_2 p(y)$ бит
Совместная энтропия	$H(X,Y) = -\sum_X \sum_Y p(x,y)\log_2 p(x,y)$ (7.36)
Условная энтропия	$H(X/Y) = -\sum_X \sum_Y p(x,y)\log_2 p(x/y)$ (7.37) $H(Y/X) = -\sum_X \sum_Y p(x,y)\log_2 p(y/x)$
Передача информации	$I(X;Y) =$ (7.38) $\sum_X \sum_Y p(x,y)\log_2 \frac{\text{апостериорная вер.}}{\text{априорная вер.}} =$ $= \sum_X \sum_Y p(x,y)\log_2 \frac{p(y/x)}{p(y)}$ бит $=$ $= \sum_X \sum_Y p(x,y)\log_2 \frac{p(x/y)}{p(x)}$ бит
Некоторые важнейшие зависимости (знак равенства справедлив только для независимых источников)	$H(X) \geq H(X/Y)$ и $H(Y) \geq H(Y/X)$ (7.39) $H(X,Y) = H(X) + H(Y/X) =$ (7.40) $= H(Y) + H(X/Y)$ $H(X,Y) = H(X) + H(Y) - I(X;Y)$ (7.41) $I(X;Y) \geq 0$ (7.42)

7.5. Пропускная способность канала

Величина $I(X;Y)$ играет особую роль в теории информации и описывает передачу информации по каналу связи. Из определения (7.9) следует, что $I(X;Y)$ зависит как от переходных вероятностей канала, так и от распределения вероятностей символов на входе канала.

Для дальнейших рассуждений рассмотрим дискретный канал без памяти с фиксированными переходными вероятностями и зададимся вопросом: Какое максимальное количество информации можно передать по данному каналу?

Пропускная способность канала с заданными переходными вероятностями равна максимуму передаваемой информации по всем входным распределениям символов источника X

$$C = \max_X I(X;Y). \tag{7.43}$$

Замечание. *Размерность пропускной способности – бит/символ. Если, например, по каналу передается один символ в сек, то можно также говорить о размерности бит/сек.*

Так как максимум ищется по всем допустимым входным источникам, то пропускная способность зависит только от переходных вероятностей канала.

С математической точки зрения, поиск пропускной способности дискретного канала без памяти сводится к поиску распределения вероятностей входных символов источника, обеспечивающего максимум информации $I(X;Y)$. При этом, на вероятности входных символов $x \in X$ накладываются ограничения

$$0 < p(x) \leq 1 \text{ и } \sum_X p(x) = 1. \tag{7.44}$$

В принципе, определение максимума $I(x,y)$ при ограничениях (7.44) возможно при использовании мультипликативного *метода Лагранжа*. Однако, такое решение требует чрезмерно больших затрат. В частном случае (симметричные каналы) найти пропускную способность помогает следующая теорема [10].

Теорема 7.5.1. В симметричных дискретных каналах без памяти пропускная способность достигается при равномерном распределении вероятностей входных символов источника X.

Замечание. *В [10] приводится также метод, позволяющий определить, является ли канал симметричным или нет.*

7.5.1. Пропускная способность

Двоичный дискретный симметричный канал без памяти (ДСК) определяется с помощью матрицы переходных вероятностей канала (7.2). Единственным параметром, характеризующим ДСК, является вероятность ошибки ε. Из равномерного распределения входных символов и симметрии переходов канала следует равномерное распределение выходных символов, т.е

$$p(x_1) = p(x_2) = p(y_1) = p(y_2) = 1/2. \tag{7.45}$$

Используя (7.9), получаем

$$\begin{aligned} \frac{C}{\text{бит}} &= \sum_i^2 \sum_j^2 p(x_i, y_i) \log_2 \frac{p(y_j/x_i)}{p(y_i)} = \\ &= \sum_i^2 \sum_j^2 p(x_i) p(y_j/x_i) \log_2 \frac{p(y_j/x_i)}{p(y_j)}. \end{aligned} \tag{7.46}$$

Подставляя числовые значения, имеем

$$\begin{aligned} \frac{C}{\text{бит}} &= (1-\varepsilon)\log_2(2(1-\varepsilon)) + \varepsilon \log_2(2\varepsilon) = \\ &= 1 + \underbrace{(1-\varepsilon)\log_2(1-\varepsilon) + 2\log_2 \varepsilon}_{-H_b(\varepsilon)}. \end{aligned} \tag{7.47}$$

Энтропия ДСК определяется через (2.32)

$$\frac{H_b(\varepsilon)}{\text{бит}} = -(1-\varepsilon)\log_2(1-\varepsilon) - \varepsilon \log_2(\varepsilon). \tag{7.48}$$

Окончательно получаем пропускную способность ДСК в компактной форме

$$C^{\text{ДСК}} = 1 \text{ бит} - H_b(\varepsilon) \tag{7.49}$$

Интересными являются два граничных случая:

1. Передача информации по бесшумному каналу:
   $H_b(\varepsilon = 0) = 0$ и $C^{\text{ДСК}} = 1$ бит.

2. Канал полностью зашумлен:
   $H_b(\varepsilon = 1/2) = 1$ бит и $C^{\text{ДСК}} = 0$ бит.

7.5.2. Пропускная способность двоичного симметричного канала со стираниями

Важным частным случаем ДСК является *двоичный симметричный канал со стираниями (ДСКС)* или двоичный канал со стираниями (Binary Erasure Channel, BEC – англ.). Как и ДСК, двоичный канал со стираниями может служить упрощенной моделью передачи информации по каналу с аддитивным белым гауссовским шумом (АБГШ). Правило принятия решения в ДСКС приведено на рис.7.11. Из рисунка видно, что наряду с решениями о переданном символе «0» или «1», здесь иногда принимается решение о стирании принятого символа «e»(Erasure-англ.). Стирание происходит в случае, если продетектированный аналоговый сигнал V попадает в зону, для которой значения условных функций плотности распределения вероятностей $f(V/0)$ и $f(V/1)$ оказываются близкими к нулю.

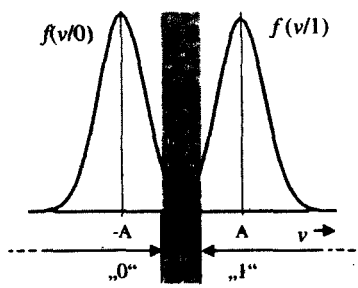


Рис. 7.11. Условные функции плотности распределения вероятностей продетектированного сигнала и области принятия решений.

Замечание. *В двоичном канале со стираниями, вместо однозначно «жесткого» решения о принятом символе «0» или «1» принимается, так называемое, «мягкое» решение. В этом случае, мы дополнительно имеем некоторую информацию о надежности принятого двоичного символа. В связи с этим, в технике передачи данных, говорят о приеме с «жестким» и «мягким» решением. «Мягкое» решение в сочетании с подходящим кодированием информации позволяет в некоторых случаях осуществить более надежную передачу данных. Один из примеров использования «мягкого» решения можно найти во второй части этой книги.*

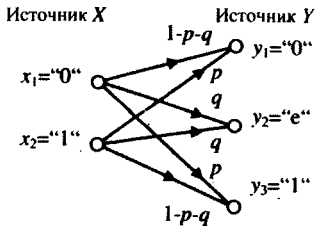


Рис. 7.12. Двоичный симметричный канал со стираниями.

Обозначим вероятность стирания через q, а вероятность ошибки нестертого символа через p.

Диаграмма переходов для канала с двумя входными и тремя выходными символами приведена на рис. 7.12. Соответствующая матрица канала, содержащая переходные вероятности, имеет вид

$$\mathbf{P}_{Y/X}^{\text{ДСКС}} = \begin{pmatrix} 1-p-q & q & p \\ p & q & 1-p-q. \end{pmatrix} \tag{7.50}$$

Найдем пропускную способность канала со стираниями. Так как канал симметричен, пропускная способность достигается при равномерном распределении входных символов

$$p(x_1) = p(x_2) = 1/2. \tag{7.51}$$

Отсюда следует, что вероятности выходных символов равны

$$p(y_1) = \frac{1-q}{2}, p(y_2) = q, p(y_3) = \frac{1-q}{2}. \tag{7.52}$$

Теперь все необходимые вероятности известны. Воспользовавшись (7.9), имеем

$$\begin{aligned} \frac{C^{\text{ДСКС}}}{\text{бит}} &= \sum_i^2 \sum_j^3 p(x_i, y_j) \log_2 \frac{p(y_j/x_i)}{p(y_j)} = \\ &= \sum_i^2 \sum_j^3 p(x_i) p(y_j/x_i) \log_2 \frac{p(y_j/x_i)}{p(y_j)}. \end{aligned} \tag{7.53}$$

Используя свойство симметрии канала, получаем

$$\begin{aligned}\frac{C^{\text{ДСКС}}}{\text{бит}} &= (1-p-q)\log_2\left(2\frac{1-p-q}{1-q}\right)+\\ &+ q\log_2\frac{q}{q}+p\log_2\left(2\frac{p}{1-q}\right)= \\ &= 1-q+(1-p-q)\log_2\frac{1-p-q}{1-q}+p\log_2\frac{p}{1-q}.\end{aligned} \tag{7.54}$$

Как мы видим, пропускная способность канала со стираниями зависит только от вероятностей p и q. График $C=f(p,q)$ представляет собой пространственную трехмерную поверхность, расположенную над плоскостью (p,q). Здесь мы ограничимся только рассмотрением двух важных частных случаев.

1. При $q=0$, мы имеем двоичный симметричный канал, уже рассмотренный ранее. Подставляя $q=0$ в (7.59) мы, как и ожидалось, получаем (7.49).

2. В канале присутствуют только стирания, т.е. при $p=0$ - ошибки или не присутствуют, или мы ими пренебрегаем. В этом случае

$$C^{\text{ДСКС}}=(1-q)\text{ бит}. \tag{7.55}$$

На рис. 7.13 показаны пропускные способности ДСК (7.49) и двоичного канала со стираниями ($p=0$). Нужно отметить, что при малых вероятностях ошибки, выбором оптимальных областей стираний в ДСКС можно достичь существенно больших пропускных способностей, чем в обычных двоичных каналах.

Замечание. *Здесь возникает вопрос о возможности увеличения пропускной способности при приеме со стираниями на практике. В этом месте обнаруживается слабость теории информации. Теория информации зачастую не может предложить конструкцию, реализующую теоретически достижимые границы. Тем не менее, небольшой пример, подробно рассмотренный во второй части этой книги, показывает, что введение стираний может иногда снижать вероятность ошибки. Рассмотрим этот пример на интуитивном уровне. Разобьем поток передаваемой информации на блоки, содержащие 7 двоичных символов (7 бит). К каждому блоку добавим один бит («0» или «1») проверки на четность. Закодированные таким образом блоки из восьми двоичных символов всегда будут содержать четное число единиц. Пусть вероятность ошибки в ДСК достаточно мала. Введем зону стирания (рис.7.11) таким образом,*

чтобы ошибки, в основном, перешли в стирания. При этом, вероятность «нестертой» ошибки будет пренебрежимо мала, а вероятность стирания будет оставаться достаточно малой. Мы получим стирающий канал (ДСКС), в котором блоки из восьми двоичных символов в подавляющем большинстве случаев или будут приняты правильно или будут содержать только один стертый двоичный символ. Качество приема существенно улучшится, так как одно стирание в блоке с четным числом единиц всегда может быть исправлено.

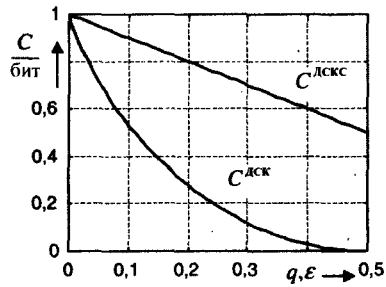


Рис. 7.13. Пропускная способность двоичного симметричного канала $C^{\text{ДСК}}$ с вероятностью ошибки ε и двоичного канала со стираниями $C^{\text{ДСКС}}$ с вероятностью стирания q и вероятностью ошибки $p = 0$.

Пример: Двоичный симметричный канал со стираниями.

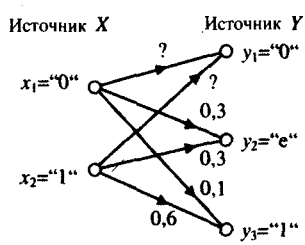


Рис. 7.14. Двоичный канала со стираниями.

На рис. 7.14 задана переходная диаграмма симметричного канала со стираниями. Определите:

1. Матрицу канала ;

2. Распределение вероятностей символов источника Y, если известно, что символы источника X равномерно распределены, т.е. $p_0 = p_1 = 1/2$;

3. Пропускную способность канала;

4. Диаграмму информационных потоков со всеми энтропиями;

5. Модель канала с матрицей $\mathbf{P}_{X/Y}$.

Решение.

1. С учетом того, что сумма вероятностей в каждой строке матрицы равна 1, получим

$$\mathbf{P}_{Y/X}^{\text{ДСКС}} = \begin{pmatrix} 0,6 & 0,3 & 0,1 \\ 0,1 & 0,3 & 0,6 \end{pmatrix}. \tag{7.56}$$

2. Исходя из равномерного распределения вероятностей символов на входе, согласно (7.52), имеем

$$p(y_1) = 0,35, \quad p(y_2) = 0,3 \text{ и } p(y_3) = 0,35. \tag{7.57}$$

3. Так как рассматриваемый канал симметричен, пропускная способность достигается при равномерном распределении входных символов. Из (7.54) с учетом (7.56) имеем

$$\frac{C^{\text{ДСКС}}}{\text{бит}} = 1 - 0,3 + (1 - 0,3 - 0,1)\log_2 \frac{1 - 0,3 - 0,1}{1 - 0,35} + \tag{7.58}$$
$$+0,1 \cdot \log_2 \frac{0,1}{1 - 0,3} = 0,7 + 0,6 \cdot \log_2 \frac{0,6}{0,7} + 0,1 \cdot \log_2 \frac{0,1}{0,7} \approx 0,2858.$$

4. Энтропия дискретного двоичного источника без памяти X с равномерным распределением вероятностей символов равна

$$H(X) = 1 \text{ бит}. \tag{7.59}$$

Энтропия источника Y равна

$$\frac{H(Y)}{\text{бит}} = -2 \cdot 0,35 \log_2 0,35 - 0,35 \log_2 0,3 \approx 1,5813. \tag{7.60}$$

Так как в симметричном канале с равномерным распределением входных символов $I(X;Y)$ совпадает с пропускной способностью

C из (7.58), совместную энтропию и две условные энтропии можно подсчитать, используя таблицу 7.3. Диаграмма информационных потоков изображена на рис. 7.15.

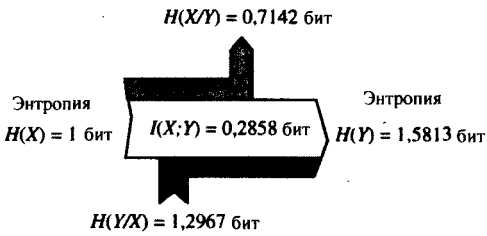


Рис. 7.15. Диаграмма информационных потоков двоичного симметричного канала со стираниями.

5. Пересчет матрицы переходных вероятностей канала $\mathbf{P}_{Y/X}$ в матрицу $\mathbf{P}_{X/Y}$ предоставляем сделать читателю в качестве самостоятельного упражнения. Диаграмма канала с входным источником Y и выходным X приведена на рис. 7.16 для контроля.

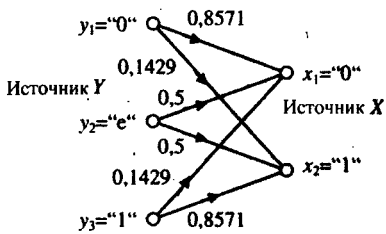


Рис. 7.16. Двоичный симметричный канал со стираниями.

7.6. Теорема кодирования для дискретных каналов без памяти

Рассмотрим дискретный канал без памяти с пропускной способностью C[бит/символ], в котором каждый символ передается в течении T_s сек. Для этого канала C[бит/сек]$= C$[бит/сек]$/T_s$.

Пусть энтропия некоторого источника X, измеренная в течении T_s сек, составляет $H(X)$ бит. Тогда имеет место следующая теорема.

Теорема 7.6.1. *Теорема кодирования для канала* (теорема Шеннона).

Для источника X со скоростью $R = H(X)/T_s$ [бит/сек] и $R < C$ существует некоторый код, с помощью которого информация источника X может быть передана по каналу связи с пропускной способностью C[бит/сек] со сколь угодно малой вероятностью ошибки.[1]

Доказательство теоремы кодирования для канала (см.,например, [10]) довольно сложно и выходит за рамки этой книги, поэтому ограничимся здесь следующими замечаниями.

- Доказательство теоремы кодирования предполагает использование случайных кодов бесконечной длины и декодера максимального правдоподобия, обеспечивающего минимальную вероятность ошибки. Доказательство не использует никаких конструктивных решений. В нем используются только статистические свойства и предельные переходы для блоковых кодов с длиной блоков, стремящейся к бесконечности. Доказательство не дает никаких указаний на конструкцию оптимальных кодов.

- Теорема кодирования определяет также верхнюю границу для скорости передачи R. [2]

- При доказательстве теоремы вводится показатель экспоненциальной оценки R_0, который может быть использован для оценки технически достижимой скорости передачи данных [4].

[1] Теорема кодирования справедлива не только для дискретных каналов, она также верна и при передаче дискретных сообщений по непрерывным каналам. – *Прим. перев.*

[2] Здесь необходимо сделать разъяснение. Существует обратная теорема кодирования, которая говорит о том, что при $R > C$ не существует никакого метода кодирования, позволяющего передавать информацию с как угодно малой вероятностью ошибки. – *Прим. перев.*

ГЛАВА 8

НЕПРЕРЫВНЫЕ ИСТОЧНИКИ И КАНАЛЫ

В главе 2 дано определение энтропии как меры неопределенности источника. При этом предполагалось, что энтропия измеряется посредством случайных экспериментов. В данной главе мы будем применять аналогичный подход к непрерывным источникам.

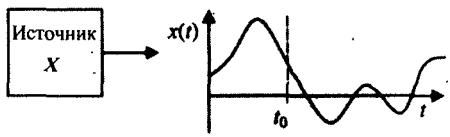


Рис. 8.1. Сигнал непрерывного источника.

Вместо источников с конечным алфавитом символов будем рассматривать источники, выходом которых являются непрерывные сигналы. Примером таких сигналов может служить изменяющееся во времени напряжение в телефонных каналах и т.д. На рисунке 8.1 представлен непрерывный источник X, выходом которого является аналоговый сигнал $x(t)$, являющийся некоторой случайной функцией от времени t. Будем рассматривать значения $x(t)$ в некоторые фиксированные моменты времени как случайные эксперименты, которые несут некоторую информацию об источнике X.

8.1. Дифференциальная энтропия

На рисунке 8.2 показаны два непрерывных источника X и Y, связанные каналом (аналогично рис. 7.4). Здесь, вместо вероятностей, стоят функции плотностей распределения вероятностей стохастических переменных.

Использование стохастических переменных и их функции плотностей распределения вероятностей позволяет вводить понятие ин-

формации, энтропии, условной и взаимной энтропии для двух непрерывных источников по аналогии с дискретными источниками.

Рис. 8.2. Два непрерывных источника без памяти, связанных каналом.

Преобразуем непрерывный источник X в дискретный. Для этого проквантуем значения аналогового выхода источника с шагом Δ (рис. 8.3).

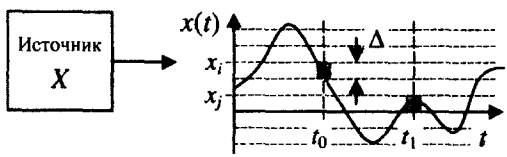


Рис. 8.3. Оцифровка непрерывного источника с интервалом квантования Δ в моменты наблюдения t_0, t_1 и т.д.

Кроме этого, как это обычно делается в теории информации, произведем дискретизацию источника по времени. В результате, получим последовательность стохастических переменных $X_0, X_1, X_2, \ldots$. Следуя таблице 7.2, определим взаимную информацию символов x_i и y_j, где x_i - значение выходного символа в момент времени t_m, а x_j - в момент времени t_n

$$\frac{I_{X_m X_n}(x_i; x_j)}{\text{бит}} = \tag{8.1}$$

$$= \log_2 \frac{P([x_i - \Delta \le X_m < x_i] \cap [x_j - \Delta \le X_n < x_j])}{P(x_i - \Delta \le X_m < x_i) P(x_j - \Delta \le X_n < x_j)} =$$

$$= \log_2 \frac{\int\limits_{x_i - \Delta}^{x_i} \int\limits_{x_j - \Delta}^{x_j} f_{X_m X_n}(x_1, x_2) dx_1 dx_2}{\int\limits_{x_i - \Delta}^{x_i} f_{X_m}(x_1) dx_1 \int\limits_{x_j - \Delta}^{x_j} f_{X_n}(x_2) dx_2}.$$

Взаимную информацию можно трактовать как «снятую» (утраченную) неопределенность попадания переменной X_n в интервале $[x_j-\Delta, x_j[$, когда известно, что переменная X_m принадлежит интервалу $[x_i-\Delta, x_i[$ или наоборот. Будем считать функцию плотности распределения вероятности непрерывной функцией. Тогда, устремляя ширину интервала квантования к нулю, получим

$$\lim_{\Delta\to\infty} \log_2 \frac{\int\limits_{x_i-\Delta}^{x_i}\int\limits_{x_j-\Delta}^{x_j} f_{X_mX_n}(x_1,x_2)dx_1dx_2}{\int\limits_{x_i-\Delta}^{x_i} f_{X_m}(x_1)dx_1 \int\limits_{x_j-\Delta}^{x_j} f_{X_n}(x_2)dx_2} = \tag{8.2}$$

$$= \lim_{\Delta\to\infty} \log_2 \frac{\Delta\Delta f_{X_mX_n}(x_i,x_j)}{\Delta f_{X_m}(x_i)\Delta f_{X_n}(x_j)} = \log_2 \frac{f_{X_mX_n}(x_i,x_j)}{f_{X_m}(x_i)f_{X_n}(x_j)},$$

т.е. результат, аналогичный выражению взаимной информации для дискретных источников. *Передаваемую информацию* можно определить как математическое ожидание

$$\frac{I(X;Y)}{\text{бит}} = \int\limits_{-\infty}^{+\infty}\int\limits_{-\infty}^{+\infty} f_{XY}(x,y)\log_2\frac{f_{XY}(x,y)}{f_X(x)f_Y(y)}dxdy. \tag{8.3}$$

Замечание. *Здесь, для приведения в соответствие обозначений этой главы с результатами таблицы 7.2, вместо X_m используется X, а вместо Y_n – Y.*

Информация источника определяется исходя из аналогичных рассуждений

$$\frac{I(x_i)}{\text{бит}} = -\log_2 P([x_i-\Delta \le X < x_i]) = \tag{8.4}$$

$$= -\log_2 \int\limits_{x_i-\Delta}^{x_i} f_X(x_i)dx \overset{\Delta\ll 1}{\approx} -\log_2(\Delta f_X(x_i)) =$$

$$= -\log_2\Delta - \log_2 f_X(x_i).$$

В отличие от выражения (8.3) для взаимной информации, в (8.4) появляется слагаемое, зависящее от интервала квантования Δ.

При $\Delta\to\infty$, величина $\log_2(\Delta)$ также стремится к бесконечности. В результате, выражение для $I(x_i)$ также стремится к ∞. Это не удивительно, так как с уменьшением шага квантования, число отдельных событий (символов алфавита источника) возрастает и, следовательно, неопределенность источника также растет.

Величина $\log_2(\Delta)$ не зависит от источника и совершенно не уместна для его описания, поэтому, кажется вполне естественно использовать только функцию плотности распределения вероятности непрерывного источника. Таким образом, мы переходим к следующему определению.

Средняя информация непрерывного источника, так называемая *дифференциальная энтропия*, определяется как

$$\frac{H(X)}{\text{бит}} = -\int_{\infty}^{\infty} f(x)\log_2 f(x)dx. \tag{8.5}$$

Прежде всего отметим, что такое произвольное определение дифференциальной энтропии подтверждает свою пригодность тем, что энтропийные отношения для дискретных источников оказываются справедливыми и для случая непрерывных источников и каналов. В частности, для непрерывных источников имеют место соотношения (7.39) - (7.42).

Таким образом, дифференциальная энтропия непрерывного источника зависит только от функции плотности распределения вероятности, которая в общем случае является бесконечной величиной, поэтому, поставим вопрос о том, как велико может быть значение дифференциальной энтропии. Прежде всего отметим, что характеристиками стохастического процесса являются две величины: среднее значение, которое принимает стохастическая переменная (обладающая свойством линейности) μ и стандартное отклонение стохастической переменной σ.

Среднее значение или математическое ожидание μ не оказывает никакого влияния на дифференциальную энтропию. С ростом же σ, неопределенность источника возрастает, что приводит также к возрастанию дифференциальной энтропии. В связи с этим, сравнение различных функций плотностей распределения вероятностей относительно соответствующих им энтропий имеет смысл производить при одинаковых σ.

Замечание. *В информационной технике за исходный параметр принимают σ^2 - дисперсию, которая определяет среднюю мощность стохастического процесса [10]. Ясно, что с увеличением мощности передатчика, количество передаваемой информации увеличивается и, наоборот, с увеличением мощности шумов, возрастает неопределенность, т.е. в единицу времени передается меньше информации.*

Из теории информации следует, что дифференциальная энтропия достигает своего максимума при гауссовском распределении вероятности.

Теорема 8.1.1. При заданной дисперсии σ^2, максимальной дифференциальной энтропией обладает источник с *гауссовским распределением* вероятности, причем,

$$H_{Gaus}(X) = \frac{1}{2}\frac{\ln(2\pi\sigma^2 e)}{\ln 2} \text{ бит.} \tag{8.6}$$

Пример: Дифференциальная энтропия гауссовского источника.

Из (8.5) следует, что дифференциальная энтропия гауссовского источника равна

$$\frac{H(X)}{\text{нат}} = \tag{8.7}$$

$$= -\int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \log_2 \left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right)\right] dx =$$

$$= -\int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \log_2 \left[\frac{1}{\sqrt{2\pi\sigma^2}} - \frac{x^2}{2\sigma^2}\right] dx.$$

Выражение в квадратных скобках может быть разложено на два интеграла. Таким образом, окончательно имеем

$$\frac{H(X)}{\text{нат}} = \ln\sqrt{2\pi\sigma^2} + \frac{1}{2\sigma^2}\frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{+\infty} x^2 \exp\left(-\frac{x^2}{2\sigma^2}\right) dx = \tag{8.8}$$

$$= \frac{1}{2}\ln[2\pi\sigma^2] + \frac{1}{2} = \frac{1}{2}\ln[2\pi\sigma^2 e].$$

Численные примеры для трех, наиболее употребительных распределений, приведены в таблице 8.1.

Пример: Телефония.

Практическая польза приведенных выше результатов может быть наглядно показана при помощи оценки достижений скорости передачи информации (в битах) в цифровых телефонных линиях. Современные стандартные методы цифровой передачи речи (логарифмические PCM) требуют затраты 8 бит на кодирование одного отсчета,

Таблица 8.1. Пример дифференциальной энтропии.

Распределения	Функция плотности распределения вероятности	Дифференциальная энтропия
Равномерное	$f(x) = \begin{cases} \frac{1}{2\sqrt{3}} & \text{для } \lvert x\rvert \le \sqrt{3} \\ 0 & const \end{cases}$	$\frac{\ln(2\sqrt{3}\sigma)}{\ln 2} = 1,79$
Лапласа	$f(x) = \frac{1}{\sqrt{2\sigma^2}} \exp\left(-\sqrt{2}\frac{\lvert x\rvert}{\sigma}\right)$	$\frac{\ln(\sqrt{2}\sigma e)}{\ln 2} = 1,94$
Гауссовское	$f(x) = \frac{1}{\sqrt{2\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$	$\frac{\ln(2\pi\sigma^2 e)}{2\ln 2} = 2,04$

при частоте отсчетов 8 кГц. Таким образом, скорость передачи речи составляет 64 кбит/сек.

Исходя из равномерного распределения вероятностей в интервале [-1,1], опытным путем получим $\sigma^2 = 1/3$. Таким образом, дифференциальная энтропия на один отсчет составляет

$$\frac{\hat{H}(X)}{\text{бит}} = \left.\frac{\ln(2\sqrt{3}\sigma)}{\ln 2}\right|_{\sigma^2=1/3} = 1. \tag{8.9}$$

Так как отсчеты производятся с частотой 8 кГц, получаем, что необходимая скорость передачи речи составляет 8 кбит/сек. При оценке энтропии мы не принимали во внимание связи между соседними отсчетами (память источника) и, поэтому, реальная дифференциальная энтропия источника речи будет еще меньше. В самом деле, мы знаем, что современные алгоритмы кодирования речи позволяют осуществлять передачу речевого сигнала со скоростью около 8 кбит/сек при качестве, сравнимом со стандартным PCM.

8.2. Пропускная способность канала и граница Шеннона

Аналогично дискретным каналам, можно определить пропускную способность для непрерывных каналов. Будем искать, как и ранее,

наибольшее значение переносимой информации по всем возможным функциям плотностей распределения вероятностей.

$$C = \sup_{f(x)} I(X;Y). \tag{8.10}$$

Поиск точной верхней грани, в общем случае, представляет собой довольно сложную задачу. Рассмотрим важнейший частный случай– передачу информации по каналу с аддитивным белым гауссовским шумом (АБГШ) с ограниченной полосой. Модель такого канала изображена на рис. 8.4. Для гауссовского распределения вероятности амплитуды сигнала на входе канала получаем формулу пропускной способности канала, хорошо известную в теории информации.

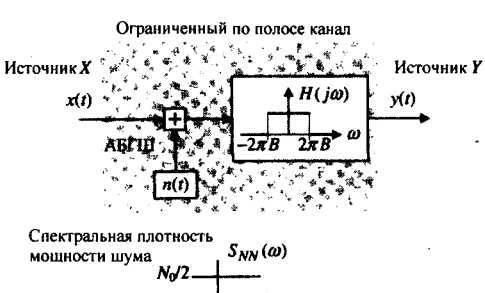


Рис. 8.4. Модель ограниченного по полосе канала с АБГШ.

Теорема 8.2.1. *Пропускная способность* канала (Хартлей – Шеннон).

Пропускная способность идеального канала с шириной полосы пропускания B и аддитивным белым гауссовским шумом мощности $N = N_0 B$ равна

$$\frac{C}{\text{бит/сек}} = B \log_2(1 + \frac{S}{N}). \tag{8.11}$$

Здесь, как и ранее, S – мощность сигнала в полосе пропускания канала. Размерность C – бит/сек.

Для интерпретации пропускной способности непрерывного канала C, рассмотрим выражение (8.11) при ширине полосы пропускания, равной 1 Гц (рис. 8.5). Можно отметить асимптотически линейный характер поведения функции $C/B = \log_2 f(S/N)$ (рис. 8.5 выполнен в полулогарифмическом масштабе).

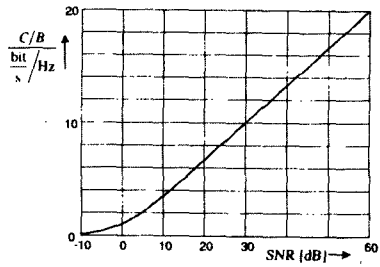


Рис. 8.5. Пропускная способность на 1 Гц полосы пропускания как функция отношения сигнал/шум.

Пусть задано отношение сигнал/шум (Signal to Noise Ratio – SNR) больше 0 дБ. Из формулы (8.11) следует, что удвоение пропускной способности требует квадратичного увеличения отношения сигнал/шум SNR, т.е. квадрата мощности передатчика при постоянном шуме.

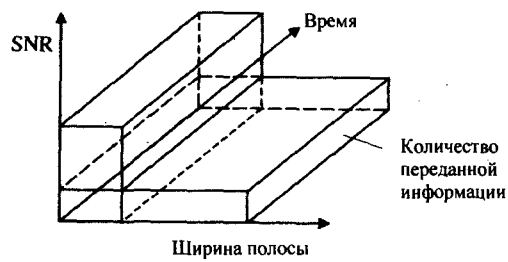


Рис. 8.6. Соотношение между шириной полосы пропускания, SNR, временем передачи и максимальным количеством переданной информации.

Пусть заданы отношение сигнал/шум, полоса пропускания B и время передачи t. Тогда можно определить пропускную способность канала и максимальный объем информации, который передется за заданный промежуток времени. Заметим так же, что при фиксированном объеме информации v бит, можно произвольно варьировать два из трех параметров (SNR, B, t), а третий параметр будет определяться из соотношения (8.11) и условия $v = C \cdot t$. Все вышесказанное иллюстрирует рис. 8.6.

Из рис. 8.6 видно, что объем передаваемой информации за определенное время можно представить как объем параллелепипеда в координатах SNR, B и t.

При внимательном рассмотрении соотношения (8.11), возникает важный вопрос: Как будет меняться пропускная способность канала при стремлении ширины полосы пропускания к бесконечности? Ответ не очевиден, так как расширение полосы влечет за собой увеличение мощности шума N. Мощность шума при постоянной спектральной плотности N_0 пропорциональна ширине полосы пропускания. Чем шире полоса, тем больше шум на выходе приемника (рис. 8.7).

$$N = N_0 B. \tag{8.12}$$

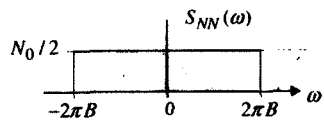


Рис. 8.7. Спектр белого гауссовского шума в ограниченной полосе

Исследуем предельный переход

$$\frac{C_\infty}{\text{бит/сек}} = \lim_{B\to\infty} \frac{C}{\text{бит/сек}} = \lim_{B\to\infty} B\ln(1+\frac{S}{N})\log_2 e. \tag{8.13}$$

Замечание. *Мы использовали натуральный логарифм для упрощения вычисления предела. Переход от размерности пропускной способности нат/сек к размерности бит/сек достигается умножением правой части выражения (8.13) на* $\log_2(e)$.

Подставляя (8.12) в предельный переход (8.13), получаем неопределенность

$$\lim_{B\to\infty} B\ln(1+\frac{S}{N_0 B}) = \lim_{B\to\infty} \frac{\ln(1+S/N_0 B)}{1/B}. \tag{8.14}$$

Раскрывая неопределенность по правилу Лапиталя, получаем конечное значение предела

$$\lim_{B\to\infty} B\ln(1+\frac{S}{N_0 B}) = \lim_{B\to\infty} \frac{(S/N_0)(-1/B^2)}{(1+S/N_0 B)(-1/B^2)} = \frac{S}{N_0}. \tag{8.15}$$

Выражение (8.15) совместно с (8.13) определяет *границу Шеннона.*

Пропускная способность непрерывного канала с АБГШ и неограниченной полосой пропускания равна

$$\frac{C_\infty}{\text{бит/сек}} \approx \frac{S}{N_0}\log_2 e \approx 1,44\frac{S}{N_0}. \tag{8.16}$$

В технике связи при передаче цифровой информации часто используется относительная величина – *энергия сигнала* E_b, приходящаяся на бит переданной информации. Так как максимальная скорость передачи информационных бит/сек определяется как

$$R_{\max} = C_\infty, \tag{8.17}$$

минимальная длительность передачи одного бита равна

$$T_b = \frac{1}{C_\infty}. \tag{8.18}$$

Энергия, затрачиваемая на передачу бита информации, определяется произведением $E_b = S \cdot T_b$. Используя (8.18) и (8.16), переходим к следующему утверждению.

Для передачи одного бита цифровой информации необходимо, чтобы отношение энергии на бит E_b к *спектральной плотности мощности* белого гауссовского шума N_0 равнялось, как минимум

$$\left.\frac{E_b}{N_0}\right|_{\min} = \frac{1}{\log_2 e} \approx 0,69 \cong -1,59 \text{ dB}. \tag{8.19}$$

Замечание. *Заметим, что в литературе спектральная плотность шума иногда определяется не как* $N_0/2$, *а как* N_0. *Это приводит к появлению дополнительного слагаемого в правой части (8.19), равного* $1,42$ *дБ.*

Наглядно связь между SNR, B и битовой скоростью R можно представить в виде диаграммы. Для этого, прежде всего, формально подставим R вместо C и N_0B вместо N в (8.11), разрешим равенство относительно $S/(N_0B)$ и получим

$$\frac{S}{N_0B} = 2^{R/B} - 1. \tag{8.20}$$

Устраним зависимость левой части от полосы пропускания B путем умножения обеих частей (8.20) на B/R, получим

$$\frac{S/R}{N_0} = \frac{B}{R}(2^{R/B} - 1). \tag{8.21}$$

Энергия E_b, затрачиваемая на передачу одного бита информации, определяется отношением $E_b = S/R$. Полученная зависимость между E_b/N_0 и B/R приведена на рис. 8.8.

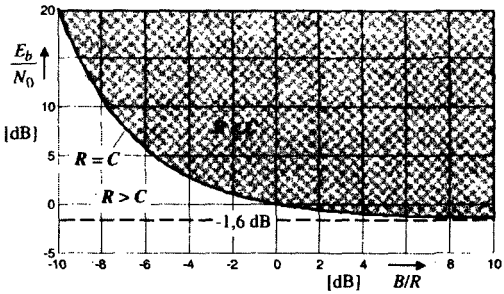


Рис. 8.8. Зависимость между шириной полосы пропускания и SNR при передаче информации.

Рис. 8.8 можно рассматривать как стандарт, который позволяет оценить эффективность выбранного метода кодирования в реальных системах связи. Пусть передача информации осуществляется при некоторых фиксированных значениях R, B и SNR. Этим значениям соответствует некоторая точка на диаграмме рис. 8.8. Согласно теореме кодирования для канала, скорость R не должна превышать пропускную способность канала C, поэтому наша точка всегда лежит выше кривой, задаваемой (8.21). Расстояние от граничной линии, которая соответствует значению $R = C$, позволяет оценить потенциальную возможность улучшения выбранного нами метода кодирования.

Замечание. *Современные цифровые системы связи используют прогрессивные методы кодирования, такие, например, как турбо - коды. Применение таких конструкций позволяет приблизиться к граничной кривой ценой некоторой задержки декодирования. При оценке соотношения цена - эффективность, учитываются многие дополнительные факторы, поэтому оптимальность системы, в смысле приближения скорости к пропускной способности канала, иногда отходит на второй план.*

8.3. Примеры

Пример: Видеотелефония.

Рассмотрим возможность передачи изображений между абонентами телефонной сети стандарта ISDN. В этой системе для передачи двумерных изображений используется канал со скоростью 64 кбит/сек, поэтому, необходимо, чтобы поток видеоинформации был сжат до этой величины. При этом, алгоритм сжатия должен устранять как избыточность с точки зрения теории информации, так и детали изображения, не существенные для абонента или многократно повторяющиеся. Одной из таких деталей может быть, например, не меняющийся длительное время второй план изображения и т.д. В дальнейшем, мы рассмотрим пример идеального сжатия видеосигнала.

При вычислении необходимой скорости передачи данных, мы будем исходить из стандарта QCIF (Quarter Common Intermediate Format).

В этом стандарте для передачи цветных изображений используется один сигнал яркости (Y) и два сигнала цветности (U, V). Параметры этих сигналов приведены в таблице 8.2.[1]

Таблица 8.2. Пример сигналов для передачи изображений.

	Количество точек в строке	Количество строк в кадре	Количество бит на точку	Частота смены кадров
Сигнал яркости (Y)	176	144 (120)	8	5...15 Гц
Сигнал цветности (U, V)	88	72(66)	8	5...15 Гц

Из табл. 8.2 следует, что отображение цветного изображения на экране достигается с помощью четырехточечных матриц, каждую из которых глаз воспринимает как одну точку. Внутри матрицы две

[1]Видеосигнал в стандартах QCIF PAL и SECAM содержит – 288 строк и 352 точки в строке для сигнала яркости (Y) и два сигнала цветности (U, V) по 176 строк и 144 точки на каждый, для видеосигнала в стандарте QCIF NTSC - 240 строк по 352 точки в строке (Y) и, соответственно, 120 строк и 144 точки (U, V). - *Прим. перев.*

точки принадлежат сигналу одной цветности, а две – другому. Яркость каждой точки определяется сигналом Y, который квантуется на $2^8 = 256$ уровней. Будем считать, что смена кадров производится с частотой 10 Гц. При быстром движении камеры и относительно медленной смене кадров, на экране возникают, так называемые, размывания изображения.

1. Найдите информационый поток в видеоканале при условии, что частота смены кадров равна 10 Гц. Будем исходить из следующей упрощенной модели: проквантованные видеосигналы имеют равномерное распределение вероятности во времени и пространстве и взаимно независимы.

2. Найдите минимальную ширину полосы пропускания идеального телевизионного приемника, если требуемое для удовлетворительного качества изображения отношение сигнал/шум составляет 30 дБ.

Решение.

1. Из табл. 8.2 следует, что число символов, необходимых для передачи кадра, равно

$$\frac{N_S}{\text{символ}} = 176 \cdot 144 + 2 \cdot 88 \cdot 72 = 38016. \tag{8.22}$$

Так как частота смены кадров равна 10 Гц, скорость передачи символов составляет

$$r_s = N_s \cdot 10 \text{ Гц} = 3800160 \text{ символ/сек}. \tag{8.23}$$

Учитывая, что вероятности $2^8 = 256$ возможных квантованных значений амплитуд сигналов Y, U, V распределены равномерно, информация каждого символа равна

$$I_s = -\log_2 \frac{1}{2^8} \frac{\text{бит}}{\text{символ}} = 8 \text{ бит/символ}. \tag{8.24}$$

Таким образом, искомый информационный поток составляет

$$I = r_s I_s = 380160 \frac{\text{символ}}{\text{сек}} \cdot 8 \frac{\text{бит}}{\text{символ}} = 3,04128 \text{ Мбит/сек}. \tag{8.25}$$

Замечание. *Заметим, что информация символа равна длине двоичной записи символа только в случае равномерного распределения*

вероятностей символов. В реальных каналах передачи цифровых видеосигналов информация одного символа существенно меньше 8 бит. Сравнение (8.25) с ISDN-B каналом, по которому информация передается со скоростью 64 кбит/сек показывает, что фактор сжатия в этом канале должен быть равен, приблизительно, 50.

2. Для передачи информационного потока 3,04128 Мбит/сек должен использоваться канал с большей пропускной способностью. Из шенноновской формулы (8.11) получаем

$$B > \frac{I}{\log_2(1+\frac{S}{N})} = \frac{3,04128 \cdot 10^6}{\log_2(1+10^3)} \frac{1}{s} \approx 305 \text{ кГц}. \qquad (8.26)$$

Пример: Импульсно-кодовая модуляция.

Аналоговый сигнал с верхней спектральной частотой 4 кГц подвергается дискретизации во времени с частотой отсчетов, в 1,25 раз превышающей минимально необходимую для восстановления сигнала. Каждый отсчет квантуется на 256 уровней символами, содержащими 8 бит. Предполагается, что символы независимы и имеют равномерное распределение вероятностей.

1. Найдите информационный поток оцифрованного источника.

2. Может ли этот информационный поток быть передан по каналу с АБГШ с полосой $B = 10$ кГц и SNR$-$20 дБ с пренебрежимо малой вероятностью ошибки?

3. Каково должно быть минимальное SNR, необходимое для передачи информационного потока в полосе $B = 10$ кГц с пренебрежимо малой вероятностью ошибки?

4. Каково минимальное значение полосы пропускания B, обеспечивающее передачу информации по каналу с АБГШ при SNR$-$20 дБ с пренебрежимо малой вероятностью ошибки?

Решение.

1. Частота отсчетов равна

$$f_S = 1,25 \cdot 4 \cdot 2 = 10 \text{ кГц}. \qquad (8.27)$$

Скорость передачи символов равна

$$r_S = 10^4 \text{ сим/сек}. \qquad (8.28)$$

Информация одного символа –

$$I_S = 8 \text{ бит}. \tag{8.29}$$

Информационный поток –

$$I = r_S I_S = 80 \text{ кбит/сек}. \tag{8.30}$$

2. Пропускная способность канала равна

$$C = 10 \text{ кГц} \cdot \log_2(1 + 10^2) \approx 66 \text{ кбит/сек}, \tag{8.31}$$

таким образом, $C < I$. Безошибочная передача невозможна.

3. Минимально необходимое отношение сигнал/шум определяется из (8.11)

$$10 \text{ кГц} \cdot \log_2(1 + \frac{S}{N}) = 80 \text{ кбит/сек} \tag{8.32}$$

и равно

$$\left.\frac{S}{N}\right|_{\min} = 255 \approx 24,1 \text{ дБ}. \tag{8.33}$$

4. Минимально необходимое значение полосы пропускания вычисляется, исходя из (8.11)

$$B \log_2(1 + 10^2) = 80 \text{ кбит/сек}. \tag{8.34}$$

Таким образом, получаем

$$B|_{\min} = 12 \text{ кГц}. \tag{8.35}$$

Пример: Телефонный канал.

Стандартный аналоговый телефонный канал имеет полосу пропускания от 300 Гц до 3,4 кГц. Будем предполагать, что телефонный узел связи обеспечивает на выходе заданное отношение сигнал/шум.

Каждый абонент связан с коммутатором парой проводов. Затухание сигнала в полосе до 4 кГц составляет, приблизительно, 2 дБ/км.

Постройте график пропускной способности канала, как функцию удаленности абонента от узла связи.

Решение.

Подставим в выражение для пропускной способности (8.11) значение отношения сигнал/шум как функции расстояния l. Будем считать ширину полосы пропускания 4 кГц., тогда

$$C = 4 \log_2(1 + 10^{0,1[SNR - 2l/km]}) \text{ кбит/сек}. \tag{8.36}$$

Графики зависимости пропускной способности от расстояния приведены на рис. 8.9.

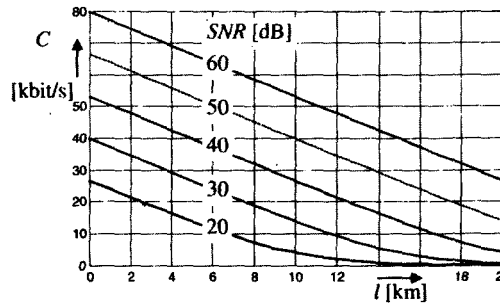


Рис. 8.9. Зависимость пропускной способности от дальности подключения.

Значение пропускной способности C как функции l, соответствующее отношению сигнал/шум 60 дБ, падает почти линейно с ростом $0 < l < 20$ км с 80 кбит/сек до 26 кбит/сек. Подобным же образом ведут себя кривые, соответствующие меньшим начальным отношениям сигнал/шум, однако, с уменьшением SNR, свойство линейности утрачивается.

Замечание. *Статистика показывает, что 99,5 % абонентов, подключенных к центральному узлу связи, были удалены от него не более чем на 8 км. Фактически, пропускная способность каналов связи существенно выше, чем приведенная в примере. Это объясняется тем фактом, что ширина полосы пропускания пары телефонных проводов во много раз выше 4 кГц.*

Современные методы передачи информации по проводным линиям связи (xDSL) позволяют реализовать скорость передачи данных до нескольких Мбит/сек. Например, в стандарте ADSL, передача информации осуществляется со скоростью выше 64 кбит/сек и используется полоса частот от 26...138 кГц. При соединении абонентов в сеть со скоростью 6 Мбит/сек, полоса частот составляет 26...1104 кГц.

Пример: Телеграфия.

В телеграфии для передачи сообщений используются два символа – точка и тире. Отдельные символы разделены паузами. Будем

считать, что длительность тире составляет 1 сек., а длительность точки и паузы – 1/3 сек.

Найдите средний информационный поток телеграфного источника.

Решение.

Из таблицы 3.1 следует, что отношение частот встречаемости точек и тире равно, приблизительно, 2:1 (с учетом вероятностей букв алфавита в тексте). Значит, вероятности появления точки и тире связаны соотношением

$$P_{\text{точка}} = 2P_{\text{тире}} \text{ и } P_{\text{точка}} + P_{\text{тире}} = 1. \tag{8.37}$$

Следовательно,

$$P_{\text{точка}} = 2/3 \text{ и } P_{\text{тире}} = 1/3. \tag{8.38}$$

Энтропия двоичного телеграфного источника

$$\frac{H(x)}{\text{бит}} = -P_{\text{точка}} \log_2 P_{\text{точка}} - P_{\text{тире}} \log_2 P_{\text{тире}} = 0,92. \tag{8.39}$$

Средняя длительность одного символа, включая последующую паузу, составляет

$$\overline{T} = 2/3 t_{\text{точка}} + 1/3 t_{\text{тире}} + t_{\text{пауза}} = \tag{8.40}$$
$$= 2/3 \cdot 1/3 \text{ сек} + 1/3 \text{ сек} + 1/3 \text{ сек} = 8/9 \text{ сек}.$$

Таким образом, средний информационный поток равен

$$I = \frac{H(X)}{\overline{T}} = 1,04 \text{ бит/сек}. \tag{8.41}$$

Часть II

Помехоустойчивое кодирование

ГЛАВА I

ВВЕДЕНИЕ

Структурная схема процесса цифровой связи приведена на рис. 1.1. Источник выдает сообщение, которое в общем виде представляет собой некоторый электрический сигнал. Аналоговый сигнал преобразуется в цифровую форму, удобную для дальнейшей обработки.

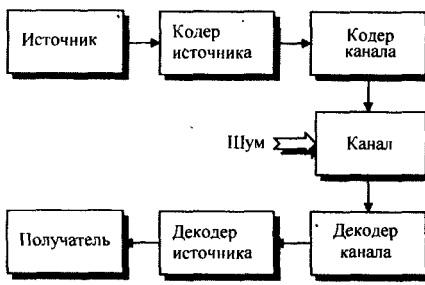


Рис. 1.1. Модель передачи информации.

Заметим, что аналого - цифровое преобразование, как правило, ограничивает полосу сигнала. Далее производится сжатие информации (кодирование источника, см. часть I). Кодирование источника удаляет несущественную информацию и минимизирует, а иногда и полностью устраняет избыточность сообщения. Таким образом, кодирование источника снижает затраты на хранение и передачу информации. Далее сообщение должно быть передано по зашумленному каналу. Для того, чтобы в дальнейшем сообщение могло быть доведено до потребителя в неискаженном виде, перед передачей в канал производится помехоустойчивое кодирование информации (кодирование канала). На приемном конце информация, поступающая из канала подвергается обратным преобразованиям. Декодер канала исправляет ошибки в принятом слове, а декодер источника преобразует исправленное слово в форму, удобную потребителю.

В учебной литературе основное внимание уделяется не методам устранения ошибок в канале, а оптимальным процедурам помехоустойчивого кодирования и декодирования, позволяющим обнару-

живать и исправлять эти ошибки. При этом, говоря о кодах, контролирующих ошибки, различают две стратегии их использования: с непосредственным *исправлением ошибок за счет избыточности* (Forward Error Correction – FEC) и с обнаружением ошибок с последующими *запросами на повторную передачу ошибочно принятой информации* (Automatic Repeat Request – ARQ).

При проектировании реальных систем связи, обычно, сочетают выбор канала с выбором конкретных методов помехоустойчивого кодирования. При этом, стремятся оптимизировать соотношение между затратами и качеством передачи информации. Под качеством, как правило, понимают среднюю долю ошибочных бит (Bit Error Rate – BER), которая определяется, как *средняя вероятность ошибки* одного бита передаваемой информации.

Первым шагом разработки системы связи является выбор конструкций передатчика и приемника, а также среды, в которой будет осуществляться передача данных. С точки зрения теории информации, этим самым мы выбираем один из каналов, среди всех реально существующих. Опыт показывает, что на первом шаге разработчики системы связи очень часто сталкиваются с, так называемым, «эффектом насыщения», который показан на рис. 1.2. Используя стандартную технику передачи данных, можно получить некоторое гарантированное качество связи, например, для мобильных каналов $BER \approx 10^{-2}$, для проводных каналов $BER \approx 10^{-5}$, для волоконнооптических каналов $BER \approx 10^{-12}$. При дальнейшем улучшении качества связи средствами аналоговой техники, затраты на реализацию резко возрастают.

Рис. 1.2. Зависимость между затратами и качеством в системах передачи информации.

Здесь на помощь приходит цифровая техника и помехоустойчивое кодирование. Путем применения оптимального кодирования,

можно обеспечить как угодно малую BER (при условии $R < C$). При выборе методов кодирования и декодирования, руководствуются многими факторами, взаимосвязь которых показана на рис. 1.3. В общую сложность входят аппаратурные и программные затраты на реализацию кодера и декодера, цены специализированных микросхем и микропроцессоров, стоимость памяти для хранения информации и т.д. Интенсивность потока данных включает в себя передачу полезной информации, проверочных разрядов, а также передачу запросов и повторений по этим запросам отдельных блоков сообщений.

Рис. 1.3. Взаимосвязь между параметрами кодовых конструкций.

Методы помехоустойчивого кодирования довольно многообразны. В этой книге мы ограничимся лишь описанием наиболее важных классов блоковых и сверточных кодов. При этом, основное внимание уделяется циклическим кодам из-за простоты их реализации. Циклические коды образуют подмножество линейных блоковых кодов, поэтому, наш вводный курс мы начнем с описания структуры и общих свойств линейных блоковых кодов (подробнее см., например, [5]).

ГЛАВА 2

ЛИНЕЙНЫЕ БЛОКОВЫЕ КОДЫ

2.1. Помехоустойчивое кодирование

Реальные системы передачи данных не совершенны. Применяя информационную технику, мы должны учитывать возможность возникновения ошибок (вероятность ошибок) при передаче и хранении информации. Это в первую очередь относится к

- хранению информации на носителях с высокой плотностью записи (магнитные носители, $CD-ROM$, DVD).
- передаче данных при ограниченной мощности сигнала (спутниковая и мобильная связь)
- передаче информаций по сильно зашумленным каналам (мобильная связь, высокоскоростные проводные линии связи)
- каналам связи с повышенными требованиями к надежности информации (вычислительные сети, линии передачи со сжатием данных)

Во всех вышеперечисленных случаях используются коды, контролирующие ошибки. Теория помехоустойчивого кодирования для каждого конкретного канала позволяет выбрать наиболее эффективный метод обнаружения и исправления ошибок. Существуют два взаимодополняющих метода борьбы с помехами.

- Кодирование для исправления ошибок – приемник обнаруживает и исправляет ошибки;
- Кодирование для обнаружения ошибок – приемник распознает ошибки и, в случае необходимости, производит запрос на повторную передачу ошибочного блока.

Последний метод предполагает наличие канала обратной связи и находит свое применение в каналах с достаточно малой вероятностью ошибки в случае, если эту вероятность ошибки необходимо

еще понизить. Такая ситуация часто возникает в вычислительных сетях и в интернете. Типичное значение вероятности ошибки на бит без кодирования в вычислительных сетях составляет 10^{-6}. Использование простейших кодов с небольшой избыточностью позволяет достигнуть вероятности 10^{-9} и ниже.

Замечание. *Требование к вероятности ошибки* 10^{-9} *не является чрезмерно завышенным. В вычислительных сетях, например, может возникнуть обрыв связи в результате повреждения оптоволокна при производстве земляных работ, небрежного подключения кабеля к модему и т.д. Такой обрыв должен быть быстро обнаружен декодером, который в случае резкого возрастания частоты переспросов выдает сигнал обрыва связи.*

В последующих разделах идеи помехоустойчивого кодирования будут подробно объяснены на примерах линейных блоковых кодов. Здесь же мы рассмотрим простейшую модель передачи данных с использованием помехоустойчивого кодирования (рис. 2.1).

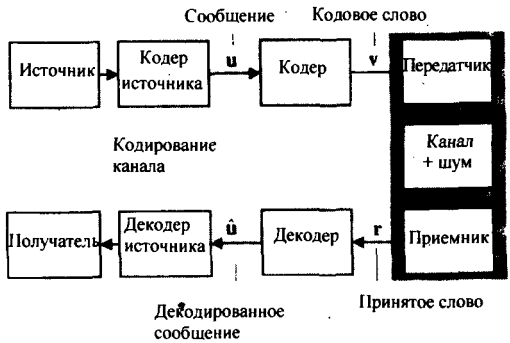


Рис. 2.1. Модель канала связи с кодированием.

Будем исходить из того, что при передаче информации используется блоковый код Хэмминга[1], структура которого будет подробно раскрыта в дальнейшем. Сейчас мы ограничимся его табличным описанием. Пусть кодер источника последовательно выдает *информационные слова* фиксированной длины. Кодер канала заменяет каждое информационное слово $\mathbf{u}$ *кодовым словом* $\mathbf{v}$ в соответствии с табл. 2.1.

Передатчик генерирует сигналы, соответствующие кодовому слову $\mathbf{v}$ и посылает их в канал. Приемник производит обратное пре-

[1]Ричард В. Хэмминг: 1915/1998, американский математик.

Таблица 2.1. Кодовая таблица (7,4)-кода Хэмминга.

Информационное слово	Кодовое слово	Информационное слово	Кодовое слово
0000	000 0000	0001	101 0001
1000	110 1000	1001	011 1001
0100	011 0100	0101	110 0101
1100	101 1100	1101	000 1101
0010	111 0010	0011	010 0011
1010	001 1010	1011	100 1011
0110	100 0110	0111	001 0111
1110	010 1110	1111	111 1111

образование, в результате которого на декодер поступает двоичное *принятое слово* $\mathbf{r}$.

Декодер сравнивает принятое слово $\mathbf{r}$ со всеми кодовыми словами табл. 2.1. Если слово $\mathbf{r}$ совпадает с одним из кодовых слов, то соответствующее информационное слово $\widetilde{\mathbf{u}}$ выдается потребителю. Если $\mathbf{r}$ отличается от всех кодовых слов, то в канале произошла обнаруживаемая ошибка.[2]

Из всего вышесказанного уже можно сделать два важных вывода:

- Если в процессе передачи по зашумленному каналу кодовое слово отобразится в другое кодовое слово, не совпадающее с переданным, то происходит необнаружимая ошибка. Назовем ее остаточной *ошибкой декодирования*.

- «Хорошие коды» обладают некоторой математической структурой, которая позволяет эффективно распознать, а в некоторых случаях и исправлять ошибки, возникающие при передаче информации по каналу связи.

[2]Из структуры кода Хэмминга следует одно интересное свойство, которое может быть проверенно простым перебором:

Для любого произвольного вектора $\mathbf{r}$ существует ближайшее кодовое слово, которое или полностью совпадает с $\mathbf{r}$ или отличается от него только в одном двоичном разряде. Таким образом, если в векторе $\mathbf{v}$ при передаче по каналу произошла только одна ошибка, она всегда может быть исправлена в процессе декодирования. – *Прим. перев.*

2.2. Порождающая матрица

Важное семейство кодов образуют *линейные двоичные блоковые коды*. Эти коды замечательны тем, что представляя информационные и кодовые слова в форме двоичных векторов, мы можем описать процессы кодирования и декодирования с помощью аппарата линейной алгебры, при этом, компонентами вводимых векторов и матриц являются символы 0 и 1. Операции над двоичными компонентами производятся по привычным правилам двоичной арифметики, так называемой, *арифметики по модулю 2* (Табл. 2.2).

Таблица 2.2. Арифметика по модулю 2.

Сложение			Умножение		
$\oplus$	0	1	$\odot$	0	1
0	0	1	0	0	0
1	1	0	1	0	1

Замечание. *С математической точки зрения, определив операции с двоичными символами согласно таблице 2.2, мы построили поле Галуа характеристики 2 первого порядка $GF(2)$. Общая теория полей Галуа позволяет строить поля характеристики p порядка m - $GF(p^m)$, где p - простое, m - любое конечное целое. Переход к расширенным полям $GF(p^m)$ дает возможность конструировать коды, обладающие рядом новых свойств, полезных по сравнению с двоичными кодами.*

В частности, коды Рида-Соломона с символами из $GF(2^m)$, $m > 2$ с успехом применяются для защиты информации в аудио, CD проигрывателях (Полям Галуа и кодам Рида-Соломона посвящена глава 5 данной книги).

Кодер двоичого блокового (n, k)-кода отображает множество 2^k возможных двоичных информационных слов в множество 2^k n-мерных кодовых слов (в теории кодирования между этими множествами всегда существует взаимно однозначное соответствие) (см. рис. 2.2).

Вместо k бит информационного вектора в канал передается n бит кодового вектора. В этом случае говорят об *избыточном кодировании* со *скоростью*

$$R = \frac{k}{m}. \tag{2.1}$$

Чем ниже скорость, тем больше избыточность кода и тем большими

возможностями для защиты от ошибок он обладает (здесь, однако, надо учитывать, что с увеличением избыточности, затраты на передачу информацию также возрастают).

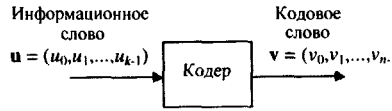


Рис. 2.2. Кодер блокового (n, k)-кода.

Кодирование линейного блокового (n, k)-кода задается *порождающей матрицей* $\mathbf{G}_{k\times n}$. В рассмотренном выше (7,4)-коде Хэмминга порождающая матрица имеет вид

$$\mathbf{G}_{4\times 7} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.2}$$

Таким образом, *кодовое слово* $\mathbf{v}$ и информационное слово $\mathbf{u}$ связаны соотношением

$$\mathbf{v} = \mathbf{u} \odot \mathbf{G}. \tag{2.3}$$

Например, информационный вектор $\mathbf{u} = (1010)$ отображается в кодовый вектор

$$\mathbf{v} = (1010) \odot \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = (0011010). \tag{2.4}$$

Первое, что сразу же бросается в глаза из табл. 2.1, это совпадение последних четырех разрядов кодовых слов с информационными векторами. Такой код относится к семейству систематических кодов.

Коды, в которых информационное слово может быть непосредственно выделено из соответствующего ему кодового вектора, называются *систематическими*.

Порождающую матрицу любого *систематического кода* всегда можно путем перестановки столбцов привести к виду

$$\mathbf{G}_{k\times n} = (\mathbf{P}_{k\times(n-k)} \quad \mathbf{I}_k), \tag{2.5}$$

где нижние индексы обозначают размерность матрицы, а $\mathbf{I}_k$ - единичная матрица размерности $k \times k$.

Замечание. *В литературе часто единичная матрица ставится на первое место. Заметим, что перестановка столбцов матрицы не оказывает никакого влияния на корректирующую способность кода.*

Таким образом, в *кодовом векторе* систематического кода всегда можно выделить *информационные* и *проверочные* символы

$$\mathbf{v} = (\underbrace{v_0 \quad \dots \quad v_{n-k-1}}_{n-k \text{ проверочных символов}} \qquad \underbrace{v_{n-k} \quad \dots \quad v_{n-1}}_{k \text{ информационных символов}}). \quad (2.6)$$

Роль проверочных символов и их использование будут подробно разъяснены в следующих разделах.

2.3. Синдромное декодирование

Задача декодера заключается в том, чтобы используя структуру кода, по принятому слову $\mathbf{r}$, восстановить переданный информационный вектор.

Для рассмотренного выше (7, 4)-кода Хэмминга можно предложить следующий алгоритм обнаружения ошибок. Так как рассматриваемый код является систематическим, выразим каждый из трех проверочных символов через символы информационного вектора $v_0 = v_3 \oplus v_5 \oplus v_6$, $v_1 = v_3 \oplus v_4 \oplus v_5$ и $v_2 = v_4 \oplus v_5 \oplus v_6$. Если в канале произошла ошибка, то в принятом векторе $\mathbf{r}$ хотя бы одно из равенств не будет выполняться. Запишем полученные проверочные соотношения в виде системы уравнений для компонент вектора $\mathbf{r}$:

$$\begin{array}{ccccccccc} r_0\oplus & & & r_3\oplus & & r_5\oplus & r_6\oplus & = & s_0 \\ & r_1\oplus & & r_3\oplus & r_4\oplus & r_5\oplus & & = & s_1 \\ & & r_2\oplus & & r_4\oplus & r_5\oplus & r_6\ominus & = & s_2. \end{array} \quad (2.7)$$

Таким образом, из первых трех столбцов порождающей матрицы $\mathbf{G}$ (2.2), мы получили систему трех проверочных уравнений, в которой операция $\oplus$ производится по правилам арифметики по модулю 2 (см. табл. 2.2). Если в полученной системе уравнений хотя бы одна из компонент $\{s_0, s_1, s_2\}$ не равна нулю, то в канале произошла ошибка.

Запишем систему проверочных уравнений в общем виде. Для любого систематического кода с порождающей матрицей (2.5), проверочная матрица определяется как

$$\mathbf{H}_{(n-k)\times n} = (\mathbf{I}_{n-k} \quad \mathbf{P}^T_{k\times(n-k)}), \quad (2.8)$$

где $\mathbf{H}_{k\times(n-k)}$ – транспонированная матрица, т. е. матрица размерности $k \times (n-k)$, получаемая из $\mathbf{P}_{k\times(n-k)}$ путем замены строк матрицы на ее столбцы. Тогда систему *проверочных уравнений* можно записать в виде

$$\mathbf{s} = \mathbf{r} \odot \mathbf{H}^T. \tag{2.9}$$

Вектор $\mathbf{s}$ принято называть *синдромом*. Таким образом, ошибка будет обнаружена, если хотя бы одна из компонент $\mathbf{s}$ не равна нулю.

Равенство (2.9) можно переписать в виде

$$\mathbf{s} = \mathbf{r} \odot \begin{pmatrix} \mathbf{I}_{n-k} \\ \mathbf{P}_{k\times(n-k)} \end{pmatrix}. \tag{2.10}$$

Замечание. *В медицине термин синдром используется для обозначения сочетания признаков, характеризующих определенное болезненное состояние организма.*

Пример: Синдромное декодирование (7, 4)-кода Хэмминга.

Используя (2.5) и (2.8), построим проверочную матрицу из порождающей матрицы кода Хэмминга (2.2). Она имеет вид

$$\mathbf{H}_{3\times7} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}. \tag{2.11}$$

При передаче информационного слова $\mathbf{u} = (1010)$ по каналу без шума $\mathbf{r} = \mathbf{v} = (0011010)$. Можем убедиться, что в этом случае синдром равен

$$\mathbf{s} = (0011010) \odot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} = (000). \tag{2.12}$$

Если, например, в кодовом слове произошла одиночная ошибка на

Таблица 2.3. Таблица синдромов однократной ошибки (7,4)-кода Хэмлинга.

Кодовое слово $\mathbf{r}$	r_0	r_1	r_2	r_3	r_4	r_5	r_6
Синдром $\mathbf{s}$	100	010	001	110	011	111	101

четвертой позиции ($\mathbf{r} = (0010010)$), то синдромом является четвертая строка транспонированной проверочной матрицы

$$\mathbf{s} = (0010010) \odot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} = (110). \qquad (2.13)$$

Перебрав все возможные позиции одиночной ошибки, получим полную таблицу синдромов однократных ошибок – таблицу соответствий номера ошибочного разряда получающемуся при этом синдрому (табл. 2.3). Можно заметить, что ошибке в i-ой позиции кодового слова соответствует синдром, образованный i-ым столбцом матрицы $\mathbf{H}$. Так как все столбцы матрицы различны, мы можем с помощью таблицы 2.3 исправлять одиночную ошибку, вносимую каналом.

Обобщим приведенные выше рассуждения, используя аппарат линейной алгебры.

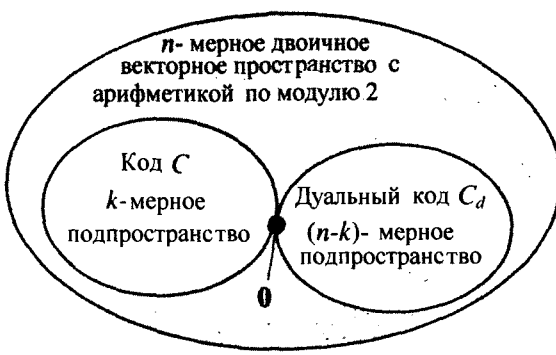


Рис. 2.3. Структура кодовых векторных пространств.

Исходным материалом для построения кодовых конструкций служит n-мерное двоичное *векторное пространство*, в котором заданы операции арифметики по модулю 2 (табл. 2.2).

В него вложено k-мерное линейное пространство, содержащее 2^k кодовых слов (рис. 2.3). Код C образуется с помощью 2^k комбинаций k линейно независимых базисных векторов $\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}$. Иногда говорят, что код C «натянут» на векторы $\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}$. Эти векторы образуют строки *порождающей матрицы* кода C

$$\mathbf{G}_{k\times n} = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \vdots \\ \mathbf{g}_k \end{pmatrix} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k,1} & g_{k,2} & \cdots & g_{k,n} \end{pmatrix} = (\mathbf{P}_{k\times(n-k)} \ \mathbf{I}_k). \qquad (2.14)$$

Заметим, что порождающая матрица может быть разложена на матрицу $\mathbf{P}$ и единичную матрицу $\mathbf{I}$ только в случае систематических кодов.

Для кода C существует дуальный код C_d такой, что скалярное произведение любой пары векторов, один из которых принадлежит пространству C, а другой – пространству C_d, всегда равно нулю. Это значит, что векторы кода C_d ортогональны векторам кода C. С другой стороны, если некоторый вектор ортогонален всем векторам кода C, то он принадлежит коду C_d и наоборот.

Дуальное векторное подпространство «натянуто» на $n-k$ линейно независимые базисные векторы $\{\mathbf{h}_1, \ldots, \mathbf{h}_{n-k}\}$. Эти векторы образуют строки *проверочной матрицы*

$$\mathbf{H}_{(n-k)\times n} = \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \vdots \\ \mathbf{h}_{n-k} \end{pmatrix} = \qquad (2.15)$$

$$= \begin{pmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,n} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{(n-k),1} & h_{(n-k),2} & \cdots & h_{(n-k),n} \end{pmatrix} = (\mathbf{I}_{n-k} \ \ \mathbf{P}^T_{k\times(n-k)}),$$

причем, правая часть равенства справедлива только для систематических кодов.

При синдромном кодировании приемник использует свойство ортогональности кодов

$$\mathbf{G} \odot \mathbf{H}^T = \mathbf{0}. \qquad (2.16)$$

Таким образом, для каждого кодового слова $\mathbf{v} \in C$ справедливо

$$\mathbf{s} = \mathbf{v} \odot \mathbf{H}^T = \mathbf{0}. \tag{2.17}$$

Каждому принятому слову, не принадлежащему коду, соответствует отличный от нуля синдром

$$\mathbf{s} = \mathbf{r} \odot \mathbf{H}^T \neq \mathbf{0} \text{ для } \mathbf{r} \notin C. \tag{2.18}$$

Проведем анализ синдромного декодирования на уровне двоичных компонент (рис. 2.4). В канале производится покомпонентное сложение по модулю 2 кодового вектора $\mathbf{v}$ с двоичным вектором ошибки $\mathbf{e}$. Таким образом, если i-ая компонента вектора i равна «1», то в i-ой компоненте кодового вектора $\mathbf{v}$ возникает ошибка.

В рассмотренном выше примере, единичной ошибке в четвертом разряде соответствует вектор $\mathbf{e} = (0001000)$.

Замечание. *Операция сложения по модулю 2 двоичных символов эквивалентна операции «исключающего или» XOR.*

Рис. 2.4. Модель передачи информации на двоичном уровне.

В силу свойств линейности и ортогональности векторов имеем

$$\mathbf{s} = \mathbf{r} \odot \mathbf{H}^T = (\mathbf{v} \oplus \mathbf{e}) \odot \mathbf{H}^T = \mathbf{e} \odot \mathbf{H}^T. \tag{2.19}$$

Последнее равенство является основой синдромного декодирования. В процессе декодирования могут возникнуть следующие ситуации:

Случай 1: $\mathbf{s} = \mathbf{0} \Leftrightarrow \mathbf{e} \in C$

Случай 1.1: $\mathbf{e} = \mathbf{0}$ – безошибочная передача информации;

Случай 1.2: $\mathbf{e} \neq \mathbf{0}$ – передача информации с неисправляемой ошибкой;

Случай 2: $\mathbf{s} \neq \mathbf{0} \Leftrightarrow \mathbf{e} \notin C$ — ошибка будет обнаружена при декодировании.

Ясно, что в первом случае, декодер всегда выдает принятое слово $\mathbf{r}$ потребителю, при этом существует некоторая вероятность неисправления ошибки. Во втором случае возможны два режима работы декодера.

- Распознавание ошибок. Декодер всегда определяет наличие ошибки в принятом векторе $\mathbf{r}$. В зависимости от требований потребителя, принятое информационное слово или «стирается», или производится запрос на его повторную передачу.
- Коррекция ошибок. Корректирующая способность декодера может быть пояснена на примере (7,4)-кода Хэмминга, рассмотренного выше.

Таблица 2.3 показывает, что в случае одночной ошибки, ее позиция однозначно определяется по синдрому, и, таким образом, однократная ошибка всегда исправляется. Ошибки же большей кратности декодер всегда исправляет как одиночные, и потребителю выдается ошибочное информационное слово. Пусть, например, передается кодовое слово $\mathbf{v} = (0011010)$, соответствующее информационному вектору $\mathbf{u} = (1010)$, и вектор ошибки равен $\mathbf{e} = (1100000)$, т.е. в канале произошла двукратная ошибка. Тогда для принятого слова $\mathbf{r} = (1111010)$ синдром равен $\mathbf{s} = (110)$. Из табл. 2.3 следует, что этот синдром соответствует четвертой ошибочной компоненте вектора $\mathbf{r}$. Таким образом, потребителю будет выдан вектор $\hat{\mathbf{u}} = (0010)$.

Декодер может выдавать потребителю ошибочное информационное слово тогда и только тогда, когда в канале произошли *необнаружимые ошибки*, или кратность канальной ошибки превышает корректирующую способность кода. Из рассмотренного выше примера следует, что эффективность конкретного кода зависит от области его применения и, в особенности, от канала связи. Если мы передаем информацию по каналу с аддитивным белым гауссовским шумом (АБГШ), то ошибки в кодовом слове независимы. Если при этом отношение сигнал/шум достаточно велико, то вероятность одиночной ошибки во много раз превышает вероятность ошибок высших кратностей, поэтому, использование в таком канале кода Хэмминга с исправлением однократной ошибки может оказаться весьма эффективным. С другой стороны, в каналах, где преобладают многократные

ошибки (например, в каналах с замираниями), исправление одиночных ошибок лишено смысла. При практическом выборе конкретного помехоустойчивого кода необходимо также учитывать скорость его декодирования и сложность технической реализации.

2.4. Свойства линейных блоковых кодов

В предыдущих разделах на примере (7,4)-кода Хэмминга были показаны основные свойства линейных блоковых кодов и приведен метод синдромного декодирования. Теперь возникает следующий вопрос: Чем отличаются «хорошие» коды от «плохих» и как их искать? В следующих параграфах, раскрывая структуру линейных кодов более подробно, мы постараемся коротко сформулировать ответ на этот вопрос.

2.4.1. Расстояние Хэмминга и корректирующая способность

Для лучшего понимания процесса декодирования будем использовать геометрическое представление *векторного пространства*. На рис. 2.5 показан сектор n-мерного двоичного векторного пространства. В нем особое место занимает n-мерный кодовый вектор $\mathbf{0}$, который всегда принадлежит любому линейному коду (это следует из свойств линейных кодов). Здесь же показаны n-мерные кодовые векторы $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, обозначенные символами «о». Возможные принимаемые векторы $\mathbf{r}$ обозначены символами «×». Области декодирования кодовых слов $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ показаны как окружности с центрами в точках, соответствующих изображениям этих кодовых слов на плоскости. Это значит, что если принятый вектор $\mathbf{r}$ находится, например, внутри затемненной области (рис. 2.5), то декодер выдаст потребителю кодовое слово $\mathbf{v}_1$.[1]

Пусть в канал посылается слово $\mathbf{v}_1$. В результате искажения в канале могут иметь место три варианта приема и декодирования (рис. 2.5).

- В первом случае, вектор ошибки $\mathbf{e}_1$ отображает переданный вектор в точку, принадлежащую области декодирования $\mathbf{v}_1$. Декодер выдает потребителю переданное слово $\mathbf{v}_1$, исправляя при этом возникающие в канале ошибки.

[1] Представление n-мерных векторов точками на плоскости не должно смущать читателя, т.к. такое наглядное описание кодов позволяет только лучше понять их свойства и никакой другой цели перед собой не ставит. *Прим. перев.*

- Во втором случае, вектор $\mathbf{e}_2$ переводит переданный вектор $\mathbf{v}_1$ в область декодирования $\mathbf{v}_2$. Таким образом, потребителю выдается ошибочное слово $\mathbf{v}_2$ вместо $\mathbf{v}_1$. Тем не менее, ошибка распознается, так как принятый вектор $\mathbf{r}_2$ не является кодовым словом.

- В третьем случае, вектор ошибки $\mathbf{e}_3$ отображает переданное слово в ошибочное кодовое слово $\mathbf{v}_3$. Имеет место необнаруженная ошибка.

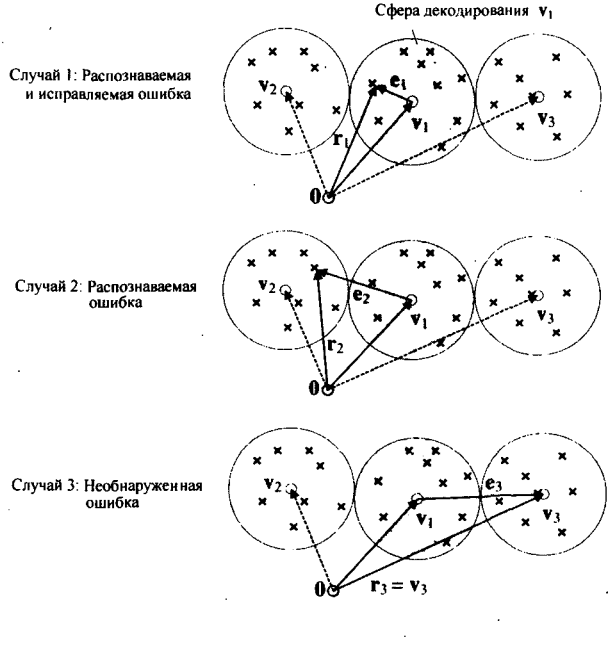


Рис. 2.5. Векторное пространство с кодовыми словами «○» и принятыми словами «×».

Из рисунка ясно, что корректирующая способность кода зависит от расстояния между кодовыми словами. Так как мы имеем дело с двоичными векторами и расстояние между ними определяется числом несовпадающих компонент, можно записать

$$d(\mathbf{v}_i, \mathbf{v}_j) = \sum_{l=0}^{n-1} v_{i,l} \oplus v_{j,l}. \qquad (2.20)$$

Расстояние, измеренное таким образом, называется *расстоянием Хэмминга*. Его также можно определить как число отличных от нуля компонент (вес Хэмминга) скалярной суммы векторов $\mathbf{v}_i$ и $\mathbf{v}_j$

$$d(\mathbf{v}_i, \mathbf{v}_j) = \omega_H(\mathbf{v}_i \oplus \mathbf{v}_j). \tag{2.21}$$

Пример: Синдромное декодирование (7,4)-кода Хэмминга.

В качестве примера найдем расстояние Хэмминга между кодовыми векторами $\mathbf{v}_1 = (1101000)$ и $\mathbf{v}_2 = (0110100)$ из таблицы 2.1.

Согласно (2.20) имеем

$$\begin{aligned} d(\mathbf{v}_1, \mathbf{v}_2) &= (1 \oplus 0) + (1 \oplus 1) + (0 \oplus 1) + (1 \oplus 1) + \\ &+ (0 \oplus 1) + (0 \oplus 0) + (0 \oplus 0) = 3 \end{aligned} \tag{2.22}$$

и, с другой стороны, определяя расстояние Хэмминга как (2.21), получаем

$$d(\mathbf{v}_1, \mathbf{v}_2) = \omega_H(\mathbf{v}_1 \oplus \mathbf{v}_2) = \omega[(1010100)] = 3. \tag{2.23}$$

Важнейшим параметром, определяющим корректирующую способность кода, является *минимальное кодовое расстояние* $d_{\min}$. Для его определения мы должны вычислить расстояние Хэмминга между всеми парами слов и найти наименьшее. В случае линейных кодов вычисления можно существенно сократить. Используем для этой цели основное свойство линейных кодов – свойство «замкнутости» векторного кодового пространства. Это свойство следует непосредственно из определения линейных кодов и формулируется следующим образом: любая линейная комбинация кодовых слов является кодовым словом. Рассмотрим множество двоичных кодовых слов $\{\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{2^k-1}\}$, образующих код C. Сложим каждое слово этого множества по модулю два с некоторым зафиксированным произвольным кодовым словом $\mathbf{v}_i$. Тогда множество кодовых слов отобразится само в себя, а вектор $\mathbf{v}_i$ перейдет в нулевое кодовое слово. Так как при таком отображении попарные расстояния кодовых слов не изменятся, а вектор $\mathbf{v}_i$ выбран произвольно, то $d_{\min}$ определяется как

$$d_{\min} = \min_{\mathbf{v} \in C \setminus \{\mathbf{0}\}} \omega_H(\mathbf{v}), \tag{2.24}$$

т.е минимальное кодовое расстояние $d_{\min}$ линейного кода равно минимальному весу ненулевого кодового слова.[2]

[2]Из линейности кода также следует симметричность распределения кодовых расстояний относительно любого кодового вектора *Прим. перев.*

Из таблицы 2.1 следует, что $d_{\min}$ (7,4)-кода Хэмминга равно 3.

Обобщая все приведенные выше рассуждения и примеры, мы можем определить *корректирующую способность* линейного кода следующим образом.

Линейный двоичный (n, k)-код с минимальным расстоянием Хэмминга $d_{\min} \geq 2t + 1$ может обнаружить $d_{\min} - 1$ ошибок и исправить t ошибок.[3]

2.4.2. Совершенные коды и граница Хэмминга

На рис. 2.5 изображен случай, когда все возможные принимаемые векторы $\mathbf{r}$ принадлежат областям декодирования кодовых слов.

Коды, в которых непересекающиеся сферические области декодирования охватывают все векторное пространство размерности n называются *совершенными* или *плотноупакованными*.

При использовании совершенных кодов всегда возможна коррекция ошибок (не обязательно правильная). Помимо кодов Хэмминга, в настоящее время известно мало совершенных кодов.

Найдем соотношение между параметрами совершенных двоичных (n, k)- кодов, способных исправлять t ошибок. Будем исходит из того, что область декодирования совершенного (n, k)-кода с $d_{\min} = 2t+1$ образуют 2^k непересекающихся сфер радиуса t в n-мерном векторном пространстве. Каждая сфера содержит все n-мерные векторы, находящиеся на расстоянии l от соответствующего кодового слова, причем, $0 \leq l \leq t$. Таким образом, каждой сфере принадлежит ровно

$$1 + n + \binom{n}{2} + \cdots + \binom{n}{t} = \sum_{l=0}^{t} \binom{n}{l} \tag{2.25}$$

n- мерных векторов.

Так как общий объем непересекающихся сфер не может превышать объем n-мерного векторного пространства, для двоичных кодов имеем

$$2^n \geq 2^k \sum_{l=0}^{t} \binom{n}{l} \tag{2.26}$$

или

$$2^{n-k} \geq \sum_{l=0}^{t} \binom{n}{l}. \tag{2.27}$$

[3]Если код исправляет все ошибки кратности $u \leq t$, то области декодирования представляют собой сферы радиуса t в n-мерном пространстве. - *Прим. перев.*

Равенство имеет место только для совершенных двоичных кодов. Выражение (2.27) называется границей Хэмминга. *Граница Хэмминга* является нижней оценкой необходимого числа проверочных символов двоичного кода длины n, способного исправлять t ошибок.

Из (2.27) следует, что рассмотренный нами (7,4)-код Хэмминга является совершенным, так как

$$2^{7-4} = 8 \geq \sum_{l=0}^{1} \binom{7}{l} = \binom{7}{0} + \binom{7}{1} = 1 + 7 = 8. \qquad (2.28)$$

2.4.3. Вероятность ошибки декодирования

Исходя из предыдущих рассуждений, мы можем определить вероятность необнаружимой ошибки. На самом деле, ошибка не обнаруживается, если посланное кодовое слово в канале переходит в другое кодовое слово. Из свойства замкнутого векторного пространства относительно операции сложения кода C следует, что в этом случае сама ошибка должна являться кодовым словом. Таким образом, вероятность необнаружимой ошибки определяется суммой вероятностей независимых событий $\mathbf{e} = \mathbf{v}_i$, где $\mathbf{v}_i \in C$ и $1 \leq i \leq 2^k$. Так как мы рассматриваем ДСК без памяти с вероятностью ошибки P_e, вероятность события, например, $\mathbf{e} = (0011010)$, где (0011010) – кодовое слово из табл. 2.1, равна $P_e^3(1-P_e)^4$. Обозначим через A_i число кодовых слов (n,k)-кода C веса i. Тогда *вероятность необнаружимой ошибки* для кода C равна

$$P_r = \sum_{i=d_{\min}}^{n} A_i P_e^i (1-P_e)^{n-i}. \qquad (2.29)$$

Для (7,4)-кода Хэмминга значения A_i (распределения весов) можно получить из таблицы 2.1. Имеем $A_0 = 1, A_1 = A_2 = 0, A_3 = A_4 = 7, A_5 = 0, A_6 = 0, A_7 = 1$. Если вероятность одного двоичного символа P_e известна, то можно найти вероятность необнаружимой ошибки, используя (2.29).

Не зная распределения весов, вероятность необнаружимой ошибки можно оценить сверху как

$$P_r = P_e^{d_{\min}} \sum_{i=d_{\min}}^{n} A_i \underbrace{P_e^{i-d_{\min}}(1-P_e)^{n-i}}_{<1} < (2^k - 1) P_e^{d_{\min}}. \qquad (2.30)$$

Пример: Передача данных с использованием (7,4)-кода Хэмминга.

Данные кодируются (7,4)-кодом Хэмминга и передаются по каналу с АБГШ. Отношение сигнал/шум в канале равно 6 дБ, что эквивалентно вероятности ошибки двоичного символа, равной 0,023. Скорость передачи – 16 кбит/сек. Если при декодировании обнаруживается ошибка, то по сигналу переспроса производится повторная передача кодового слова.

1. Какова вероятность, что кодовое слово будет приниматься без ошибок?

2. Какова вероятность необнаружимой ошибки?

3. Определите среднюю «эффективную» скорость в битах (т.е среднее число передаваемых информационных бит в секунду).

4. Сравните «эффективную» скорость с максимальной теоретически достижимой.

Решение.

1. Кодовое слово будет передаваться без ошибок, если все 7 двоичных символов будут переданы верно. Для ДСК без памяти с вероятностью ошибки на символ P_e, вероятность безошибочной передачи кодового слова равна

$$P_c = (1 - P_e)^7. \tag{2.31}$$

Для канала с АБГШ вероятность P_e определяется как функция от SNR и равна

$$P_e = \frac{1}{2}\operatorname{erfc}\sqrt{\frac{S}{2N}} = \frac{1}{2}\operatorname{erfc}\sqrt{\frac{10^{6/10}}{2}} \approx 0,023. \tag{2.32}$$

Подставляя P_e в (2.31), получаем

$$P_c = (1 - 0,023)^7 \approx 0,85. \tag{2.33}$$

2. Вероятность необнаружимой ошибки получим из (2.29)

$$P_r = 7 \cdot 0,023^3 \cdot 0,977^4 + 7 \cdot 0,023^4 \cdot 0,977^3 + 0,023^7 \approx 7,9 \cdot 10^{-5}. \tag{2.34}$$

Верхняя оценка P_r (2.30) дает для сравнения

$$(2^k - 1)P_e^{d_{\min}} \approx 15 \cdot 0,023^3 \approx 18 \cdot 10^{-5}. \tag{2.35}$$

3. Ввиду пренебрежимо малой вероятности необнаруженной ошибки, будем считать, что в среднем 85% кодовых слов принимается верно

без переспроса. Учитывая также, что доля информационных бит в кодовом слове равна k/n получаем, что при скорости передачи R_b эффективная скорость равна

$$R_{b,eff} = \frac{k}{n} P_c R_b \approx \frac{4 \cdot 0,85 \cdot 16 \text{ кбит/сек}}{7} \approx 7,77 \text{ кбит/сек}. \quad (2.36)$$

Замечание. *Выбранное в примере SNR, равное 6 дБ, приводит к недопустимо высокой вероятности ошибки на бит. Недопустимо высокими являются также потери эффективной скорости передачи.*

4. Пропускная способность ДСК без памяти была рассмотрена в первой части этой книги. При вероятности ошибки двоичного символа ε, равной $\varepsilon = P_e$, пропускная способность канала на один передаваемый двоичный символ составляет

$$C^{\text{ДСК}} = 1 \text{ бит} - H_b(\varepsilon) = 0,842 \text{ бит}. \quad (2.37)$$

При заданной скорости 16 кбит/сек максимально достижимая эффективная скорость передачи информации с пренебрежимо малой вероятностью ошибки равна

$$R_{\max} \leq 13,4 \text{ кбит/сек}, \quad (2.38)$$

что почти в два раза превышает величину из (2.36).

В рассмотренном примере мы использовали зависимость вероятности ошибочного бита от соотношения сигнал/шум (SNR) при передаче данных по каналу с АБГШ. Здесь мы сталкиваемся с «энергетическим» аспектом цифровой передачи информации. Рассмотрим этот аспект более подробно.

Будем исходить из постоянства некоторых параметров передачи. Пусть этими параметрами являются эффективная скорость передачи данных и средняя мощность передатчика. Пусть, далее, передача информации осуществляется по каналу с аддитивным белым гауссовским шумом (АБГШ) и прием информации производится с применением согласованных фильтров. В таком канале SNR оказывается пропорциональным длительности двоичного символа, поэтому, при постоянной мощности передатчика переход от четырех двоичных символов (передача без кодирования) к семи символам ((7,4)-код Хэмминга) внутри фиксированного интервала времени эквивалентен уменьшению SNR в 7/4 раза, что равно, приблизительно, 2,4 дБ.

И, наоборот, SNR на один двоичный символ при передаче без кодирования на 2,4 дБ выше, чем при использовании (7,4)-кода Хэмминга и составляет, в нашем случае, 8,4 дБ. (Здесь мы даже не учитываем вероятность переспроса при передаче с кодированием). Согласно (2.32), SNR, равному 8,4 дБ, соответствует вероятность ошибки на бит, равная $P_b = 0,0043$. Отсюда, вероятность безошибочной передачи блока, содержащего 4 информационных символа, составляет

$$P_c = (1 - 0,0043)^4 = 0,98. \tag{2.39}$$

Отметим, что при постоянной мощности передатчика, применяя кодирование, мы увеличиваем вероятность ошибки двоичного символа (в нашем случае от 0,0043 до 0,023). Однако, корректирующая способность кода позволяет снизить результирующую вероятность необнаружимой ошибки (в нашем случае с $1-0,98 = 0,02$ до $7,9 \cdot 10^{-5}$ на блок из четырех символов).

В настоящее время существует несколько критериев для оценки эффективности кодирования. В спутниковой связи, например, чаще всего пользуются энергетическим критерием. Сущность его в следующем: так как спутниковые линии связи близки к каналам с АБГШ, вначале находится SNR на бит передаваемой информации, обеспечивающее заданную вероятность битовой ошибки P_b при передаче без кодирования.

Аналогичное SNR на кодовый символ подсчитывается для передачи с кодированием при условии $BER = P_b$, где P_b задано. После этого, находится SNR на бит полезной информации с учетом скорости кода. Разность энергетических затрат на бит передаваемой информации при передаче без кодирования и с кодированием называется *энергетическим выигрышем кода* (ЭВК).

Замечание. *Использование кодов большой длины с довольно сложной алгебраической структурой в современных спутниковых линиях связи позволяет достичь ЭВК = 6 - 8 дБ при $P_b = 10^{-5}$. Заметим, что при снижении P_b ЭВК возрастает.*

2.4.4. Коды Хэмминга

Коды Хэмминга образуют важное семейство простейших линейных блоковых кодов. Для каждого натурального $m \geq 3$ существует двоичный код Хэмминга со следующими параметрами:

Коды Хэмминга.

- длина кодовых слов $n = 2^m - 1$

- число информационных разрядов $k = 2^m - 1 - m$
- число проверочных разрядов $m = n - k$
- корректирующая способность $t = 1, d_{\min} = 3$
- совершенные коды $\surd$

Конструкция кодов Хэмминга определяется следующими свойствами проверочной матрицы вида (2.15).

- Так как минимальное расстояние кода Хэмминга $d_{\min} = 3$, все столбцы проверочной матрицы должны быть попарно различными (проверочная матрица не должна содержать одинаковых столбцов).
- Из $d_{\min} = 3$ следует также, что каждая строка порождающей матрицы должна содержать как минимум три единицы, так как строки порождающей матрицы в свою очередь являются кодовыми словами. Если порождающая матрица представлена в виде (2.14), то это значит, что строки матрицы $\mathbf{P}$ должны содержать как минимум две единицы. (Следовательно, это относится и к столбцам транспонированной матрицы $\mathbf{P}^T$).
- Рассмотрим проверочную матрицу. Согласно (2.8), она имеет вид $\mathbf{H}_{(n-k)\times n} = (\mathbf{I}_{n-k}\mathbf{P}^T_{k\times(n-k)})$. Матрица $\mathbf{P}^T$ содержит k столбцов и $m = n-k$ строк. Используя двоичные символы «0» и «1», можно образовать 2^m различных столбцов. Но, так как ранее было сказано, что каждый столбец должен содержать как минимум две единицы, то следует отбросить один нулевой столбец и m столбцов, содержащих по одной единице. Таким образом, остается $2^m - m - 1$ возможностей. Так как $2^m - m - 1 = k$ и матрица $\mathbf{P}^T$ содержит ровно k столбцов, то все эти возможности использованы. Отсюда следует, что столбцы транспонированной матрицы $\mathbf{P}^T$ представляют собой все возможные двоичные

слова длины $m = n - k$, содержащие не менее двух единиц.[4]

Пример: (15,11)-код Хэмминга.

На примере (15,11)-кода Хэминга можно наглядно пояснить все перечисленные выше особенности конструкции:

$$\mathbf{H}_{4\times 15} = \left(\begin{array}{cccc|cccccc|cccc|c} 1&0&0&0&1&0&0&1&0&1&1&0&1&1&1\\ 0&1&0&0&1&1&0&0&1&0&1&1&0&1&1\\ 0&0&1&0&0&1&1&1&0&0&1&1&1&0&1\\ 1&0&0&1&0&0&1&0&1&1&0&1&1&1&1 \end{array}\right). \quad (2.40)$$

$\underbrace{\qquad}_{\mathbf{I}_4}$ $\underbrace{\qquad\qquad\qquad}_{\text{Вес Хэмминга } \omega_H=2}$ $\underbrace{\qquad\qquad}_{\omega_H=3}$ $\underbrace{\quad}_{\omega_H=4}$

По проверочной матрице с помощью (2.15) и (2.14) строится порождающая матрица

$$\mathbf{G}_{11\times 15} = \left(\begin{array}{cccc|ccccccccccc} 1&1&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&1&1&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&1&1&0&0&1&0&0&0&0&0&0&0&0\\ 1&0&1&1&0&0&0&1&0&0&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&1&0&0&0&0&0&0\\ 1&0&0&1&0&0&0&0&0&1&0&0&0&0&0\\ 1&1&1&0&0&0&0&0&0&0&1&0&0&0&0\\ 0&1&1&1&0&0&0&0&0&0&0&1&0&0&0\\ 1&0&1&1&0&0&0&0&0&0&0&0&1&0&0\\ 1&1&0&1&0&0&0&0&0&0&0&0&0&1&0\\ 1&1&1&1&0&0&0&0&0&0&0&0&0&0&1 \end{array}\right). \quad (2.41)$$

$\underbrace{\qquad}_{\mathbf{P}_{11\times 4}}$ $\underbrace{\qquad\qquad\qquad\qquad\qquad}_{\mathbf{I}_{11}}$

Пример: Передача данных с использованием (15,11)-кода Хэмминга.

---

[4] Отметим, что n столбцов матрицы $\mathbf{H}_{m\times n} = (\mathbf{I}_{n-k}\mathbf{P}^T_{k\times(n-k)})$ содержат все возможные двоичные слова длины m, за исключением нулевого, т.е. $n = 2^m - 1$ и все столбцы различны. Такое представление позволяет сразу же раскрыть метод коррекции одиночных ошибок. В самом деле, уравнение синдромного декодирования (2.19) имеет вид $\mathbf{s} = \mathbf{e} \odot \mathbf{H}^T$. Если $\mathbf{e}$ – вектор одиночной ошибки, то он содержит только одну единицу в i-ом ошибочном разряде кодового слова. В этом случае синдром (вектор $\mathbf{s}$) представляет собой i-ый столбец матрицы $\mathbf{H}$. Так как все столбцы матрицы $\mathbf{H}$ различны, то n возможным позициям одиночной ошибки в кодовом слове соответствуют в точности n различных синдромов, поэтому, по значению синдрома $\mathbf{s}$ можно однозначно определить номер разряда кодового слова, в котором произошла одиночная ошибка, т.е. ее исправить. – *Прим. перев.*

В данном примере мы сравниваем результаты использования для передачи данных (15,11)-кода Хэмминга с ранее полученными результатами для (7,4)-кода Хэмминга. Скорость (15,11)-кода существенно выше (7,4)-кода, так как $R_{(15,11)} = 15/11 \approx 0,73$, а $R_{(7,4)} = 4/7 \approx 0,57$. Посмотрим, как изменятся другие параметры кодирования. Для этого выполним задания 1. 2. 3. из предыдущего примера.

Решение.

1. Кодовое слово будет принято правильно, если все его разряды будут приняты без ошибок, таким образом

$$P_c = (1 - P_e)^{15}. \tag{2.42}$$

Для вероятности ошибки на двоичный символ $P_e = 0,023$, взятой из предыдущего примера (2.32), имеем

$$P_c = (1 - 0,023)^{15} \approx 0,705. \tag{2.43}$$

2. Вероятность необнаружимой ошибки получим из оценки (2.30)

$$(2^k - 1)P_e^{d_{\min}} \approx 2047 \cdot 0,023^3 \approx 0,025. \tag{2.44}$$

3. При скорости передачи двоичных символов 16 кбит/сек, эффективная скорость передачи данных составляет

$$R_{b,eff} = \frac{k}{n} P_c R_b \approx \frac{11 \cdot 0,705 \cdot 16 \text{ кбит/сек}}{15} \approx 8,27 \text{ кбит/сек}. \tag{2.45}$$

В приведенных расчетах мы исходили из одной и той же вероятности ошибки двоичного символа $P_e = 0,023$ для (15,11)- и (7,4)-кодов.

Проведем аналогичные расчеты при одинаковой мощности передатчика. В этом случае, потери энергетики на один символ по сравнению с передачей без кодирования составляют для (15,11)-кода только $10 \log_2 \frac{15}{11} \approx 1,4$ дБ (для (7,4)-кода эта величина составляет 2,4 дБ – см. предыдущий пример). Таким образом, отношению сигнал/шум равному 7 дБ соотвтствует вероятность ошибки двоичного символа, равная 0,013, поэтому

$$P_c = (1 - 0,013)^{15} \approx 0,82, \tag{2.46}$$

и вероятность необнаружения ошибки оценивается сверху как

$$(2^k - 1)P_e^{d_{\min}} \approx 2047 \cdot 0,013^3 \approx 0,0045. \tag{2.47}$$

Эффективная скорость передачи также возрастает

$$R_{b,eff} = \frac{k}{n} P_c R_b \approx \frac{11 \cdot 0,82 \cdot 16 \text{ кбит/сек}}{15} \approx 9,62\text{кбит/сек}. \quad (2.48)$$

Условие неизменной мощности передатчика более близко к практике, чем условие равенства вероятностей ошибки двоичных символов. По сравнению с (7,4)-кодом, для (15,11)-кода мы получаем некоторый выигрыш по эффективной скорости $R_{b,eff} = 9,62$ кбит/сек за счет возрастания вероятности необнаружимой ошибки.

Рассмотренные примеры показывают, что выбор кода для каждой конкретной системы должен осуществляться с особой тщательностью.

2.4.5. Расширенные коды Хэмминга

В этом разделе мы рассмотрим весьма полезное расширение кодов Хэмминга. Оно заключается в дополнении кодовых векторов дополнительным двоичным разрядом таким образом, чтобы число единиц, содержащихся в каждом кодовом слове, было четно. Коды Хэмминга с проверкой на четность обладают следующими двумя преимуществами.

- Длины кодов увеличиваются с $2^n - 1$ до 2^n, что удобно с точки зрения хранения и передачи информации.
- Минимальное расстояние $d_{\min}$ расширенных кодов Хэмминга равно 4 вместо 3, что дает возможность обнаруживать 3-кратные ошибки.

Дополнительный разряд проверки на четность позволяет использовать декодер в новом режиме – гибридном режиме обнаружения и коррекции ошибок.

В качестве примера, рассмотрим расширение (15,11)-кода Хэмминга. Каждый кодовый вектор $\tilde{\mathbf{v}} = (\tilde{v}_0, \tilde{v}_1, \ldots, \tilde{v}_{15})$ *расширенного (16,11)-кода Хэмминга* получается из кодового вектора $\mathbf{v} = (v_0, v_1, v_2, \ldots, v_{14})$ (15,11)-кода путем добавления дополнительного разряда проверки на четность, т.е.

$$\tilde{\mathbf{v}} = (\tilde{v}_0, \tilde{v}_1, \ldots, \tilde{v}_{15}) = (\tilde{v}_0, v_0, v_1, \ldots, v_{14}), \quad (2.49)$$

где

$$\tilde{v}_0 = \sum_{i=0}^{14} v_i. \quad (2.50)$$

Проверочная матрица (16,11)-кода получается из проверочной матрицы (15,11)-кода Хэмминга в два приема:

-допишем к матрице (15,11)-кода Хэмминга слева нулевой столбец;

-дополним получившуюся матрицу строчкой, полностью состоящей из одних единиц.

В результате получим

$$\widetilde{\mathbf{H}}_{5\times 16} = \begin{pmatrix} 1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,1\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1\,0\,1\,1\,1 \\ 0\,0\,1\,0\,0\,1\,1\,0\,0\,1\,0\,1\,1\,0\,1\,1 \\ 0\,0\,0\,1\,0\,0\,1\,1\,1\,0\,0\,1\,1\,1\,0\,1 \\ 0\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1\,0\,1\,1\,1\,1 \end{pmatrix}. \tag{2.51}$$

При синдромном декодировании

$$\widetilde{\mathbf{s}} = \widetilde{\mathbf{v}} \odot \widetilde{\mathbf{H}}^T, \tag{2.52}$$

причем, правая компонента равна

$$\widetilde{s}_0 = \sum_{i=0}^{15} \widetilde{v}_i. \tag{2.53}$$

С учетом (2.49) и (2.50), получаем

$$\widetilde{s}_0 = \sum_{i=0}^{15} \widetilde{v}_i = 0. \tag{2.54}$$

Из (2.49) и (2.51) следует, что и другие компоненты синдрома $\widetilde{\mathbf{s}}$ также равны нулю.

Утверждение, что минимальное кодовое расстояние $d_{\min}$ кодов Хэмминга с проверкой на четность равно 4, следует непосредственно из определения веса кодовых слов. В самом деле, так как минимальный вес всех векторов кода Хэмминга, за исключением нулевого, равен 3, то дополнительный разряд проверки на четность увеличивает этот минимальный вес до 4. Расширенный код Хэмминга также является линейным и его минимальное расстояние $d_{\min} = 4$.

Прежде чем говорить о декодировании кодов Хэмминга с проверкой на четность, напомним два возможных режима декодирования обычных кодов Хэмминга.

1. Режим обнаружения ошибок.

Если синдром $\mathbf{s} \neq \mathbf{0}$, то декодер выдает сигнал ошибки. Так как $d_{\min}$ кода Хэмминга равно 3, то ошибки кратности ≤ 2 всегда обнаруживаются.

2. Режим коррекции ошибок.

   Если синдром $\mathbf{s} \neq \mathbf{0}$, то декодер всегда исправляет один из разрядов кодового слова (так как код Хэмминга является плотно-упакованным сферами радиуса $t = 1$). Таким образом, декодер исправляет все однократные ошибки.

Из всего вышесказанного видно, что код Хэмминга или только обнаруживает все ошибки кратности не выше 2, или только исправляет все однократные ошибки.

Теперь перейдем к кодам Хэмминга с проверкой на четность. Так как $d_{\min}$ таких кодов равно 4, то в режиме обнаружения фиксируются все ошибки кратности 3 и ниже. Режим же коррекции ошибок можно существенно улучшить, благодаря наличию в кодовых словах дополнительного разряда проверки на четность.

Прежде всего заметим, что процесс исправления одиночных ошибок во всех разрядах, исключая проверочный, ничем не отличается от обычного кода Хэмминга. Таким образом, одиночная ошибка всегда может быть исправлена. (В этом случае, признаком одиночной ошибки в проверочном разряде, является равенство нулю всех компонент синдрома $\widetilde{\mathbf{s}}$ за исключением s_0, которая равна единице). С другой стороны, заметим, что при одиночной ошибке всегда выполняется равенство $\widetilde{s}_0 = 1$. При двукратной же ошибке, компонента $\widetilde{s}_0$ всегда равна «0». Таким образом, получаем следующий улучшенный алгоритм коррекции ошибок в расширенном коде Хэмминга.

1. Если $\widetilde{s}_0 = 1$, то производится исправление одиночной ошибки.

2. Если $\widetilde{s}_0 = 0$ и $\widetilde{\mathbf{s}} \neq \mathbf{0}$, то считаем, что в канале произошла неисправляемая ошибка и принятое слово или должно быть стерто, или в обратный канал должен быть подан сигнал переспроса.

Таким образом, код Хэмминга с проверкой на четность способен или только обнаруживать ошибки, кратности не выше трех, или исправлять все одиночные ошибки и, одновременно, обнаруживать все двукратные.

2.5. Приложение: Поля Галуа

Множество A образует поле, если для любых элементов $a_i, a_j, a_k \in A$ определены операции сложения «+» и умножения «×» и выполняются следующие аксиомы:

Сложение «+»

(A1) Замкнутость $a_i + a_j \in A$

(A2) Ассоциативность $(a_i + a_j) + a_k = a_i + (a_j + a_k)$

(A3) Существование единственного нулевого элемента $0 \in A$, такого, что $0 + a_i = a_i$

(A4) Обратный элемент $(-a_i) \in A$ такой, что $(-a) + a_i = 0$

(A5) Коммутативность $a_i + a_j = a_j + a_i$

Умножение «×»

(M1) Замкнутость $a_i \times a_j \in A$

(M2) Ассоциативность $(a_i \times a_j) \times a_k = a_i \times (a_j \times a_k)$

(M3) Существование единственного единичного элемента $1 \in A$ такого, что $1 \times a_i = a_i$

(M4) Обратный элемент $a_i^{-1} \times a_i = 1$

(M5) Коммутативность $a_i \times a_j = a_j \times a_i$

Сложение и умножение

(D) Дистрибутивность $a_i \times (a_j + a_k) = a_i \times a_j + a_i \times a_k$

Из перечисленных выше аксиом следуют важнейшие правила арифметики, справедливые в любом поле:

Для $0, 1, a, b, c \in A$ имеет место

1. $a + 0 = 0 \quad \Rightarrow \quad a = 0$

2. $a, b \neq 0 \quad \Leftrightarrow \quad a \times b \neq 0$

3. $a \neq 0$ и $a \times b = 0 \quad \Rightarrow \quad b = 0$

4. $-(a \times b) = (-a) \times b = a \times (-b)$

5. $a \neq 0$ и $a \times b = a \times c \quad \Rightarrow \quad b = c.$

Отметим также, что операции сложения и умножения имеют обратные операции: вычитания и деления, причем, вычитание определяется как $a - b = a + (-b)$, а деление – как $a \div b = a \times b^{-1}$.

Неподготовленного читателя может смутить и даже испугать столь громоздкое аксиоматическое построение алгебраической структуры, называемой полем. Однако, эти страхи должны довольно быстро исчезнуть после того, как мы убедимся, что множество рациональных чисел образует поле. Напомним, что множество рациональных чисел содержит все положительные и отрицательные целые числа (включая ноль), а также все числа вида n/m, где n, m – целые и $m \neq 0$. Операциями сложения, вычитания, умножения и деления в поле рациональных чисел являются обычные арифметические операции, которые мы изучали еще в начальной школе. Нетрудно заметить, что эти операции удовлетворяют всем перечисленным выше аксиомам.

Расширениями поля рациональных чисел являются поля вещественных и комплексных чисел, они также содержат бесконечное множество элементов.

Так как в каналах связи множество передаваемых сигналов всегда конечно, основой теории кодирования являются поля, содержащие конечное число элементов (поля Галуа). Простейшим полем Галуа является двоичное поле $GF(2)$, операции в котором (сложение и умножение) выполняются по правилам арифметики «по модулю 2». Нетрудно заметить, что правила арифметики по $\bmod 2$ удовлетворяют всем вышеперечисленным аксиомам (с учетом того, что обратным элементом к «1» по сложению и умножению является сама «1»).

В высшей алгебре доказывается, что число элементов q конечного поля всегда удовлетворяет условию

$$q = p^m, \tag{2.55}$$

где p – простое, а $m = 1, 2, \ldots$.

Другими словами, если число элементов q некоторого множества не удовлетворяет условию (2.55), то для этого множества невозможно определить операции сложения и умножения, удовлетворяющие аксиомам поля. Так, например, невозможно образовать поле с числом элементов, равным 6, 10, 12, 14 и т.д., но можно построить поле, с числом элементов, равным 2, 3, 4, 5, 7, 8, 9, 11 и т.д.

Таблица 2.4. Операции сложения и умножения в $GF(5)$.

+	0	1	2	3	4	×	0	1	2	3	4
0	0	1	2	3	4	0	0	0	0	0	0
1	1	2	3	4	0	1	0	1	2	3	4
2	2	3	4	0	1	2	0	2	4	1	3
3	3	4	0	1	2	3	0	3	1	4	2
4	4	0	1	2	3	4	0	4	3	2	1

Наиболее просто операции сложения и умножения выполняются в поле с числом элементов, равным простому числу ($m = 1$). Здесь они определены как операции сложения и умножения по $\bmod p$, а сами элементы образуют последовательность чисел

$$\{0, 1, 2, \ldots, p-1\}. \tag{2.56}$$

Для примера, в таблице 2.4. приведены результаты сложения и умножения всех пар элементов$a_i, b_j \in \{0, 1, 2, 3, 4\}$, т.е сумма $a_i + b_j (\bmod 5)$ и произведение $a_i \cdot b_j (\bmod 5)$. Непосредственной проверкой можно убедиться в выполнении аксиом (A1) – (A5), (M1) – (M5) и (D) для операций сложения и умножения. Таким образом, множество элементов $a \in \{0, 1, 2, 3, 4\}$ с операциями сложения и умножения, заданными табл. 2.4, образуют поле GF(5).

В теории полей Галуа доказывается следующая, очень важная теорема.

Теорема 2.5.1. В поле Галуа $GF(p)$, содержащем q элементов, существует по крайней мере один примитивный элемент α такой, что каждый ненулевой элемент из $GF(p)$ может быть представлен как некоторая степень α.

Так, в поле $GF(5)$ существует два примитивных элемента $\alpha_1 = 2$ и $\alpha_2 = 3$, так как $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 (\bmod 5) = 3$ и $3^0 = 1$, $3^1 = 3$, $3^2 (\bmod 5) = 4$, $3^3 (\bmod 5) = 2$.

Сложнее обстоит дело с построением полей Галуа $GF(p^m)$, где $m > 1$ (простое число p называется характеристикой поля).

Так как теория кодирования имеет дело, в основном, с полями характеристики 2, рассмотрим основные методы построения полей $GF(2^m)$.

Прежде всего заметим, что каждый элемент $GF(2^m)$ можно представить в виде слова длины m над $GF(2)$ или многочлена с двоичными коэффициентами, степень которого меньше, чем m. Так, например, любой элемент $a \in GF(2^3)$ можно записать как двоичное слово $a_2a_1a_0$ или как многочлен $a_2X^2 + a_1X + a_0$, где $\{a_2, a_1, a_0\} \in \{0, 1\}$. В этом случае, сложение элементов из $GF(2^m)$ выполняется по правилу сложения представляющих их многочленов в поле $GF(2)$. Если при этом умножение элементов мы определим как умножение представляющих эти элементы многочленов по модулю некоторого заданного неприводимого многочлена над $GF(2)$ степени m, то тем самым мы построим поле Галуа $GF(2^m)$.

Неприводимым называется многочлен, неразложимый на произведение многочленов с коэффициентами из $GF(2)$.

Проводя аналогию с полем $GF(p)$, можно сказать, что роль элементов $GF(p)$ в поле $GF(2^m)$ играют двоичные слова или многочлены степени, меньшей m, а роль простого числа p - неприводимый многочлен степени m.

Для реализации операций в поле $GF(2^m)$ в качестве неприводимого многочлена степени m удобнее выбирать примитивный многочлен.

Примитивным многочленом $p(X)$ над $GF(2)$ называется неприводимый многочлен степени m, такой, что в поле $GF(2^m)$, построенного по модулю $p(X)$, элемент поля X является примитивным.

В теории полей Галуа доказывается следующая теорема.

Теорема 2.5.2. Для каждого поля Галуа существуют примитивные многочлены всех степеней.

Таблицы неприводимых и примитивных многочленов над $GF(2)$ степени, не превосходящей 34, приведены в [11].

В качестве примера приведем поле Галуа $GF(2^4)$ (табл. 2.5). Для его построения был выбран примитивный многочлен четвертой степени $X^4 + X + 1$. Из таблицы видно, что степени примитивного элемента $\alpha = X$ образуют все множество ненулевых элементов $GF(2^4)$. В таком представлении операция умножения в поле $GF(2^4)$ реализуется очень просто. Пусть, например, нам требуется найти произведение элементов $(1011) \cdot (1010)$. Из таблицы 2.5 находим, что элемент (1011) можно представить в виде α^7, а (1010) в виде α^9. Из этого следует, что $(1011) \cdot (1010) = \alpha^7 \cdot \alpha^9 = \alpha^{(7+9) \bmod 15} = \alpha^1$. Используя

Таблица 2.5. Представление поля $GF(2^4)$ (Таблица антилогарифмов).

α^0	=				1	=(0001)
α^1	=			α		=(0010)
α^2	=		α^2			=(0100)
α^3	=	α^3				=(1000)
α^4	=			α	+1	=(0011)
α^5	=		α^2	$+\alpha$		=(0110)
α^6	=	α^3	$+\alpha^2$			=(1100)
α^7	=	α^3		$+\alpha$	+1	=(1011)
α^8	=		α^2		+1	=(0101)
α^9	=	α^3		$+\alpha$		=(1010)
α^{10}	=		α^2	$+\alpha$	+1	=(0111)
α^{11}	=	α^3	$+\alpha^2$	$+\alpha$		=(1110)
α^{12}	=	α^3	$+\alpha^2$	$+\alpha$	+1	=(1111)
α^{13}	=	α^3	$+\alpha^2$		+1	=(1101)
α^{14}	=	α^3			+1	=(1001)
α^{15}	=				1	$=\alpha^0$

табл. 2.5 еще раз, определяем, что α^1 соответствует элементу (0010). Окончательно получаем $(1011) \cdot (1010) = (0010)$.

При программной реализации умножения в полях Галуа, как правило, используют так называемые таблицы логарифмов и антилогарифмов. Таблица антилогарифмов поля $GF(2^4)$ совпадает с табл. 2.5. Используя эту таблицу, очень легко определить двоичный эквивалент элемента α^i по заданному i, $0 \leq i \leq 14$. Таблица логарифмов (табл. 2.6), наоборот, позволяет быстро найти степень примитивного элемента α по его двоичному представлению.

Так как операция деления в полях Галуа эквивалентна умножению на обратный элемент, весьма полезной при вычислениях оказывается таблица обратных элементов (табл. 2.7), которая для поля $GF(2^4)$ строится следующим образом. Пусть, например, нам нужно найти обратный элемент к (1010). По таблице логарифмов находим, что (1010) соответствует α^9. Обратным элементом к α^9 является $\alpha^{15-9} = \alpha^6$. По таблице антилогарифмов находим, что двоичным

Таблица 2.6. Таблица логарифмов элементов $GF(2^4)$.

0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
α^0	α^1	α^4	α^2	α^8	α^5	α^{10}	α^3	α^{14}	α^9	α^7	α^6	α^{13}	α^{11}	α^{12}

эквивалентом α^6 является (1100). Таким образом, окончательно получаем $(1010)^{-1} = (1100)$.

При небольших значениях m для ускорения умножения при программной реализации можно построить таблицу умножений элементов поля $GF(2^m)$ размерности $2^m \times 2^m$. После того, как все необходимые арифметические таблицы построены, можно заменить двоичные обозначения на целочисленные.

Реализация операции сложения (и совпадающей с ней операции вычитания) элементов в поле $GF(2^m)$ не представляет проблем. Сложение элементов из $GF(2^m)$ сводится к покомпонентному сложению их двоичных представлений по модулю 2. Так, например, в поле $GF(2^4)$ $(0011) + (1101) = (1110)$.

Таблица 2.7. Таблица обратных элементов поля $GF(2^4)$.

α_i	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
α_i^{-1}	0001	1001	1110	1101	1011	0111	0110	1111	0010	1100	0101	1010	0100	0011	1000

С подробным обоснованием построения полей $GF(2^m)$ и исследованием их алгебраической структуры читатель может ознакомиться в [5].

В заключение отметим, что при всей кажущейся сложности, использование полей Галуа в теории помехоустойчивого кодирования имеет глубокий математический смысл. Свойства полей Галуа позволяют при построении кодов использовать законы линейной алгебры, справедливые для полей действительных и комплексных чисел. Отличие заключается лишь в том, что арифметические операции необходимо производить по правилам, определенным для данного конечного поля.

ГЛАВА 3

ЦИКЛИЧЕСКИЕ КОДЫ

3.1. Введение

Прежде всего покажем, что применение на практике простейших линейных блоковых кодов с их последующим синдромным декодированием связано с чрезмерными техническими затратами. Для этой цели рассмотрим два примера.

Первым примером является протокол передачи данных по телефонному каналу ISDN-D, в котором используется формат передачи данных LAPD (Link Asset Procedure on D-channel). Все передаваемые данные заносятся в отведенные им поля в потоке данных, согласно стандарту (см. рис. 3.1). Длины полей заданы в байтах (один байт содержит блок из 8 бит). Под проверочные символы отводится поле FCS (Frame Check Sequens) длиной 2 байта. С помощью проверочных сумм производится обнаружение ошибок в поле адреса A (Adress), поле команд C (Control) и информационном поле I (Information). Таким образом, общая длина блока составляет (2+2+260+2)=266 байт или 2128 бит. При использовании для защиты данных от ошибок простейшего линейного кода с 16-ю проверочными битами потребовалась бы порождающая матрица размерности 2112×2128 и порождающая матрица размерности 16×2128.

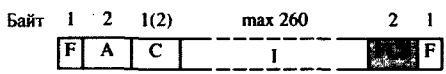


Рис. 3.1. Формат передачи данных LAPD с флагом F=(01111110), полем адреса A, полем команд C и информационным полем I.

В качестве второго примера рассмотрим формат передаваемых данных, используемый в стандарте 802.3-CSMA/CD (Carrier Sense Multiple Access/ Collision Detection) для передачи данных в локальных сетях связи (Lokal Area Network, LAN) (рис 3.2).

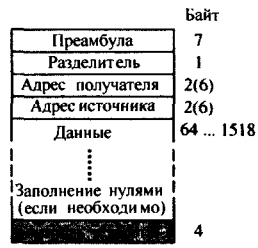


Рис. 3.2. Формат передачи данных 802.3-CSMA/CD (Carrier Sense Multiple Access/ Collision Detection).

В этом случае защита информации от помех методами, рассмотренными в предыдущей главе этой книги, также требует использования кодовых слов очень большой длины и, связанных с этим, чрезмерных технических затрат.

Оба примера показывают, что на практике кодируются и декодируются информационные потоки относительно большой длины. Применяемые при этом методы контроля ошибок должны быть максимально эффективными. В рассмотренных двух примерах этим требованиям отвечают двоичные циклические коды.

Циклические коды используются в беспроводной телефонии в стандарте DECT (Digital Enhanced Cordless Telephony) и в мобильной связи. В мобильной связи циклические коды применяются как в стандарте GSM (Global System For Mobile Communication), так и в стандарте CDMA (Code Division Multiple Access).

Далее будет показано, что, при передаче в стандарте ATM (Asynchronins Transfer Mode), циклические коды, используемые в HEC (Header Error Control), позволяют также обнаруживать ошибки синхронизации.

Замечание. *Последующее изложение материала базируется на [7]. Математические основы формулируются в виде тезисов и по мере необходимости доказываются. Доказательства иллюстрируются короткими примерами. Практические применения циклических кодов поясняются с помощью регистров с обратными связями, которые выполняют роль кодеров и декодеров. Такое изложение теории на примерах схемной реализации вначале может показаться немного непривычным. С другой стороны, детальное усвоение процессов реализации кодирования и декодирования в дальнейшем может принести Вам большую пользу. Изучив последующие разделы,*

Вы будете подготовлены к самостоятельной программной реализации изученных алгоритмов.

3.2. Определение и свойства двоичных циклических кодов

Циклические коды являются подмножеством линейных кодов. Они обладают новыми специфическими свойствами, позволяющими упрощать процессы кодирования и декодирования. При этом, корректирующая способность циклических кодов в большинстве случаев довольно высока.

Упростив изложение, мы ограничимся описанием только двоичных циклических кодов. Заметим, что операции с компонентами двоичных кодов производятся по правилам арифметики по модулю 2.

Замечание. *Двоичные циклические коды образуют линейные векторные пространства над полем Галуа $GF(2)$. На практике широко используются циклические коды с компонентами из расширенных полей Галуа $GF(2^m)$. Такими кодами являются коды Боуза-Чоудхури-Хоквингема (БЧХ) и коды Рида-Соломона (РС). Коды РС, в частности, используются в проигрывателях компакт дисков.*

Линейный (n, k)-код C является *циклическим*, если циклический сдвиг любого кодового слова из C также принадлежит коду C.

Рассмотрим *кодовое слово*

$$\mathbf{v} = (v_0, v_1, \ldots, v_{n-1}), \tag{3.1}$$

с компонентами $v_i \in \{0, 1\}$. *Циклический сдвиг* соответствует сдвигу всех компонент на один разряд вправо, причем, освободившееся место слева занимает крайняя правая компонента

$$\mathbf{v}^{(1)} = (v_{n-1}, v_0, v_1, \ldots, v_{n-2}). \tag{3.2}$$

При i-кратном циклическом сдвиге получаем

$$\mathbf{v}^{(i)} = (v_{n-i}, \ldots, v_{n-1}, v_0, v_1, \ldots, v_{n-i-1}). \tag{3.3}$$

Циклический сдвиг реализуется с помощью *регистра сдвига* длины n с обратной связью (рис. 3.3).

Циклические коды можно описать, представив кодовые векторы в виде *многочленов*. Такое представление позволяет обнаружить

некоторые дополнительные полезные математические свойства циклических кодов. Использование этих свойств приводит к построению простых и эффективных процедур кодирования и декодирования.

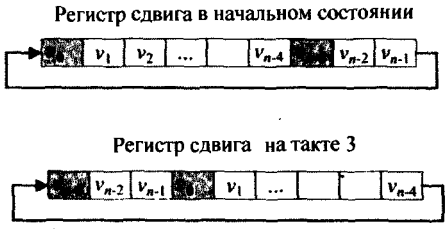


Рис. 3.3. Регистр сдвига с обратной связью.

Существует взаимно-однозначное соответствие между кодовым вектором $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ и степенью многочлена $n-1$

$$v(X) = v_0X^0 + v_1X^1 + \cdots + v_{n-1}X^{n-1}. \tag{3.4}$$

При необходимости можно переходить от векторного представления кодового слова к представлению в виде многочлена и наоборот.

Замечание. *С математической точки зрения, представление кодовых слов в виде многочлена изоморфно линейному векторному пространству кодовых векторов. При этом, операции с коэффициентами многочленов производятся по привычным правилам арифметики по модулю 2. Можно так же заметить, что степени переменной X используются только для обозначения места соответствующей компоненты кодового вектора в регистре сдвига и никакой иной смысловой нагрузки не несут.*

Представим циклический сдвиг кодового слова в виде многочлена

$$\begin{aligned} v^i(X) &= v_{n-i} + v_{n-i+1}X^1 + \cdots + v_{n-1}X^{i-1} + v_0X^i \\ &+ v_1X^{i+1} + \cdots + v_{n-i-1}X^{n-1}. \end{aligned} \tag{3.5}$$

Сравним (3.5) с результатом умножения $v(x)$ на x^i

$$X^i \cdot v(X) = v_0X^i + v_1X^{i+1} + \cdots + v_{n-1}X^{i+n-1}. \tag{3.6}$$

Внимательное рассмотрение (3.5) и (3.6) позволяет обнаружить связь между

$$\begin{aligned} v^{(i)}(X) &= \underbrace{v_{n-i} + v_{n-i+1}X^1 + \cdots + v_{n-1}X^{i-1}}_{q(x)} + \\ &+ v_0X^i + \cdots + v_{n-i-1}X^{n-1} \end{aligned} \tag{3.7}$$

и

$$\begin{aligned} X^i \cdot v(X) &= v_0 X^i + v_1 X^{i+1} + \cdots + v_{n-i-1} X^{n-1} + \\ &+ \underbrace{v_{n-i} X^n + \cdots + v_{n-1} X^{n+i-1}}_{q(X)\cdot X^n}. \end{aligned} \tag{3.8}$$

Эту связь можно выразить в виде

$$X^i \cdot v(X) = q(X) \cdot (X^n + 1) + v^{(i)}(X), \tag{3.9}$$

причем, операция сложения $q(X)$ с $v^{(i)}(X)$ устраняет ненужные компоненты.

Замечание. *Мы рассматриваем векторные пространства кодовых слов над $GF(2)$. В двоичном поле справедливо $1 \oplus 1 = 0$, т.к. обратным элементом к «1» является сама «1». Это полезное свойство не всегда является справедливым в случае произвольного поля Галуа $GF(p^m)$, что может привести к усложнению вычислений.*

Из (3.9) следует, что многочлен, соответствующий i-кратному циклическому сдвигу вектора $\mathbf{v}$ можно получить как остаток от деления многочлена $X^i v(X)$ на $(X^n + 1)$. В дальнейшем, мы покажем, что данное свойство может быть использовано для эффективного обнаружения ошибок.

Теорема 3.2.1. В каждом циклическом коде существует единственный, отличный от нуля, *кодовый многочлен* $g(X)$ минимальной степени.

Доказательство. Пусть существуют два многочлена минимальной степени $g(X)$ и $g'(X)$, отличных от нуля. Из свойства замкнутости линейного векторного кодового пространства следует, что сумма $g(X)$ и $g'(X)$ является кодовым многочленом. Отсюда получаем

$$\begin{aligned} g(X) &= g_0 + g_1 X + \cdots + g_r X^r \\ g'(X) &= g'_0 + g'_1 X + \cdots + g'_r X^r \\ g(X) + g'(X) &= (g_0 + g'_0) + (g_1 + g'_1) X + \cdots + \underbrace{(g_r + g'_r)}_{0} X^r. \end{aligned} \tag{3.10}$$

Таким образом, приходим к противоречию. ■

Теорема 3.2.2. Если $g(X)$ – *кодовый многочлен* наименьшей степени r, то его коэффициент $g_0 = 1$.

Доказательство. Сдвинем многочлен $g(X)$ $n-1$ раз. Получим многочлен

$$g^{(n-1)}(X) = g_1 + g_2X^1 + \cdots + g_rX^{r-1} + 0 \cdot X^r + 0 + \cdots + 0 + g_0X^{n-1}, \quad (3.11)$$

который также принадлежит коду. Его степень по определению не может быть меньше r, поэтому $g_0 = 1$. ■

Теорема 3.2.3. Пусть $g(X)$ – кодовый многочлен минимальной степени r. В этом случае, $v(X)$ является *кодовым многочленом* тогда и только тогда, когда он кратен $g(X)$.

Доказательство. На первом шаге докажем достаточность утверждения 3.2.3.

Пусть $v(X) = a(X)g(X)$. В этом случае

$$\begin{aligned} v(X) &= a(X) \cdot g(X) = \quad (3.12)\\ &= (a_0 + a_1X^1 + \cdots a_{n-1-r}X^{n-1-r}) \cdot g(X) = \\ &= a_0g(X) + a_1X^1 \cdot g(X) + \cdots + a_{n-1-r}X^{n-1-r} \cdot g(X). \end{aligned}$$

Так как степень многочлена $g(X)$ не превосходит r, а степень многочлена $a(X)$ не превосходит $n-1-r$, произведение $a(X)g(X)$ не содержит членов степени, большей $n-1$, то $X^ig(X)$ можно рассматривать как i-кратный сдвиг многочлена $g(X)$. Таким образом,

$$v(X) = a_0 + a_1g^{(1)}(X) + \cdots + a_{n-1-r}g^{(n-1-r)}(X). \quad (3.13)$$

Так как любой циклический сдвиг $g(X)$ так же является кодовым многочленом, то $v(X)$ представляет собой линейную комбинацию кодовых слов, т.е является кодовым словом.

На втором шаге докажем необходимость утверждения 3.2.3. Пусть

$$v(X) = c(X) \cdot g(X) + b(X), \quad (3.14)$$

где $b(x)$ - возможный остаток от деления $v(x)$ на $g(x)$. Решим уравнение относительно $b(x)$

$$b(X) = c(X) \cdot g(X) + v(X). \quad (3.15)$$

Правая часть (3.15) представляет собой сумму двух кодовых многочленов, поэтому, $b(x)$ также является кодовым многочленом. Так как, по определению, степень $b(x)$ должна быть меньше степени минимального многочлена $g(x)$, $b(x)$ соответствует нулевое кодовое слово. ■

Теорема 3.2.4. В каждом циклическом (n, k)-коде существует только один многочлен минимальной степени $r = n - k$, называемый *порождающим многочленом* $g(X) = g_0 + g_1X + \ldots + g_rX^r$ такой, что любой кодовый многочлен делится на $g(X)$.

Для поиска порождающих многочленов важным является следующее утверждение:

Теорема 3.2.5. *Порождающий многочлен* циклического кода $g(X)$ делит $X^n + 1$ без остатка.

Доказательство.

Умножив $g(X)$ на X^k, получим многочлен степени $k+r = n$. Этот же результат можно получить, если к циклическому сдвигу $g^k(X)$ прибавить $1+X^n$ для устранения лишней «1» при X^0 и компенсации недостаточной компоненты X^n. Таким образом,

$$X^k \cdot g(X) = g_0X^k + \cdots + g_rX^n = g^{(k)}(X) + 1 + X^n. \tag{3.16}$$

Представим (3.16) как результат деления $x^kg(X)$ на $X^n + 1$

$$X^k \cdot g(X) = 1 \cdot (X^n + 1) + g^{(k)}(X), \tag{3.17}$$

причем, $g^{(k)}(X)$ является остатком.

Так как циклический сдвиг $g^{(k)}(X)$ сам является кодовым многочленом, то, согласно утверждению (3.3), существует такой многочлен $a(X)$, что

$$g^{(k)}(X) = a(X) \cdot g(X). \tag{3.18}$$

Подставляя (3.18) в (3.17) и переставляя слагаемые, получим

$$X^n + 1 = [X^k + a(X)] \cdot g(X). \tag{3.19}$$

Таким образом, $g(X)$ делит $X^n + 1$ без остатка. ■

Справедливо также обратное утверждение.

Теорема 3.2.6. Если некоторый многочлен $g(X)$ степени $n-k$ делит X^n+1 без остатка, то $g(X)$ порождает некоторый циклический (n, k)-код.

Доказательство.

Докажем, что все возможные произведения $g(X)$ на многочлены, степень которых не превышает $k-1$, образуют некоторый линейный (n, k)-код и этот код является циклическим.

Все произведения $g(X)$ на многочлен степени не выше $k-1$ можно представить в виде

$$v(X) = v_0 + v_1X + \cdots v_{n-1}X^{n-1} = \\ = (a_0 + a_1X + \cdots + a_{k-1}X^{k-1}) \cdot g(X). \tag{3.20}$$

В соответствии с (3.20), всем возможным 2^k наборам двоичных коэффициентов от a_0 до a_{k-1} соответствуют 2^k различных кодовых слов. Полученный код является линейным, так как сумма двух любых кодовых слов также принадлежит коду.

Покажем теперь, что этот код является также и циклическим.

Рассмотрим произведение $X \cdot v(X)$

$$X \cdot v(X) = v_0X + v_1X^2 + \cdots + v_{n-2}X^{n-1} + v_{n-1}X^n = \\ = v^{(1)}(X) + v_{n-1}(X^n + 1). \tag{3.21}$$

Из этого следует, что для многочлена $v^{(1)}(X)$, соответствующего циклическому сдвигу $v(X)$, справедливо

$$v^{(1)}(X) = X \cdot v(X) + v_{n-1}(X^n + 1). \tag{3.22}$$

Так как $g(X)$ делит $v(X)$ и $X^n + 1$, он также является делителем $v^{(1)}(X)$. Таким образом, циклический сдвиг любого кодового слова также принадлежит коду.

Итак, множество 2^k различных многочленов, делящихся на $g(X)$, образуют линейное векторное пространство циклического (n, k)-кода. ■

Теорему 3.2.6 можно использовать как руководство к построению циклических кодов. На самом деле, пусть, например, существует некоторый многочлен степени $r = n - k$, на который делится $X^n + 1$. Тогда, этот многочлен является порождающим многочленом $g(X)$ циклического (n, k)-кода. При больших значениях n двучлен $X^n + 1$ может иметь несколько делителей степени r. В связи с этим возникает вопрос: Какой из этих делителей порождает наилучший код? К сожалению, на этот вопрос не существует однозначного ответа, тем не менее, во многих случаях можно пользоваться таблицей наилучших двоичных циклических кодов, предлагаемой ITU (International Telecommunication Union) (табл. 3.8).

Пример: Порождающий многочлен циклического (7,4)-кода.

Рассмотрим простейший циклический (7,4)-код. Для его построения требуется порождающий многочлен $g(X)$ степени $r = 7 - k = 3$,

Таблица 3.1. Циклический (7,4)-код с порождающим многочленом $g(X) = 1 + X + X^2$.

Инф. слово	Код. слово	Многочлен
0000	0000000	$v_0(X) = 0 \cdot g(X) = 0$
1000	1101000	$v_1(X) = 1 \cdot g(X) = 1 + X + X^3$
0100	0110100	$v_2(X) = X \cdot g(X) = X + X^2 + X^4$
1100	1011100	$v_3(X) = [1 + X] \cdot g(X) = 1 + X^2 + X^3 + X^4$
0010	0011010	$v_4(X) = X^2 \cdot g(X) = X^2 + X^3 + X^5$
1010	1110010	$v_5(X) = [1 + X^2] \cdot g(X) = 1 + X + X^2 + X^5$
0110	0101110	$v_6(X) = [X + X^2] \cdot g(X) = X + X^3 + X^4 + X^5$
1110	1000110	$v_7(X) = [1 + X + X^2] \cdot g(X) = 1 + X^4 + X^5$
0001	0001101	$v_8(X) = X^3 \cdot g(X) = X^3 + X^4 + X^6$
1001	1100101	$v_9(X) = [X + X^3] \cdot g(X) = 1 + X + X^4 + X^6$
0101	0111001	$v_{10}(X) = [X + X^3] \cdot g(X) = X + X^2 + X^3 + X^6$
1101	1010001	$v_{11}(X) = [1 + X + X^3] \cdot g(X) = 1 + X^2 + X^6$
0011	0010111	$v_{12}(X) = [X^2 + X^3] \cdot g(X) = X^2 + X^4 + X^5 + X^6$
1011	1111111	$v_{13}(X) = [1 + X^2 + X^3] \cdot g(X) = 1 + X + X^2 + X^3 + X^4 + X^5 + X^6$
0111	0100011	$v_{14}(X) = [X + X^2 + X^3] \cdot g(X) = X + X^5 + X^6$
1111	1001011	$v_{15}(X) = [1 + X + X^2 + X^3] \cdot g(X) = 1 + X^3 + X^5 + X^6$

являющийся делителем $X^7 + 1$. Воспользуемся разложением

$$X^7 + 1 = (1 + X) \cdot (1 + X + X^3) \cdot (1 + x^2 + X^3). \tag{3.23}$$

Правильность (3.23) можно проверить вычислением правой части в $GF(2)$.

Выберем в качестве порождающего многочлена многочлен

$$g(X) = 1 + X + X^3. \tag{3.24}$$

Информационные и кодовые слова циклического (7,4)-кода, образованного с помощью $g(X)$ из (3.24), а также соответствующие им многочлены приведены в таблице 3.1.

3.3. Систематические циклические коды

Приведенные в таблице 3.1 кодовые слова образуют несистематический код. Однако, путем некоторой модификации алгоритма кодирования можно получить систематический циклический код с теми же параметрами.

Рис. 3.4. Сдвиг информационного многочлена в регистре сдвига с обратными связями длины n на $r = n - k$ позиций.

Для этой цели рассмотрим информационный многочлен степени $k-1$

$$u(X) = u_0 + u_1 X + \cdots + u_{k-1} X^{k-1} \tag{3.25}$$

и его $r = n - k$-кратный сдвиг.

$$X^r u(X) = u_0 X^r + u_1 X^{r+1} + \cdots + u_{k-1} X^{n-1}. \tag{3.26}$$

Из рис. 3.4 видно, что такой сдвиг не вызывает переполнения n-разрядного регистра сдвига (поэтому и может быть записан в виде (3.26)) и соответствует заполнению k правых крайних двоичных разрядов регистра информационным словом. Заполним теперь свободные r левых двоичных разрядов таким образом, чтобы вектор, содержащийся в n-разрядном регистре, принадлежал коду. Для этого представим многочлен $X^r u(X)$ в виде

$$X^r u(X) = a(X) \cdot g(X) + b(X), \tag{3.27}$$

где $b(X)$ – остаток от деления $X^r u(X)$ на $g(X)$.

Из (3.27) следует

$$X^r u(X) + b(X) = a(X) \cdot g(X). \tag{3.28}$$

Из (3.28) вытекает алгоритм кодирования систематического циклического (n, k)-кода:

1. Информационный многочлен $u(X)$ степени $k-1$ умножается на X^r, где $r = n - k$;

2. Находится остаток $b(X)$ от деления $X^r u(X)$ на $g(X)$;

3. Многочлен $b(X)$ заносится в r левых разрядов регистра сдвига (рис. 3.4).

Заметим, что эта операция всегда возможна, так как степень $b(X)$ по определению не превышает $r-1$. Таким образом, в регистре сдвига будет сформирован многочлен

$$v(X) = \underbrace{b_0 + b_1 X + \cdots b_{r-1} X^{r-1}}_{r \text{ проверочных символов}} + \underbrace{u_0 X^r + u_1 X^{r+1} + \cdots u_{k-1} X^{n-1}}_{k \text{ информационных символов}}. \quad (3.29)$$

Многочлен $v(X)$ принадлежит циклическому коду, так как в силу (3.28) он делится на $g(X)$ без остатка. Более того, этот код является систематическим, так как из (3.29) следует, что старшие k разрядов кодовых векторов являются информационными векторами.

Следующий пример наглядно поясняет алгоритм кодирования циклических систематических кодов.

Пример: Циклический (7,4)-код в систематической форме.

В качестве порождающего многочлена будем использовать уже известный из предыдущего примера многочлен $g(X) = 1 + X + X^3$ (3.24). Пусть задан информационный вектор

$$\mathbf{u} = (1001). \quad (3.30)$$

Ему соответствует информационный многочлен

$$u(X) = 1 + X^3. \quad (3.31)$$

Умножим информационный многочлен на X^3

$$X^3 u(X) = X^3 + X^6 \quad (3.32)$$

и определим остаток $b(X)$ от деления (3.32) на $g(X)$. Процесс нахождения остатка $b(X)$ в соответствии с *алгоритмом деления Евклида* показан в табл. 3.2. В результате получим

$$X^3 u(X) = (X + X^3) g(X) + \underbrace{X + X^2}_{b(X)}. \quad (3.33)$$

Таблица 3.2. Определение проверочных символов для $u(X) = 1 + X^3$ и $g(X) = 1 + X + X^3$.

X^6	X^5	X^4	X^3	X^2	X	1	
1	0	0	1	0	0	0	$= X^3u(X)$
1	0	1	1	0	0	0	$= X^3g(X)$
-	-	1	0	0	0	0	$= X^3u(X) + X^3g(X)$
		1	0	1	1	0	$= Xg(X)$
		-	-	1	1	0	$= b(X)$

Так как кодовый многочлен определяется как

$$v(X) = b(X) + X^3u(X), \tag{3.34}$$

то

$$\mathbf{v} = (011\ \ 1001). \tag{3.35}$$

Повторяя процесс кодирования для всех 16 возможных информационных векторов, получим систематический циклический (7,4)-код. Его информационные и кодовые векторы приведены в табл 3.3. Заметим, что циклический систематический (7,4)-код, образованный порождающим многочленом $1 + X + X^3$, совпадает с рассмотренным нами ранее систематическим (7,4)-кодом Хэмминга (см. табл. 2.1).

3.4. Порождающая и проверочная матрицы

Циклические коды образуют подмножество линейных блоковых кодов и, помимо общих особенностей линейных блоковых кодов, они обладают некоторыми специфическими свойствами и методами описания. Рассмотрим сначала *порождающую матрицу* циклического кода. В соответствии с теоремой 3.2.3, каждый многочлен циклического кода может быть представлен в виде произведения

$$v(X) = a(X)\cdot g(X) = a_0g(X) + a_1Xg(X) + \cdots + a_{k-1}X^{k-1}g(X). \tag{3.36}$$

Заметим, что каждое слагаемое в (3.36) представляет собой сдвиг порождающего многочлена $g(X)$, которому соответствует вектор

$$\mathbf{g} = (1, g_1, \ldots, g_{r-1}, 1), \tag{3.37}$$

Таблица 3.3. Циклический (7,4)-код, образованный порождающим многочленом $g(X) = 1 + X + X^3$ в систематическом виде.

Информационное слово	Кодовое слово	Информационное слово	Кодовое слово
0000	000 0000	0001	101 0001
1000	110 1000	1001	011 1001
0100	011 0100	0101	110 0101
1100	101 1100	1101	000 1101
0010	111 0010	0011	010 0011
1010	001 1010	1011	100 1011
0110	100 0110	0111	001 0111
1110	010 1110	1111	111 1111

поэтому, кодовый вектор $\mathbf{v}$, соответствущий многочлену $v(X)$, может быть представлен в виде произведения информационного вектора $\mathbf{a}$ на порождающую матрицу $\mathbf{G}$

$$\mathbf{v}_{1\times n} = \mathbf{a}_{1\times k} \odot \mathbf{G}_{k\times n}, \tag{3.38}$$

где порождающая матрица $\mathbf{G}$ имеет вид

$$\mathbf{G}_{k\times n} = \begin{pmatrix} 1 & g_1 & g_2 & \cdots & g_{r-1} & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & g_1 & \cdots & g_{r-2} & g_{r-1} & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & \cdots & & g_{r-1} & 1 & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & & \ddots & \ddots & 0 \\ 0 & \cdots & & 0 & 1 & g_1 & g_2 & \cdots & g_{r-1} & 1 \end{pmatrix}. \tag{3.39}$$

Пример: Порождающая и проверочная матрица циклического (7,4)-кода Хэмминга.

Используя уже известный порождающий многочлен $g(X) = 1 + X + X^3$ (3.24), получим порождающую матрицу несистематического циклического (7,4)-кода

$$\mathbf{G}_{4\times 7} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}. \tag{3.40}$$

Как уже указывалось ранее, кодовое векторное пространство, образованное порождающим многочленом $g(X) = 1+X+X^3$, совпадает с линейным векторным пространством (7,4)-кода Хэмминга , следовательно матрицу $\mathbf{G}$ из (3.40) можно также рассматривать, как порождающую матрицу циклического кода Хэмминга в несистематической форме.

Путем элементарных матричных преобразований порождающая матрица $\mathbf{G}$ из (3.40) может быть приведена к систематическому виду. Так как каждая строка матрицы $\mathbf{G}$ является кодовым словом, замена любой строки суммой этой строки с другой, отличной от нее, не меняет кодового векторного пространства (меняется лишь система соответствий между информационными и кодовыми векторами). Заменим в (3.40) вначале третью и четвертую строку их суммами с первой строкой и, далее, полученную четвертую строку - ее суммой с первой. В результате, получим порождающую матрицу систематического циклического кода Хэмминга, совпадающую с (2.2)

$$\mathbf{G}'_{4\times 7} = (\mathbf{P}_{4\times 3} \quad \mathbf{I}_4) = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.41}$$

Возьмем информационный вектор из (3.30) $\mathbf{u} = (1\ 0\ 0\ 1)$. Ему будет соответствовать кодовый вектор

$$\mathbf{v} = \mathbf{u} \odot \mathbf{G}'_{4\times 7} = (1001) \odot \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 1\,1\,1\,0\,0\,1\,0 \\ 1\,0\,1\,0\,0\,0\,1 \end{pmatrix} = (011\ \ 1001), \tag{3.42}$$

что совпадает с полученным ранее (3.35) кодовым вектором систематического циклического кода и табл. 3.3.

Проверочная матрица циклического систематического (7,4)-кода может быть получена из порождающей матрицы G' (3.41) по рассмотренным ранее формальным правилам. Здесь, однако, мы хотим получить проверочное соотношение и построить проверочную матрицу, исходя только из свойств циклических кодов. Для этого, воспользуемся сформированным ранее утверждением 3.2.5 о том, что порождающий многочлен $g(X)$ делит $X^n + 1$ без остатка. Следовательно, можно написать

$$X^n + 1 = h(X) \cdot g(X). \tag{3.43}$$

Так как кодовый многочлен можно представить в виде

$$v(X) = a(X) \cdot g(X), \tag{3.44}$$

то, с учетом (3.43), произведение $v(X)h(X)$ равно

$$\begin{aligned} v(X) \cdot h(X) &= a(X) \cdot h(X) \cdot g(X) = \\ &= a(X) \cdot [1 + X^n] = a(X) + a(X)X^n. \end{aligned} \tag{3.45}$$

Так как степень многочлена $a(X)$ не превышает $k-1$, правая часть равенства (3.45) не содержит в качестве слагаемых члены с X^k, $X^{k+1} \dots X^{n-1}$. Используя это условие для коэффициентов произведения $v(X)h(X)$, можно записать $n-k$ проверочных равенств

$$\begin{array}{rrrrrrr} v_0 \odot h_k \oplus & v_1 \odot h_{k-1} \oplus & \cdots & v_k \odot h_0 & & & = 0 \\ v_1 \odot h_k \oplus & \cdots & v_k \odot h_1 \oplus & v_{k+1} \odot h_0 & & & = 0 \\ v_2 \odot h_k \oplus & \cdots & v_{k+1} \odot h_1 \oplus & v_{k+2} \odot h_0 & & & = 0 \\ \ddots & & \ddots & & \vdots & & \end{array} \tag{3.46}$$

Эти равенства можно записать в матричной форме. Введя *проверочную матрицу*

$$\mathbf{H}_{(n-k)\times n} = \begin{pmatrix} h_k & h_{k-1} & h_{k-2} & \cdots & h_1 & h_0 & 0 & 0 & \cdots & 0 \\ 0 & h_k & h_{k-1} & h_{k-2} & & h_1 & h_0 & 0 & \cdots & 0 \\ 0 & 0 & h_k & h_{k-1} & h_{k-2} & & h_1 & h_0 & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & & \ddots & \ddots & 0 \\ 0 & \cdots & & 0 & h_k & h_{k-1} & h_{k-2} & \cdots & h_1 & h_0 \end{pmatrix} \tag{3.47}$$

получим уже известное матричное уравнение для синдромного декодирования

$$\mathbf{v} \odot \mathbf{H}^T = \mathbf{0}. \tag{3.48}$$

Определение 3.4.1. Пусть задан порождающий многочлен $g(X)$ степени $r = n - k$ циклического (n, k)-кода C. В этом случае, многочлен $h(X)$ степени k такой, что $X^n + 1 = g(X)h(x)$ называется *проверочным многочленом.*

Многочлен, взаимо обратный проверочному многочлену $h(X)$, т.е многочлен $X^k(h(X^{-1})) = h_k + h_{k-1}X + \cdots + h_0X^k$ является порождающим многочленом $(n, n-k)$-кода C_d, *дуального коду* C.

Замечание. *Преобразованию* $X^i h(X^{-1})$ *соответствует зеркальное отображение его коэффициентов.*

Пример: Порождающий многочлен кода, дуального циклическому (7,4)-коду Хэмминга.

Рассмотрим опять циклический код с порождающим многочленом $g(X) = 1 + X + X^3$ из (3.24). Из разложения на множители $X^7 + 1$ (3.23) следует, что его проверочный многочлен имеет вид

$$h(x) = (1 + X) \cdot (1 + X^2 + x^3) = 1 + X + X^2 + X^4, \tag{3.49}$$

поэтому, порождающий многочлен дуального кода определяется как

$$X^4 h(X^{-1}) = X^4 + X^3 + X^2 + 1 \tag{3.50}$$

и он порождает циклический (7,4)-код.

Теперь перед нами стоит задача получить порождающую матрицу систематического циклического (n, k)-кода, с заданным порождающим многочленом $g(X)$, не прибегая к элементарным преобразованиям матрицы (3.39). Здесь ключевая идея будет состоять в том, что любые k линейно независимых векторов, делящихся на $g(X)$, образуют одно и то же кодовое векторное пространство. Надо лишь выбрать эти векторы таким образом, чтобы им соответствовала порождающая матрица систематического (n, k)-кода.

Представим X^{r+i} в виде

$$X^{r+i} = a_i(X)g(X) + b_i(X), \tag{3.51}$$

где $i = 0, 1, \cdots k - 1$ и

$$b_i(X) = b_{i,0} + b_{i,1}X + \cdots + b_{i,r-1}X^{r-1}. \tag{3.52}$$

Равенству (3.51) соответствуют k линейно независимых кодовых многочлена

$$v_i(X) = a_i(X)g(X) = X^{r+i} + b_i(X). \tag{3.53}$$

Таким образом, векторное пространство систематического циклического (n, k)-кода «натянуто» на набор k векторов $X^{r+i} + b_i(X)$, где $i = 0, 1, \cdots k - 1$. Здесь единицы при X^{r+i} образуют единичную подматрицу и являются признаком независимости базисных многочленов $v_i(X)$.

Порождающая матрица, соответствующая (3.53), имеет вид

$$\mathbf{G}_{k\times n} = \left(\underbrace{\begin{matrix} b_{0,0} & b_{0,1} & b_{0,2} & \cdots & b_{0,r-1} \\ b_{1,0} & b_{1,1} & b_{1,2} & \cdots & b_{1,r-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{k-1,0} & b_{k-1,1} & b_{k-1,2} & \cdots & b_{k-1,r-1} \end{matrix}}_{\mathbf{P}_{k\times r}} \underbrace{\begin{matrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & \cdots & 0 \end{matrix}}_{\mathbf{I}_{k\times k}} \right). \tag{3.54}$$

Таблица 3.4. Определение порождающей матрицы в систематическом виде.

X^{r-i}	Многочлен из (3.51)	Базовый кодовый многочлен
$X^3 =$	$g(X)+1+X$	$v_0(X) = 1 + X + X^3$
$X^4 =$	$Xg(X)+X+X^2$	$v_1(X) = 1 + X^2 + X^4$
$X^5 =$	$(X^2+1)g(X)+1+X+X^2$	$v_2(X) = 1 + X + X^2 + X^5$
$X^6 =$	$(x^3+X+1)g(X)+1+X^2$	$v_3(X) = 1 + X^2 + X^6$

Как уже доказывалось ранее, проверочная матрица систематического кода образуется из (3.54) в форме

$$\mathbf{H}_{r\times n} = (\mathbf{I}_{r\times r} \quad \mathbf{P}^T). \tag{3.55}$$

Пример: Порождающая матрица систематического циклического (7,4)-кода Хэмминга.

Найдем порождающую матрицу систематического циклического (7,4)-кода по уже известному порождающему многочлену $g(X) = 1+X+X^3$ из (3.24). Для этой цели определим, вначале, разложение X^{r+i} согласно (3.51) и базисные кодовые многочлены $v_i(X)$ из (3.53) для $i = 0, 1, 2, 3$ (табл. 3.4).

Порождающая матрица непосредственно строится по базисным кодовым многочленам $v_0(X), v_1(X), v_2(X), v_3(X)$ и полностью совпадает с (3.42)

$$\mathbf{G}_{4\times 7} = \left(\begin{array}{ccc|cccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right) = \left(\underbrace{\begin{array}{ccc} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{array}}_{\mathbf{P}_{4\times 3}} \quad \underbrace{\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}}_{\mathbf{I}_{4\times 4}}\right). \tag{3.56}$$

3.5. Схемная реализация циклического кодирования

Важнейшее преимущество циклических кодов по сравнению с другими методами кодирования заключается в простоте их технической реализации. Использование в схемах кодеров и декодеров регистров

Таблица 3.5. Алгоритм Евклида деления многочлена $f(x) = 1 + X + X^2 + X^4$ на $g(x) = 1 + X^2$.

X^4	–	X^2	+	X	+	1	$f(X)$
X^4		X^2					$X^2 g(X)$
-		-		X	+	1	Остаток

сдвига с обратными связями, позволяет просто и достаточно эффективно защищать от ошибок информационные массивы очень большой длины.

Замечание. *Эта особенность присуща всем циклическим кодам над полями Галуа. Для простоты изложения, здесь мы ограничимся только двоичными кодами, т.е. циклическими кодами над* $GF(2)$.

Так как *процедура деления многочленов* является основной в кодерах и декодерах циклических кодов, покажем, прежде всего, как эта процедура может быть реализована с помощью регистров сдвига с обратными связями. Для начала, сопоставим деление многочлена $f(X) = 1+X+X^2+X^4$ на многочлен $g(X) = 1+X^2$ по алгоритму Евклида (табл. 3.5) с его схемной реализацией (рис. 3.5). В таблице 3.5 делитель $g(X)$ домножается на X^2, так как степень $f(X)$ равна 4 и вычитается из делимого $f(X)$. В результате, полученный многочлен имеет степень, меньшую $g(X)$, так что мы сразу же имеем остаток от деления $f(X)$ на $g(X)$. Аналогичная процедура имеет место и на рис. 3.5, однако, здесь требуется три такта для того, чтобы младшие разряды $f(X)$ заняли место остатка.

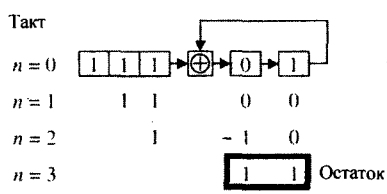


Рис. 3.5. Схема деления многочлена $1+X+X^2+X^4 : 1+X^2$.

Теперь рассмотрим схемную реализацию деления двоичных многочленов в общем случае. Пусть заданы: делимое степени m

$$f(X) = f_0 + f_1 X + f_2 X^2 + \cdots + f_m X^m \qquad (3.57)$$

и делитель степени r, причем, $r < m$

$$g(X) = g_0 + g_1 X + g_2 X^2 + \cdots + g_r X^r. \tag{3.58}$$

В результате деления мы должны получить разложение

$$f(X) = a(X)g(X) + b(X) \tag{3.59}$$

с сомножителем $a(X)$ степени $m-r$ и остатком $b(X)$ со степенью, на превышающей $r-1$.

Схема деления многочленов общего вида представлена на рис. 3.6.

Сначала регистр сдвига полностью загружается старшими разрядами делимого. Ключ $S1$ включен, а переключатель $S2$ находится в верхнем положении. Далее начинается сам процесс деления. В первом такте производится сдвиг содержимого регистра на один разряд вправо. Так как $f_m = 1$, эта единица, в соответствии с коэффициентом двигателя $g(X)$, суммируется с аналогичными разрядами делимого. В результате, мы получаем укороченный многочлен

$$\tilde{f}(X) = f(X) + X^{m-r} g(X) \tag{3.60}$$

со степенью

$$\deg[\tilde{f}(X)] = k_1 < m. \tag{3.61}$$

Эта же единица заносится в регистр формирования $a(X)$ при замкнутом переключателе $S2$ и в дальнейшем не меняется.

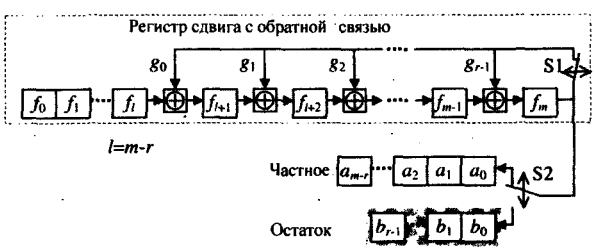


Рис. 3.6. Схема деления многочленов $f(x) : g(x) = a(x)g(x) + b(x)$.

На последующих $l = m - r$ тактах алгоритм деления остается таким же. Так, если степень укороченного многочлена $\tilde{f}(X)$ в (3.60),

равная k_1 (3.61), остается большей и равной r, то с помощью цепи обратной связи производится укорочение теперь уже многочлена $\tilde{f}(X)$ из (3.60).

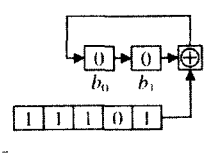
$$\tilde{\tilde{f}}(X) = \tilde{f}(X) + X^{k_1-r}g(X) = f(X) + [X^{m-r} + X^{k_1-r}]g(X) \quad (3.62)$$

со степенью

$$\deg[\tilde{\tilde{f}}(X)] = k_2 < k_1. \quad (3.63)$$

Таким образом, после $l = m - r$ тактов мы получаем разложение (3.59), причем, в регистре делимого находится остаток от деления $b(X)$. После этого, ключ $S1$ размыкается, переключатель $S2$ переводится в нижнее положение и на следующих r тактах остаток $b(X)$ заносится в регистр формирователя остатка.

Как уже упоминалось ранее, кодер систематического кода не использует при своей работе сомножитель $a(X)$ из разложения (3.59). Его задачей является только получение остатка от деления сдвига информационного многочлена на порождающий многочлен $g(X)$. Этот остаток образует затем младшие разряды кодового слова.



Такт n	$f(X)$	b_0	b_1
0	1 1 1 0 1	0	0
1	1 1 1 0	1	0
2	1 1 1	0	1
3	1 1	0	0
4	1	1	0
5	Остаток	1	1

Рис. 3.7. Схема деления многочлена $1 + X + X^2 + X^4$ на $1 + X^2$.

В связи с вышеизложенным, схему деления двух многочленов (рис. 3.6) для нужд кодирования можно существенно упростить. Заметим, что для получения остатка от деления по алгоритму Евклида, нам достаточно хранить в памяти помимо делимого только промежуточные суммы. Если степень $g(X)$ равна r, каждая промежуточная сумма занимает не более чем r двоичных разрядов и обновляется на каждом такте процедуры деления, причем, на этом же такте производится сложение очередного двоичного разряда промежуточной суммы с соответствующим разрядом делимого. Таким образом, схема деления многочлена $f(X) = 1 + X + X^2 + X^4$ на $g(X) = 1 + X^2$

(рис. 3.5) преобразуется в схему получения остатка от деления (рис. 3.7). Здесь уже не требуется предварительной загрузки делимого в регистр деления, а остаток b_0, b_1 образуется в этом регистре на пятом такте.

Выше мы подробно разобрали схемные реализации алгоритмов деления Евклида. Покажем теперь принцип действия схем кодирования циклических кодов, построенных на регистрах сдвига с обратными связями. Рассмотрим уже знакомый нам систематический код Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$ (3.24). Пусть передается информационный вектор $\mathbf{u} = (1001)$ из (3.30). Согласно (3.35), ему соответствует кодовое слово в систематическом виде $\mathbf{v} = (011\ 1001)$. Алгоритм деления Евклида для вычисления проверочных разрядов приведен в табл. 3.2. Здесь делимым является информационный многочлен $u^{(r)}(X)$, делителем – порождающий многочлен $g(X)$. Проверочные разряды определяются как коэффициенты многочлена $b(X)$, который представляет собой остаток от деления $u^{(r)}(X)$ на $g(X)$.

Для формирования кодового слова $\mathbf{v}$ по информационному вектору $\mathbf{u}$ используется цепь деления (рис. 3.8). Для получения проверочных разрядов, воспользуемся вычислениями, приведенными в табл. 3.2. Поместим информационные биты $\mathbf{u} = (1001)$ во входной регистр и обнулим разряды верхнего регистра формирователя остатка b_0, b_1, b_2. Ключ выходного регистра находится в положении $S1$, а цепь обратной связи в устройстве деления замкнута. На первом такте старший разряд двоичного информационного вектора u_3 заносится во выходной регистр сдвига, одновременно сумма u_3 и b_2 подается в цепь обратной связи регистра формирования остатка.

Результат вычисления $u_3 \oplus b_2 = 1$ заносится в разряды b_0 и b_1 верхнего регистра. Обратим внимание на тот факт, что при вычислении первой промежуточной суммы в алгоритме деления Евклида (см. табл. 3.2) происходит сложение g_1 с u_1 и g_0 с u_2. Это в точности соответствует сложению $b_1 = b_0 = 1$ с разрядами, поступающими из входного регистра сдвига (см. рис. 3.8) двумя и тремя тактами позже. Таким образом, на первом такте $b_1 = b_0 = 1$, а $b_2 = 0$.

Замечание. *Операция сложения производится по правилам арифметики по модулю 2 и обозначается знаком* $\oplus$. *Заметим, что в арифметике по модулю 2, «-1» равна «+1» и операция вычитания эквивалентна операции сложения.*

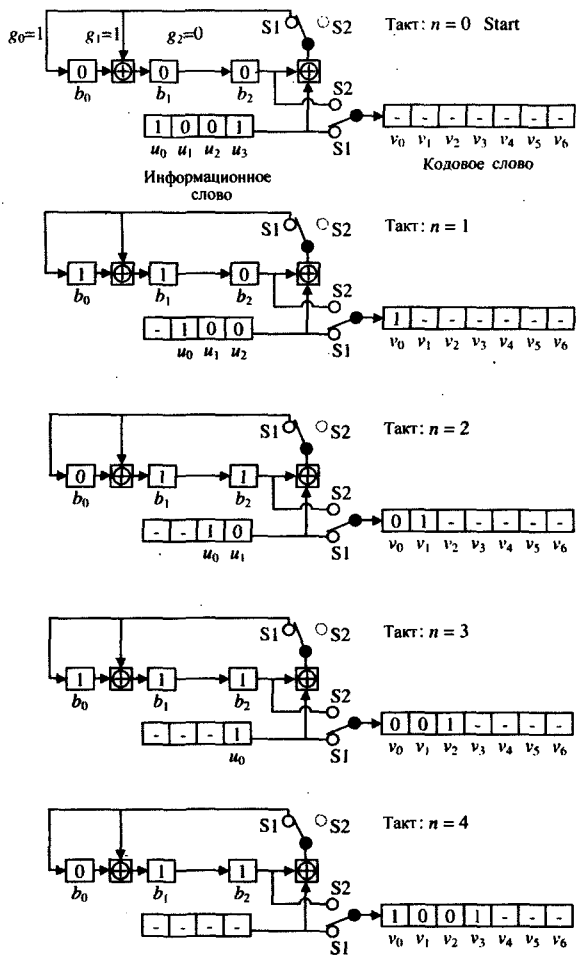


Рис. 3.8. Кодер систематического циклического (7,4)-кода с порождающим многочленом $g(X) = 1 + X + X^3$.

Так как схема получения кодового слова **v** по информационному вектору **u** остается неизменной во времени, то она обладает свойством линейности относительно операций в поле $GF(2)$. Следовательно, многократные сдвиги и суммирования коэффициентов по-

рождающего многочлена $g(X)$ и полученные промежуточные результаты соответствуют промежуточным результатам алгоритма деления Евклида (табл. 3.2).

На втором такте разряд u_2 загружается в выходной регистр формирования кодового слова. Одновременно вычисляется сумма u_2 с очередным значение b_2. Результат $u_2 \oplus b_2 = 0$ подается в цепь обратных связей. Таким образом, значение b_0 равно «0», а $b_1 = b_2 = 1$.

На третьем такте в выходной регистр загружается информационный разряд u_1 и, одновременно, вычисляется сумма $u_1 \oplus b_2$. Результат вычисления $u_1 \oplus b_2 = 1$ поступает в цепь обратных связей и после третьего такта $b_0 = b_1 = b_2 = 1$.

На четвертом такте в выходной регистр заносится младший разряд u_0. Вычисленное значение $u_0 \oplus b_2 = 0$ подается в цепь обратных связей. В результате имеем $b_0 = 0$ и $b_1 = b_2 = 1$.

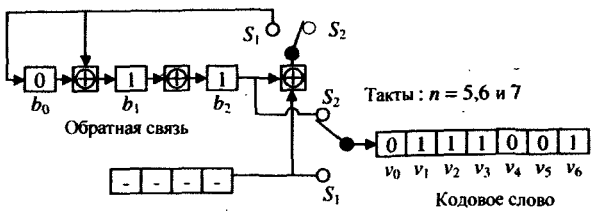


Рис. 3.9. Процедура считывания проверочных символов.

Согласно алгоритму деления Евклида, после четвертого такта многочлен $b(X) = b_0 + b_1X + b_2X^2$ равен остатку от деления $u^{(r)}(X)$ на $g(X)$. Формирование проверочных символов завершено. На пятом, шестом и седьмом тактах проверочные символы b_2, b_1 и b_0 дописываются к старшим разрядам $u^{(r)}(X)$ кодового слова $v(X)$. Для этого оба ключа предварительно переводятся в положение S_2 (рис. 3.9).

В рассмотренном примере показана реализация алгоритма деления Евклида с помощью регистра сдвига со встроенными сумматорами, поэтому данная схема не ограничивается приведенными числовыми значениями. Исходя из заданного многочлена $g(X) = 1 + g_1X + g_2X^2 + \cdots + g_{r-1}X^{r-1} + X^r$, можно построить схему кодирования для любого двоичного циклического (n, k)-кода (рис. 3.10).

Замечание. *Таким же образом строятся кодеры дуальных циклических кодов с порождающим многочленом $h(X)$.*

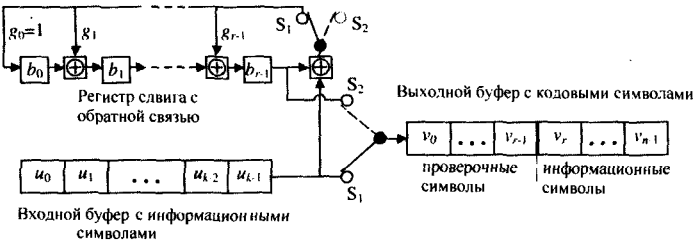


Рис. 3.10. Кодер систематического циклического (n, k)-кода с порождающим многочленом $g(X)$.

3.6. Синдром циклических кодов и контроль ошибок

Рассмотрим модель передачи информации (рис.3.11). При передаче по каналу связи с шумом к кодовому слову $v(X)$ добавляется *многочлен ошибок* $e(X)$. В результате, *многочлен принятого кодового слова* имеет вид:

$$r(X) = v(X) + e(X) \tag{3.64}$$

или

$$r(X) = a(X)g(X) + s(X), \tag{3.65}$$

где $s(X)$ представляет собой *синдром*. Если $r(X)$ является кодовым словом, то $s(X)$ - нулевой многочлен.

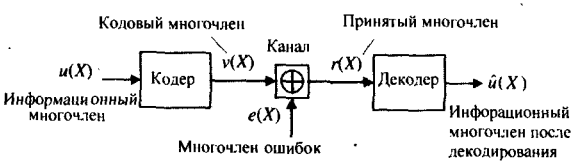


Рис. 3.11. Модель передачи информации.

Синдром $s(X)$ может быть вычислен с помощью алгоритма деления Евклида. Такое вычисление можно реализовать на простой цепи (рис. 3.12), во многом схожей с кодером систематического циклического кода (рис. 3.10). В схеме, приведенной на рис. 3.12, определяется остаток от деления некоторого многочлена на порождающий

многочлен $g(X)$. Сначала, в декодере производится обнуление двоичных разрядов синдрома $s_0, s_1, \ldots, s_{r-1}$, и в регистр синдрома заносятся r первых принятых из канала бит. Остаток от деления $r(X)$ на порождающий многочлен $g(X)$ по алгоритму Евклида заносится в регистр синдрома. Рассмотрим процедуру вычисления синдрома $s(X)$ на примере.

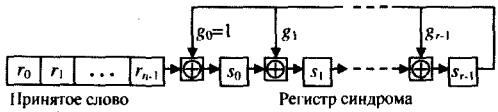


Рис. 3.12. Вычисление синдрома систематического (n, k)-кода с порождающим многочленом $g(X)$.

Пример: Вычисление синдрома циклического (7,4)-кода Хэмминга.

В качестве примера, рассмотрим уже известный циклический (7,4)-код Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$. Пусть информационный вектор $\mathbf{u} = (1001)$. Как мы уже знаем из предыдущего примера, этому вектору соответствует кодовое слово $\mathbf{v} = (011\ 1001)$.

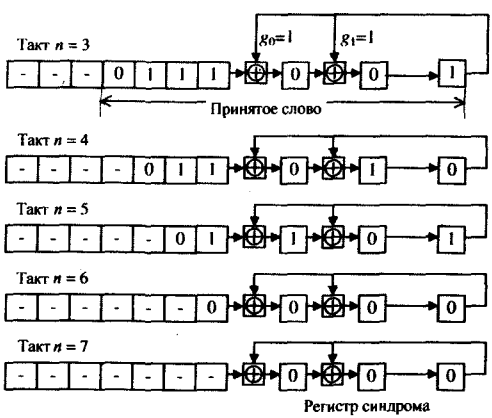


Рис. 3.13. Вычисление синдрома систематического (7,4)-кода по принятому слову в канале без шума.

При передаче по каналу без шума $\mathbf{r} = \mathbf{v}$. Процедура вычисления синдрома по тактам, в этом случае, представлена на рис. 3.13. Так как принятое из канала слово является кодовым, мы получим нулевой синдром.

Пусть, из-за воздействия шума в канале произошла одна ошибка $\mathbf{r} = (011\ 1011)$. Тогда, в результате вычисления остатка от деления $r(X)$ на $g(X)$ (рис. 3.14), мы уже получаем ненулевой синдром. Таким образом, произошло распознавание ошибки.

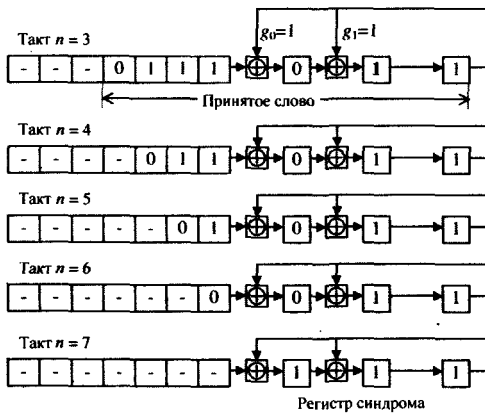


Рис. 3.14. Вычисление синдрома систематического $(7,4)$-кода при одиночной ошибке в принятом слове.

Преимущества циклических кодов не ограничиваются одной лишь простотой вычисления синдрома. Рассмотрим некоторые дополнительные свойства, которыми обладают их синдромы.

Теорема 3.6.1. Пусть $s(X)$ – *синдром* принятого из канала слова $r(X)$ некоторого циклического (n,k)-кода. Обозначим через $s_1(X)$ остаток от деления многочлена $X \cdot s(X)$ на порождающий многочлен $g(X)$. Тогда $s_1(X)$ является синдромом $r^{(1)}(X)$, т.е. остатком от деления циклического сдвига $r(X)$ на порождающий многочлен $g(X)$.

Доказательство. Рассмотрим многочлен

$$r(X) = r_0 + r_1 X + \cdots + r_{n-1} X^{n-1}. \tag{3.66}$$

Произведение $X \cdot r(X)$ имеет вид

$$X \cdot r(X) = r_0 X + r_1 X^2 + \cdots + r_{n-1} X^n. \tag{3.67}$$

Циклический сдвиг многочлена $r(X)$ можно записать следующим образом:

$$\begin{aligned} r^{(1)}(X) &= r_{n-1} + r_0 X + r_1 X^2 + \cdots + r_{n-2} X^{n-1} = \\ &= r_{n-1} \cdot [X^n + 1] + X \cdot r(X). \end{aligned} \tag{3.68}$$

Запишем $r^{(1)}(X)$ в виде $r^{(1)}(X) = a(X)g(X) + \tilde{s}(X)$, а $r(X)$ в виде $r(X) = c(X)g(X) + s(X)$, где $\tilde{s}(X)$ и $s(X)$ синдромы многочленов $r^{(1)}(X)$ и $r(X)$. Воспользуемся соотношением $X^n + 1 = g(X)h(X)$ из теоремы 3.2.5, тогда имеем

$$c(X)g(X) + \tilde{s}(X) = r_{n-1}h(X)g(X) + X[a(X)g(X) + s(X)]. \tag{3.69}$$

Переставляя слагаемые в (3.69), получим связь между синдромами $\tilde{s}(X)$ и $s(X)$

$$X \cdot s(X) = \underbrace{[c(X) + r_{n-1}h(X) + Xa(X)]}_{\text{сомножитель}} \cdot g(X) + \underbrace{\tilde{s}(X)}_{\text{остаток}}. \tag{3.70}$$

Из 3.70 непосредственно следует формулировка теоремы. ■

Пример: Вычисление синдрома однократных ошибок для циклического (7,4)-кода Хэмминга.

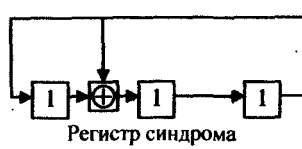


Рис. 3.15. Вычисление синдромов циклических сдвигов принятого слова.

Продолжим предыдущий пример. Из рис. 3.14 следует, что однократной ошибке в компоненте r_5 вектора $\mathbf{r} = (r_0, r_1, \ldots, r_6)$ соответствует синдром $\mathbf{s} = (s_0, s_1, s_2) = (1, 1, 1)$ (такт $n = 7$). Отключив входной регистр, получим схему, изображенную на рис. 3.15. Заметим, что исходное состояние на такте $n = 0$ определяется синдромом однократной ошибки в компоненте r_5. Произведя циклические сдвиги регистра синдрома рис. 3.15, мы на каждом такте будем находить

Таблица 3.6. Синдромы однократных ошибок циклического (7,4)-кода Хэмминга с порождающим многочленом $g(x) = 1 + X + X^3$.

Такт	s_0	s_1	s_2	Ошибочная компонента
0	1	1	1	r_5
1	1	0	1	r_6
2	1	0	0	r_0
3	0	1	0	r_1
4	0	0	1	r_2
5	1	1	0	r_3
6	0	1	1	r_4

$\tilde{s}(X)$ из (3.70). Последовательность $\tilde{s}(X)$ соответствует синдромам однократных ошибок в компонентах r_6, r_0 и т.д. (Табл. 3.6). Значения из таблицы 3.6 полностью совпадают со значениями таблицы 2.3, полученной для проверочной матрицы систематического (7,4)-кода Хэмминга.

Замечание. *В рассмотренном примере вся таблица синдромов однократных ошибок генерируется с помощью простейшей схемы, поэтому, для кодов большей длины можно хранить в памяти декодера таблицы синдромов, а не саму проверочную матрицу.*

Рассмотрим связь между синдромом $s(X)$ и многочленом ошибок $e(X)$. Из модели передачи информации (рис. 3.11) следует

$$r(X) = v(X) + e(X), \tag{3.71}$$

где

$$v(X) = c(X)g(X) = X^r u(X) + b(X). \tag{3.72}$$

Так как

$$r(X) = a(X)g(X) + s(X), \tag{3.73}$$

то

$$e(X) = [a(X) + c(X)]g(X) + s(X).^1 \tag{3.74}$$

[1] Из (3.74) следует, что $e(X) \bmod (g(X)) = s(X)$. Так как код Хэмминга исправляет все однократные ошибки, то для построения таблицы синдромов (n, k)-кода Хэмминга в качестве $e(X)$ достаточно перебрать все одночлены $X^0, X^1, \ldots, X^{n-1}$ – *Прим. перев.*

Знание синдрома не позволяет однозначно определить многочлен $e(X)$. Так, например, если $e(X)$ делится без остатка на $g(X)$, то этому случаю соответствует нулевой синдром и ошибка не может быть распознана. В этом случае говорят о *необнаружимой* или остаточной ошибке декодирования.

3.7. Пакеты ошибок

Характерной особенностью циклических кодов является способность к распознаванию *пакетов ошибок*. Под пакетом ошибок понимается группирование ошибок в одной ограниченной области кодового слова (рис. 3.16). Пакет ошибок можно описать многочленом вида

$$e(X) = X^j B(X). \tag{3.75}$$

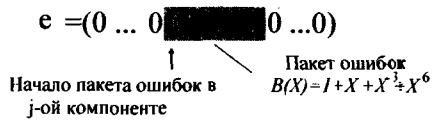


Рис. 3.16. Вектор пакета ошибок длины 7.

Если длина пакета ошибок не превосходит величины $r = n - k$, то степень многочлена ошибок меньше r. В этом случае $e(X)$ не делится на $g(X)$ без остатка и синдром принятого слова всегда отличен от нулевого, следовательно, пакет ошибок длины равной или меньшей r всегда распознается. Из теоремы 3.6.1 следует, что распознается также любой циклический сдвиг многочлена $B(X)$ степени, меньшей r, т.е. и «концевой» пакет ошибок длины меньшей или равной r (рис. 3.17), всегда распознается.

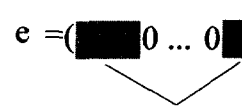


Рис. 3.17. Вектор «концевого» пакета ошибок длины 7.

Теорема 3.7.1. Циклический (n, k)-код способен обнаруживать все *пакеты ошибок* (в том числе *концевые*) длины $r = n - k$ и меньше.

Помимо распознавания всех пакетов ошибок длины r и меньше, циклические коды обладают способностью обнаруживать большую часть пакетов ошибок, длина которых превосходит r.

Рассмотрим пакет ошибок длины $r + 1$, начинающийся в j-ой компоненте. Так как первая и последняя компоненты пакета ошибок отличны от нуля, всего имеется 2^{r-1} возможных конфигураций ошибок. Необнаружимой является только одна из них, многочлен которой $B(X)$ совпадает с $g(X)$, то есть

$$e(X) = X^j B(X) = X^j g(X). \tag{3.76}$$

Теорема 3.7.2. Для циклического (n, k)-кода доля необнаружимых *пакетов ошибок* длины $l = r + 1 = n - k + 1$ равна $2^{-(r-1)}$.

Рассмотрим пакет ошибок длины $l > r + 1 = n - k + 1$, начинающийся в j-ой компоненте. Если соответствующий многочлен $B(X)$ делится на $g(X)$ без остатка, то есть

$$e(X) = X^j B(X) = X^j a(X) g(X), \tag{3.77}$$

то такой пакет не может быть обнаружен.

Пусть коэффициенты многочлена $a(X)$ степени $l - r - 1$ имеют вид $a_0, a_1, \ldots, a_{l-r-1}$. Так как пакет ошибок начинается и заканчивается единицей, $a_0 = a_{l-r-1} = 1$. Следовательно, существует 2^{l-r-2} наборов коэффициентов $a(X)$, приводящих к необнаружимым ошибкам в (3.77). С другой стороны, существует 2^{l-2} различных пакетов ошибок длины l и, таким образом, верно следующее утверждение.

Теорема 3.7.3. Для циклического (n, k)-кода доля необнаружимых *пакетов ошибок* длины $l > r + 1 = n - k + 1$ равна 2^{-r}.

Пример: Распознавание ошибок циклическим (7,4)-кодом Хэмминга.

Рассмотрим циклический (7,4)-код Хэмминга из предыдущих примеров с $r = n - k = 3$. Так как минимальное кодовое расстояние кода Хэмминга $d_{\min} = 3$, он способен обнаруживать все двойные ошибки или исправлять одиночные. Рассматриваемый (7,4)-код Хэмминга является циклическим кодом и он способен также обнаруживать

все пакеты длины $r = 3$. В частности, любые три следующие друг за другом ошибки всегда обнаруживаются.

Доля необнаружимых ошибок длины $r + 1 = 4$ равна $2^{-(3-1)} = 1/4$. При пакетах ошибок с длиной большей 4, не распознается только $2^{-3} = 1/8$ из них.

Замечание. *На практике, как правило, используются циклические коды с довольно большим числом проверочных разрядов, например,* $r = n - k = 16$. *Доля необнаружимых пакетов ошибок такими кодами достаточна мала. Так, при* $r = 16$, *обнаруживается более чем 99,9969 % пакетов длины 17 и 99,9984 % пакетов длины 18 и выше.*

3.8. Декодер Меггитта

Процесс декодирования циклических кодов (как и вообще всех линейных блоковых кодов) можно разделить на три этапа:

1. Вычисление синдрома;
2. Определение ошибочных компонент принятого слова;
3. Исправление ошибок или выдача сигнала о наличии неисправимых ошибок.

Оценивая сложность обработки принятого сигнала, можно заметить, что декодирование является «узким местом» в цепи передачи информации. Это связано с очень большими аппаратными затратами на декодирование длинных кодов с высокой корректирующей способностью.

Для циклических кодов процедура вычисления синдромов относительно проста. Из рис. 3.12 видно, что сложность вычисления синдрома мало зависит от длины кодового слова и определяется, в основном, числом проверочных символов.

Определение ошибочных компонент по синдрому для всех линейных блоковых кодов может быть сделано, в принципе, с помощью таблицы синдромов. Сложность реализации такого метода декодирования возрастает экспоненциально с ростом длины кодовых слов и корректирующей способности кода [7]. Здесь на помощь приходят некоторые особые свойства циклических кодов, которые позволяют существенно упростить процесс декодирования.

В настоящем разделе мы представляем достаточно простую схему декодера двоичных циклических (n, k)-кодов в общем виде. Будем

исходить из модели передачи информации рис. 3.11. В этой схеме принятый многочлен обозначается через $r(X)$, кодовый многочлен – через $v(X)$, а многочлен ошибок – через $e(X)$.

Первым шагом декодирования является вычисление синдрома. Здесь могут возникнуть два случая:

1. Найденный синдром соответствует ошибке в $(n-1)$-ой компоненте принятого слова, то есть $e_{n-1}=1$;

2. Синдром не соответствует такой ошибочной компоненте.

В последнем случае процесс, представленный на рис. 3.11 может быть повторно проделан для сдвинутого слова $r^{(1)}(X)$. При этом, для вычисления $s^{(1)}(X)$ нет необходимости проводить действия, аналогичные нахождению $s(X)$. Согласно теореме 3.6.1, синдром $s(X)$ преобразуется в синдром $s^{(1)}(X)$ за один такт с помощью схемы, представленной на рис. 3.15.

После того, как определен $s(X)$, мы можем последовательно получить синдромы $s^{(1)}(X), s^{(2)}(X), \ldots, s^{(n-1)}(X)$. Если синдромы не корректировать, то через n тактов мы опять придем к первоначальному синдрому $s(X)=s^{(n)}(X)$ многочлена $r(X)$. Однако, если в процессе работы получен синдром, соответствующий $e_{n-1}=1$, то некоторая компонента многочлена $r(X)$ должна быть исправлена. В этом случае также корректируется еще и некоторый синдром из последовательности $s^{(1)}(X), s^{(2)}(X), \ldots, s^{(n-1)}(X)$. Рассмотрим простейший случай. Пусть синдром $s(X)$ сразу же соответствует $(n-1)$-ой ошибочной компоненте многочлена $r(X)$, то есть $e_{n-1}=1$. Тогда скорректированный многочлен $\eta(X)$ будет иметь вид:

$$\eta(X)=r_0+r_1X+\cdots+(r_{n-1}+e_{n-1})X^{n-1}. \qquad (3.78)$$

Заметим, что на последующих шагах декодирования должен использоваться синдром модифицированного многочлена из (3.78). Так как $s(X)$ является синдромом $r(X)$, а $s'(X)$ – синдром многочлена $e'(X)=X^{n-1}$, синдром многочлена $\eta(X)$ равен

$$s_1(X)=s(X)+s'(X). \qquad (3.79)$$

Коррекцию синдрома удобнее производить только на следующем такте декодирования. В этом случае, после циклического сдвига $r(X)$ ошибочной нулевой компоненте будет соответствовать многочлен

$e'(X) = 1$, а синдром скорректированного многочлена $\eta^{(1)}(X)$ будет равен

$$\tilde{s}_1^{(1)}(X) = s^{(1)}(X) + 1. \tag{3.80}$$

Контроль ошибок производится по всем n компонентам принятого многочлена $r(X)$ и найденные ошибки исправляются. Если после n тактов контроля ошибок синдром отличен от нулевого, то выдается сигнал о наличии неисправляемых ошибок. Такой алгоритм может быть, в принципе, использован для декодирования всех циклических (n, k)-кодов. Его реализация, *декодер Меггитта*, в общем виде представлена на рис. 3.18.

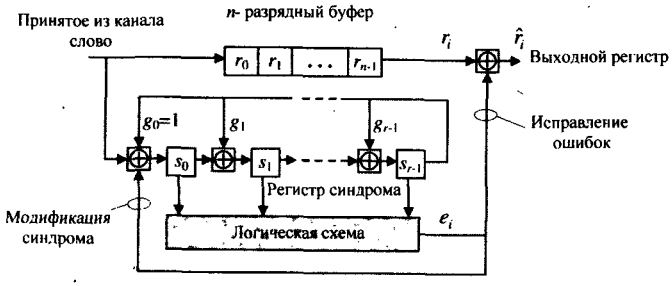


Рис. 3.18. Декодер Меггитта для двоичного циклического (n, k)-кода с порождающим многочленом $g(X)$.

Перед началом работы оба регистра декодера Меггитта обнуляются. На первых n тактах декодирования принятое слово загружается в верхний регистр, а в нижнем регистре вычисляется синдром многочлена $r(X)$. После этого вход декодера отключается и в течении последующих n тактов проводится покомпонентное обнаружение и исправление ошибок. В каждом такте с помощью логической схемы распознавания ошибок проверяется соответствие текущего синдрома ошибочной $(n-1)$-ой компоненте сдвинутого многочлена $r(X)$. При обнаружении такого соответствия на следующем такте декодирования, ошибка исправляется, а синдром модифицируется.

Пример: Декодер Меггитта циклического (7,4)-кода Хэмминга.

В качестве примера рассмотрим уже знакомый нам циклический (7,4)-кода Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$. Напомним, что код Хэмминга позволяет исправлять все однократные ошибки. Более того, так как (7,4)-код Хэмминга является совершенным, таблица синдромов однократных ошибок (см. табл.

3.6) содержит все возможные синдромы от «001» до «111» (синдром 000 интерпретируется как отсутствие ошибок). Для построения декодера Меггитта нам достаточно знать синдром одиночной ошибки в компоненте r_6 принятого слова. Из таблицы 3.6 следует, что синдром одиночной ошибки $r_6 = 1$ равен «101» или при полиномиальном представлении $1 + X^3$. Схема декодера Меггитта циклического (7,4)-кода Хэмминга приведена на рис. 3.19. Блок распознавания ошибок может быть построен при помощи логической схемы совпадения «И» (&) с инверсией входа S_1 (в соответствии с синдромом «101» для $r_6 = 1$).

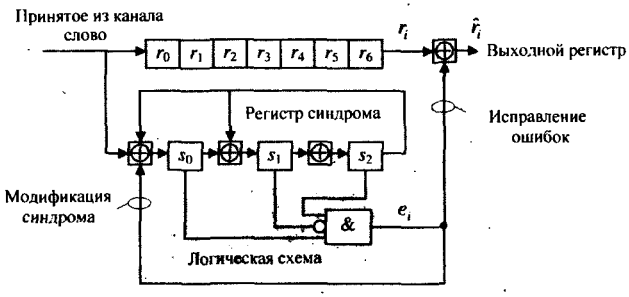


Рис. 3.19. Декодер Меггитта циклического (7,4)-кода Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$.

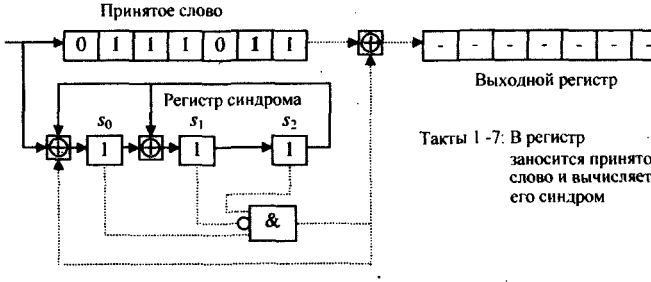


Рис. 3.20. Декодер Меггитта циклического (7,4)-кода Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$ и принятым вектором $\mathbf{r}$ (начальная фаза).

Рассмотрим процесс декодирования принятого слова с ошибочной компонентой r_5. В этом случае $\mathbf{r} = (011\ 1011)$. После первых семи тактов работы декодера слово $r(X)$ полностью занесено в буферный регистр, а в нижнем регистре находится синдром $s(X)$ для

$r(X)$ (рис. 3.20). В соответствии с таблицей 3.6 этот синдром равен «111».

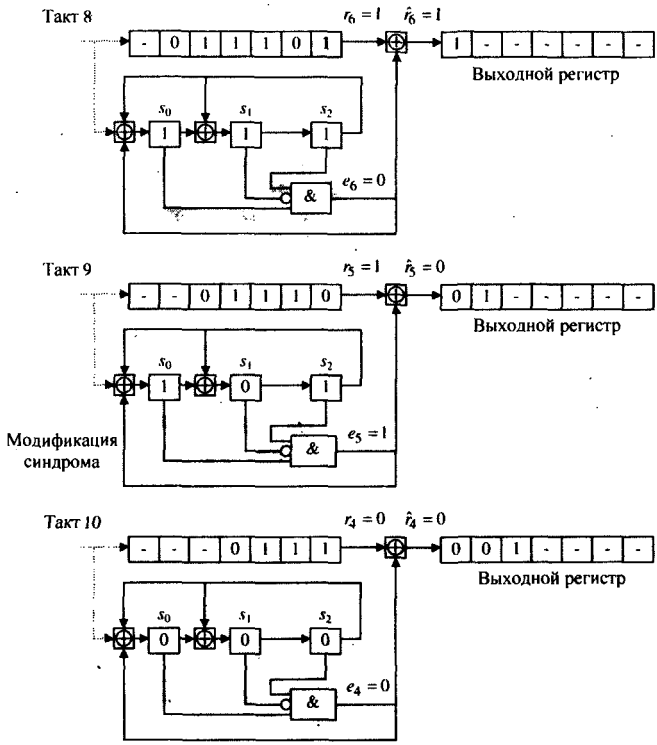


Рис. 3.21. Декодер Меггитта циклического (7,4)-кода Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$ и принятым вектором **r** (заключительная фаза).

На восьмом такте декодирования компонента r_6 поступает в выходной регистр без изменения (поскольку синдром был равен «111»), а новое значение синдрома равно «101», что соответствует ошибке в компоненте r_5 и на девятом такте эта ошибка корректируется, а синдром обнуляется (рис. 3.21). Таким образом, на тактах $n = 10 \ldots 14$ оставшиеся компоненты принятого слова загружаются в выходной регистр без коррекции, так как синдром сохраняет нулевое значение.

3.9. Циклические коды Хэмминга

Циклические коды Хэмминга образует важное семейство циклических (n, k)-кодов. Свойства кодов Хэмминга, как подмножества линейных кодов, были подробно рассмотрены в разделе 2.4. Коды Хэмминга, имеющие циклическую структуру, обладают дополнительными, весьма полезными свойствами.

При определении методов построения циклических кодов используются понятия неприводимых и примитивных многочленов.

Многочлен $p(x)$ степени m называется *неприводимым в поле* $GF(2)$, если он не делится ни на какой многочлен с коэффициентами из $GF(2)$ степени меньшей m, но большей 0.

Неприводимый многочлен $p(x)$ степени m называется *примитивным*, если наименьшая степень n, при которой многочлен $X^n + 1$ делится на $p(x)$ без остатка, равна $n = 2^m - 1$.

В таблице 3.7 приведены некоторые примитивные многочлены с коэффициентами из $GF(2)$.

Замечание. *Примитивные многочлены играют важную роль в технике передачи сообщений, например, примитивный многочлен степени $m = 23$ используется в устройствах перемешивания символов в сетях ISDN и xDSL. Примитивные многочлены являются также основой для построения порождающих многочленов псевдослучайных последовательностей. С помощью таких псевдослучайных последовательностей производится адресование сообщений в системах мобильной связи.*

Основой для построения циклических кодов Хэмминга служит следующая теорема.

Теорема 3.9.1. Любой *циклический код Хэмминга* длины $2^m - 1$ с $m \geq 3$ может быть построен с помощью некоторого примитивного многочлена степени m.

Имеется также обратная теорема. В ней утверждается, что любому примитивному многочлену степени m соответствует некоторый циклический код Хэмминга длины $2^m - 1$ [7].

Пример: Циклический (15,11)-код Хэмминга.

Для построения циклического (15,11)-кода Хэмминга используется многочлен степени степени $m = 4$. Из таблицы 3.7 следует, что

Таблица 3.7. Примитивные многочлены.

m	$p(X)$	m	$p(X)$
3	$1+X+X^3$	11	$1+X^2+X^{11}$
4	$1+X+X^4$	12	$1+X+X^4+X^6+X^{12}$
5	$1+X^2+X^5$	13	$1+X+X^3+X^4+X^{13}$
6	$1+X+X^6$	14	$1+X^2+X^6+X^{10}+X^{14}$
7	$1+X^3+X^7$	15	$1+X+X^{15}$
	$1+X^2+X^3+X^4+X^5+X^6+X^7$		$1+X+X^2+X^3+X^4+X^{12}+X^{13}+X^{14}+X^{15}$
8	$1+X^2+X^3+X^4+X^8$	23	$1+X^5+X^{23}$
9	$1+X^4+X^9$	32	$1+X+X^2+X^4+X^5+X^7+X^8+X^{10}+X^{11}+X^{12}+X^{16}+$
10	$1+X^3+X^{10}$		$+X^{22}+X^{23}+X^{26}+X^{32}$

таким многочленом является многочлен $p(X) = 1 + X + X^4$ и используя его в качестве порождающего, можно получить все кодовые слова циклического (15,11)-кода Хэмминга.

С помощью порождающих многочленов $g(X)$ можно строить порождающие и проверочные матрицы, а также схемы декодеров любых циклических кодов Хэмминга.

3.10. Двоичный код Голлея

Используя равенство (2.25) можно заметить, что $[C_{23}^0 + C_{23}^1 + C_{23}^2 + C_{23}^3]2^{12} = 2^{23}$. Возникает гипотеза о том, что 23-мерное двоичное пространство можно плотно упаковать сферами радиуса 3. Это равенство представляет собой необходимое (но недостаточное) условие существования совершенного (23,12)-кода, исправляющего все тройные ошибки. Такой код действительно существует, его удалось построить швейцарцу Голею в 1949 году. В силу особенностей своей алгебраической структуры, он является весьма притягательным для математиков различным направлений и, поэтому, подробно описан в литературе [8].

В основе конструкции кода лежит разложение

$$X^{23} + 1 = (1 + X)g_1(X) \cdot g_2(X), \tag{3.81}$$

в котором $g_1(X)$ и $g_2(X)$ представляют собой порождающие многочлены *кода Голлея*, причем,

$$g_1(X) = 1 + X^2 + X^4 + X^5 + X^6 + X^{10} + X^{11} \tag{3.82}$$

и

$$g_2(X) = 1 + X + X^5 + X^6 + X^7 + X^9 + X^{11}. \tag{3.83}$$

Коды Голлея можно декодировать с помощью *декодера с вылавливанием ошибок*. Этот метод использует вычисление синдрома и может быть реализован на регистрах сдвига с помощью схем, аналогичных рис. 3.21. В [7] описана модификация декодера кода Голлея с вылавливанием ошибок, позволяющая корректировать исправляемые, но не вылавливаемые ошибки, предложенная Т. Касами.

3.11. CRC коды

Примером использования семейства циклических кодов является контроль ошибок с помощью циклического избыточного кода, то есть *CRC кода* (Cyclic redundancy check), называемого также кодом *Абрамсона*. При передаче данных в пакетных режимах, эти коды используются для определения целостности блоков данных (FCS – Frame Checking Sequence). Примером систем с FCS являются стандарты передачи данных X.25 (HDSL), ISDN, DECT и LAN. CRC коды представляют собой расширения циклических кодов Хэмминга.

Пусть $p(X)$ – примитивный многочлен степени m, тогда порождающий многочлен CRC кода $g(X)$ можно записать в виде произведения

$$g(X) = (1 + X)p(X). \tag{3.84}$$

С помощью порождающего многочлена $g(X)$ может быть построен циклический CRC (n, k)-код с параметрами $n = 2^m - 1$, $k = 2^m - m - 2$, имеющий $m + 1$ проверочных символов и $d_{\min} = 4$.[1]

CRC-коды обладают пятью важными свойствами:

[1] CRC код отличается от расширения кода Хэмминга, описанного в разделе 2.4.5. Расширенный $(2^m, 2^m - m - 1)$-код получается из циклического $(2^m - 1, 2^m - m - 1)$-кода Хэмминга присоединением проверки на четность по всем символам и имеет минимальное расстояние, также равное 4. Этот код не является циклическим. Хотя в CRC $(2^m - 1, 2^m - m - 2)$-коде также добавлена дополнительная проверка на четность, длина кода не увеличилась, так как при этом был исключен один информационный символ. В результате CRC код представляет собой совокупность кодовых векторов четного веса первоначального кода Хэмминга и по-прежнему остается циклическим - *Прим. перев.*

Таблица 3.8. Порождающие многочлены CRC кодов.

Код	Порождающий многочлен $g(X)$
CRC-4	$1 + X + X^4$ (3.85)
	Используется в ISDN
CRC-8	$(1+X)(1+X^2+X^3+X^4+X^5+X^6+X^7) = 1+X+X^2+X^8$ (3.86)
	Используется в ATM в качестве HEC
CRC-12	$(1+X)(1+X^2+X^{11}) = 1+X+X^2+X^3+X^{11}+X^{12}$ (3.87)
CRC-16	$(1+X)(1+X^+X^{15}) = 1+X^2+X^{15}+X^{16}$ (3.88)
	IBM
CRC-16	$(1+X)(1+X+X^2+X^3+X^4+X^{12}+X^{13}+X^{14}+X^{15}) = 1+X^5+X^{12}+X^{16}$ (3.89)
	Является стандартом CCITT для HDLC и LAPD
CRC-32	$1+X+X^2+X^4+X^5+X^7+X^8+X^{10}+X^{11}+X^{12}+X^{16}+X^{22}+X^{23}+X^{26}+X^{32}$ (3.90)
	Используется в HDLC

1. Все ошибки кратности 3 или меньше обнаруживаются;
2. Все ошибки нечетной кратности обнаруживаются;
3. Все пакеты ошибок длины $l = m + 1$ или меньше обнаруживаются;
4. Доля необнаружимых пакетов ошибок длины $l = m + 2$ составляет 2^{-m};
5. Доля необнаружимых пакетов ошибок длины $l \geq m + 3$ составляет $2^{-(m-1)}$.

Все перечисленные свойства позволяют эффективно использовать CRC код при передачи данных с переспросами (протокол ARQ).

На практике часто используются укороченные CRC коды. В таблице 3.8 приведены наиболее употребляемые порождающие многочлены CRC кодов, а также указаны области их применения.

3.12. Укороченные коды

Во всех рассмотренных нами циклических кодах длина кодовых слов однозначно определяется степенью выбранного примитивного многочлена. Это обстоятельство накладывает большие ограничения на число информационных разрядов в кодируемом блоке. Между тем, в используемых в настоящее время стандартах передачи данных, длина информационных блоков может колебаться в довольно широких пределах. В соответствии с этим, кодирование также должно быть достаточно гибким.

Здесь на помощь приходят *укороченные коды*, построенные на основе циклических кодов. Пусть, например, нами выбран многочлен с $m = 5$. На базе этого многочлена можно построить циклический (31,26)-код Хэмминга. Рассмотрим подмножество слов этого кода, содержащее все кодовые слова с тремя нулями в старших разрядах[1] Это подмножество образует укороченный (28,23)-код Хэмминга. Укороченный код сохраняет все свойства циклического (31,26)-кода, так как в процессе декодирования мы можем дописать к кодовым словам три недостающих нуля и рассматривать их как векторы основного (31,26)-кода Хэмминга.

В качестве следующего примера можно привести *код Файера*, используемый в мобильной связи. Этот код является сильным укорочением кода Хэмминга. Соответствующий код Хэмминга имеет непомерно большую длину, равную 3014633 и содержит 40 проверочных символов. На его базе строится (224,184)-код Файера, способный обнаруживать все пакеты ошибок длиной до 40 символов, или исправлять все пакеты длиной до 12 символов. Этот код может эффективно бороться с замираниями и используется в мобильной связи GSM для защиты канала управляющей информации SACCH (Slow Associated Control Channel).

[1]Для определенности, здесь и в дальнейшем будем считать, что кодовый вектор поступает в канал начиная со старших разрядов, которые соответствуют старшим степеням многочлена. – *Прим. перев.*

Таблица 3.9. (6,3)-код.

Вход	Кодовое слово
000	000 000
100	110 100
010	011 010
110	101 110
001	111 001
101	001 101
011	100 011
111	010 111

Из примера кода Файера видно, что использование при кодировании простого дополнения информационного слова нулями практически неприемлемо, поэтому должен применяться более эффективный метод кодирования укороченных кодов.

Пример: Укороченный (6,3)-код.

Рассмотрим метод кодирования и декодирования кода, полученного укорочением (7,4)-кода Хэмминга. Выберем из множества кодовых слов базового кода все кодовые слова, начинающиеся с символа «0». Выбрасываемые при укорочении символы полагаются равными нулю и, поэтому, не передаются. Таким образом получаем укороченный (6,3)-код. В левой части таблицы 3.3 приведены кодовые слова (7,4)-кода Хэмминга, у которых старший разряд равен нулю. Выбрасывая из этих слов лишние нули, получаем кодовые слова укороченного (6,3)-кода (см. табл. 3.9).

Замечание. *Построенный (6,3)-код не является циклическим, так как циклический сдвиг кодовых слов не во всех случаях является кодовым словом, однако, его конструкция позволяет использовать свойства циклических кодов при декодировании.*

Проверим таблицу 3.9 с помощью алгоритма деления Евклида. Рассмотрим информационный вектор $\mathbf{u} = (101)$, многочлен которого $u(X) = 1 + X^2$. Умножая $u(X)$ на X^3 и используя алгоритм деления Евклида (см. табл. 3.10), получим многочлен остатка от деления $X^3u(X)$ на $g(X) = 1 + X + X^3$, равный $b(X) = X^2$. Таким образом,

Таблица 3.10. Определение проверочных символов (алгоритм деления Евклида) для $u(X) = 1 + X^2$ и $g(X) = 1 + X + X^3$.

X^5	X^4	X^3	X^2	X	1	
1	0	1	0	0	0	$= X^3u(X)$
1	0	1	1	0	0	$= X^2g(X)$
-	-	-	1	0	0	$= X^3u(X) + X^3g(X) = b(X)$

многочлен систематического (6,3)-кода равен

$$v(X) = X^3u(X) + b(X) = X^3[X^2 + 1] + X^2 = X^5 + X^3 + X^2, \quad (3.91)$$

что соответствует кодовому вектору $\mathbf{v} = (001\ 101)$ из табл. 3.9.

За основу кодера систематического (6,3)-кода может быть принята схема кодирования, приведенная на рис. 3.8. Так как (6,3)-код образуется укорочением (7,4)-кода Хэмминга, схема рис. 3.8 существенно упрощается: из регистров удаляются разряды u_3 и v_6 и кодирование заканчивается уже после третьего такта.

Для практической реализации декодера укороченного кода имеются три альтернативы:

1. Для декодирования (6,3)-кода можно, в принципе, использовать декодер базового (7,4)-кода Хэмминга (см. рис. 3.20). В этом случае, к принятому слову приписывается недостающий ноль и процесс декодирования занимает столько же тактов, сколько требуется для декодера базового кода.

   Для декодирования кодов с l-кратным укорочением необходимо затратить l дополнительных тактов. В случае, когда l велико (как, например, в случае кода Файера) такой метод декодирования неприемлем.

2. При коррекции ошибок и модификации синдрома принимаются во внимание особенности конструкции укороченного кода. Схема декодера (6,3)-кода приведена на рис. 3.22. Здесь вектору ошибки $\mathbf{e} = (000\ 001)$ в компоненте r_5, согласно табл. 3.6, соответствует синдром $\mathbf{s} = (1, 1, 1)$, поэтому, схема распознавания ошибок настраивается на этот синдром (а не на синдром (101) в случае (7,4)-кода). Рассмотрим теперь алгоритм модификации синдрома. В нижнем регистре вычисляются синдромы всех сдвигов ошибки в слове базового (7,4)-кода (эта схема

«не знает об укорочении») и следующим за синдромом «111» будет синдром «101». В соответствии с его числовым вектором и производится модификация текущего синдрома на рис. 3.22.

Замечание. *В силу линейности кодов и схем их реализации, рассмотренный метод декодирования может быть применен для любых укорочений кода Хэмминга.*

Такой метод декодирования, в ряде случаев, приводит к достаточно простой схеме построения декодеров и, поэтому, весьма интересен для практики. В качестве примера в [4] приведена схема декодирования укороченного (272,260)-кода.

3. Этот вариант построения декодера Меггита особенно интересен для практики, так как используемые в нем алгоритмы распознавания ошибок и коррекции синдрома не зависят от длины укорочения l и строятся с наименьшими затратами. Длина укорочения учитывается путем умножения принятого слова на некоторый многочлен, зависящий от l.

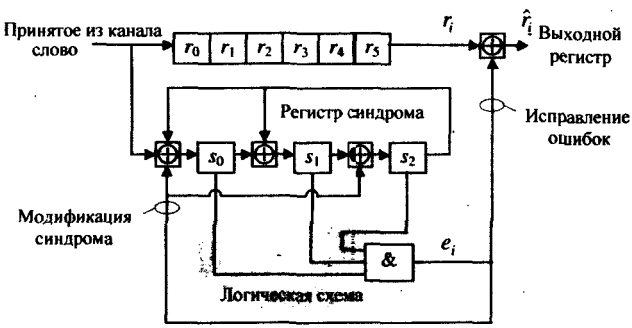


Рис. 3.22. Декодер Меггитта укороченного (6,3)-кода Хэмминга.

На примере укороченного (6,3)-кода мы покажем, как могла бы выглядеть оптимальная процедура распознавания ошибки в компоненте r_5 и коррекции соответствующего синдрома. Как уже отмечалось ранее, в нижнем регистре рис. 3.22 производится вычисление синдромов циклических сдвигов вектора ошибки для неукороченного (7,4)-кода Хэмминга, поэтому таблицу 3.6 можно использовать и для (6,3)-кода с учетом того, что первым вычисляется синдром для

$\mathbf{e}_5 = (000\ 0010)$. На последующих тактах в нижнем регистре вычисляются синдромы для

$$\mathbf{e}_6 = e_5^{(1)} = (000\ 0001), \qquad \mathbf{e}_0 = e_5^{(2)} = (100\ 0000),$$

$$\mathbf{e}_1 = e_5^{(3)} = (010\ 0000), \qquad \mathbf{e}_2 = e_5^{(4)} = (001\ 0000)$$

и т. д. Наиболее благоприятным с точки зрения затрат на модификацию в схеме рис. 3. 22 является синдром $\mathbf{s} = (0, 0, 1)$. Из таблицы 3.6 следует, что этот синдром соответствует вектору ошибки

$$\mathbf{e}_2 = \mathbf{e}_5^{(4)} = (001\ 0000),$$

т.е. ошибке в компоненте r_2. Таким образом, для реализации оптимальной процедуры исправления ошибки и коррекции синдрома в схеме рис. 3.22 необходимо предварительно произвести четыре циклических сдвига верхнего и нижнего регистров (при этом верхний регистр должен быть модифицирован: в него должны быть добавлен дополнительный разряд r_6 и цепь обратной связи (Прим. переводчика)).

На примере укороченного (6,3)-кода может показаться, что эти затраты не слишком велики, однако, для используемого в сети GSM кода Файера миллион и более сдвигов плюс затраты на модификацию регистра оказываются технически неприемлемыми. Ниже мы теоретически покажем, что дополнительные затраты такого рода отнюдь не являются необходимыми. После этого мы рассмотрим построение схемы оптимального декодера укороченного (6,3)-кода.

Будем искать схему построения декодера укороченного кода, в которой технические затраты на исправление ошибки и коррекцию синдрома были бы минимальными. Так как базовый (n, k)-код является циклическим, эта схема должна обнаруживать и исправлять ошибку в младшем разряде кодового слова укороченного $(n-l, k-l)$-кода. Многочлен такой ошибки имеет вид

$$e(X) = X^{n-1-l}, \tag{3.92}$$

где l – длина укорочения.

Постараемся найти линейную операцию, отображающую $e(X)$ в некоторый вспомогательный многочлен $e'(X)$, синдром которого при $e(X) = X^{n-1-l}$ был бы равен

$$s'(X) = X^{n-k-1}. \tag{3.93}$$

Пусть такая операция отображает многочлен $e(X) = X^{n-1-l}$ в

$$e'(X) = X^{n-k-1}. \tag{3.94}$$

Так как степень многочлена $e'(X)$ меньше степени $g(X)$, равенство (3.93) остается справедливым. Заметим, что $e'(X) = X^{n-k-l}$ можно получить из многочлена $e(X) = X^{n-1-l}$, i-кратным циклическим сдвигом, то есть

$$e'(X) = e^{(i)}(X), \tag{3.95}$$

где

$$i = n - k + l. \tag{3.96}$$

При этом после $l+1$ первых сдвигов $e'(X) = X^0$, а после оставшихся $n - k - 1$ сдвигов выполняются равенства (3.94) и (3.93). Постараемся заменить операцию i-кратного циклического сдвига линейной операцией умножения многочленов.

Из соотношения (3.9) следует, что в нашем случае справедливо равенство

$$X^i e(X) = q(X)(X^n + 1) + e^{(i)}(X), \tag{3.97}$$

где $q(X)$ – некоторый многочлен, конкретное значение которого не представляет для нас интереса. Согласно теореме 3.2.5, порождающий многочлен $g(X)$ делит $X^n + 1$ без остатка, поэтому можно записать

$$(X^n + 1) = a_1(X)g(X). \tag{3.98}$$

Применяя алгоритм деления Евклида, сомножитель X^i можно представить в виде

$$X^i = a_2(X)g(X) + d(X), \tag{3.99}$$

где $\deg[d(X)] < \deg[g(X)]$.

Подставляя (3.98) и (3.99) в (3.97), имеем

$$[a_2(X)g(X) + d(X)]e(X) = q(X)a_1(X)g(X) + e^{(i)}(X). \tag{3.100}$$

Из (3.100) следует

$$e^{(i)}(X) = [q(X)a_1(X) + a_2(X)e(X)]g(X) + d(X)e(X). \tag{3.101}$$

Равенство (3.101), на первый взгляд, может показаться довольно сложным, однако оно сильно упрощается при вычислении синдромов обоих частей равенства. Следует заметить, что выражение в

квадратных скобках умножено на $g(X)$ и, следовательно, это произведение не оказывает влияние на синдром в правой части (3.101). Таким образом, мы имеем

$$[d(X)e(X)] \mod g(X) = [e^{(i)}(X)] \mod g(X) = s'(X). \quad (3.102)$$

Обобщая все вышесказанное, можно сделать следующие выводы:

- Ошибка обнаруживается при ее сдвиге в старший разряд кодового слова укороченного $(n-l, k-l)$-кода и исправляется на следующем такте работы декодера;
- В схеме декодера вычисляется синдром вспомогательного многочлена $e'(X) = e(X)d(X)$. В этом случае ошибке $e(X) = X^{n-1-l}$ соответствует синдром $s'(X) = X^{n-k-1}$, который также корректируется на следующем такте;
- Многочлен $d(X)$ является остатком от деления X^i на $g(X)$, где i определяется длиной укорочения l и степенью производящего многочлена $g(X)$;
- Так как степень многочлена $d(X)$ всегда меньше степени $g(X)$, схема умножения $e(X)$ на $d(X)$ может быть реализована с помощью простых сдвигов.

Пример: Укороченный (6,3)-код.

Рассмотрим построение оптимального, с точки зрения затрат, декодера укороченного (6,3)-код. Напомним, что этот код образуется укорочением базового циклического (7,4)-кода Хэмминга с порождающим многочленом $g(X) = 1 + X + X^3$. В этом случае многочлену ошибки $e(X) = X^5$ должен соответствовать синдром вспомогательного многочлена $s'(X) = X^2$. Так как X^2 является четырехкратным циклическим сдвигом многочлена $e(X) = X^5$, множитель $d(X)$ определяется как

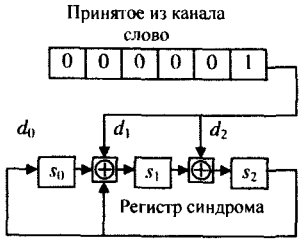
$$d(X) = [X^4] \mod (X^3 + X + 1) = X^2 + X, \quad (3.103)$$

и вспомогательный многочлен $e'(X)$ равен

$$d(X)e(X) = X^2e(X) + Xe(X). \quad (3.104)$$

Заметим, что умножение на $d(X)$ в (3.104) соответствует линейной комбинации однократного и двукратного сдвигов вектора ошибки.

Таким образом, в схеме рис. 3.23 вектор ошибки через сумматоры одновременно подается на разряды s_1 и s_2 регистра вычисления синдрома. На рис. 3.23 приведен пример вычисления синдрома для $\mathbf{e} = (0,0,0,0,0,1)$, то есть синдрома ошибки, находящейся в старшем разряде укороченного (6,3)-кода. Как и следовало ожидать, после шестого такта мы получаем синдром $\mathbf{s}' = (0,0,1)$.



Такт	s_0	s_1	s_2
0	0	0	0
1	0	1	1
2	1	1	1
3	1	0	1
4	1	0	0
5	0	1	0
6	0	0	1

Рис. 3.23. Модифицированный алгоритм вычисления синдрома с предварительным умножением вектора ошибки на $d(X) = X^2 + X$.

Схема исправления однократной ошибки и коррекции синдрома с использованием вспомогательного многочлена $e'(X)$ приведена на рис. 3.24. По вычисленному синдрому $\mathbf{s}' = (0,0,1)$ происходит обнаружение и исправление ошибки. Одновременно синдром корректируется и, тем самым, устраняется влияние ошибки на последующие шаги декодирования.

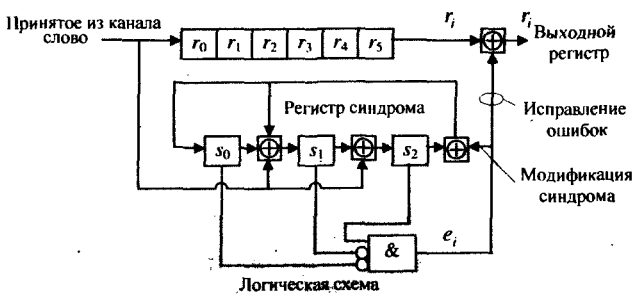


Рис. 3.24. Декодер Мегитта для укороченного (7,4)-кода Хэмминга с предварительным умножением на $d(X) = X^2 + X$.

Таблица 3.11. Вычисление синдрома неискаженного принятого слова $\mathbf{r}_1 = (0,1,1,0,1,0)$.

Такт	d_0	d_1	d_2	s_0	s_1	s_2	
0	-	0	0	0	0	0	При поступлении старшего разряда в d_1 и d_2 регистр синдрома обнуляется
1	-	1	1	0	0	0	
2	-	0	0	0	1	1	
3	-	1	1	1	1	1	
4	-	1	1	1	1	0	
5	-	0	0	0	0	0	
6	-	-	-	0	0	0	Синдром указывает на отсутствие ошибки

Таблица 3.12. Вычисление синдрома неискаженного принятого слова $\mathbf{r}_1 = (0,1,1,0,1,\mathbf{1})$.

Такт	d_0	d_1	d_2	s_0	s_1	s_2	
0	-	**1**	**1**	0	0	0	При поступлении старшего разряда в d_1 и d_2 регистр синдрома обнуляется
1	-	1	1	0	1	1	
2	-	0	0	1	0	0	
3	-	1	1	0	1	0	
4	-	1	1	0	1	0	
5	-	0	0	0	1	0	
6	-	-	-	0	0	1	Синдром указывает на ошибку в старшем разряде

Проверим работу декодера на трех примерах. Пусть кодовое слово $\mathbf{r}_1 = (0,1,1,0,1,0)$ (см. табл. 3.9) принято без ошибок. Последовательность состояний регистра синдрома приведена в табл. 3.11. После окончания загрузки слова $\mathbf{r}_1$ в буфферный регистр, в нижнем регистре также заканчивается вычисление его синдрома. В данном случае синдром оказывается нулевым.

Теперь в старший разряд слова $\mathbf{r}_1$ внесем одну ошибку и получим вектор $\mathbf{r}_2 = (0,1,1,0,1,\mathbf{1})$. После загрузки слова $\mathbf{r}_2$ в буфферный регистр, его синдром равен $\mathbf{s}'_2 = (0,0,1)$ (см. табл. 3.12). Согласно алгоритму декодирования, такой синдром указывает на ошибку в

Таблица 3.13. Вычисление синдрома искаженного принятого слова $\mathbf{r}_3 = (0,1,\mathbf{0},0,1,0)$.

Такт	d_0	d_1	d_2	s_0	s_1	s_2	
0	-	0	0	0	0	0	При поступлении старшего разряда в d_1 и d_2 регистр синдрома обнуляется
1	-	1	1	0	0	0	
2	-	0	0	0	1	1	
3	-	**0**	**0**	1	1	1	
4	-	1	1	1	0	1	
5	-	0	0	1	1	1	
6	-	-	-	1	0	1	Синдром указывает на наличие ошибки

старшем разряде слова $\mathbf{r}_2$.

И, наконец, рассмотрим декодирование принятого слова с ошибкой в третьей компоненте: $\mathbf{r}_3 = (0,1,\mathbf{0},0,1,0)$. Синдром принятого слова $\mathbf{r}_3$ равен $\mathbf{s}_3' = (1,0,1)$ (см. табл. 3.13). Это означает, что в принятом слове имеется ошибка, но она произошла не в старшем разряде, поэтому продолжим процедуру вычислений синдромов теперь уже для сдвигов слова $\mathbf{r}_3$ (см. табл. 3.14).

Эта процедура продолжается до тех пор, пока ошибочная компонента $\mathbf{r}_3$ не займет место старшего разряда. После этого синдром принимает значение $\mathbf{s}_3' = (0,0,1)$ и происходит исправление ошибки и коррекция синдрома. Далее декодирование происходит уже с нулевым синдромом, то есть после исправления ошибки в третьей компоненте мы получили слово (6,3)-кода.

3.13. Пример применения: ATM

Одним из примеров применения CRC кодов является передача данных по технологии ATM (Assynchronous Transfer Mode – Асинхронный режим передачи).

Подход, реализованный в технологии ATM, состоит в представлении потока данных от каждого канала любой природы пакетами фиксированной длины в 53 байта вместе с небольшим заголовком в 5 байт, из которых 3 байта отводятся под номер виртуального соединения, уникального в пределах всей сети ATM, а остальные 48 байт

Таблица 3.14. Декодирование искаженного принятого слова $\mathbf{r}_3 = (0, 1, \mathbf{0}, 0, 1, 0)$ после вычисления синдрома.

Такт	Принятое слово	s_0 s_1 s_2	Декодированное слово	Коментарий
0	0	1 0 1	0	
1	1	1 0 0	1	
2	0	0 1 0	0	
3	**0**	0 0 1	**1**	Ошибка в старшем разряде исправляется и синдром модифицируется
4	1	0 0 0	1	
5	0	0 0 0	0	Однократная ошибка была исправлена

могут содержать либо 6 замеров оцифрованного голоса либо 6 байт данных. Пакеты ATM называются ячейками – cell. Формат такой ячейки представлен на рис. 3.25.

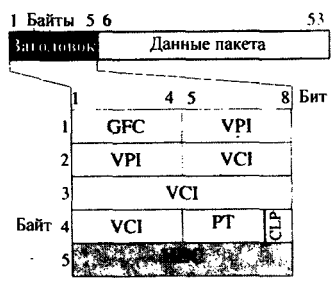


Рис. 3.25. Формат ячейки ATM.

Вся управляющая информация заголовка, занимающая 4 байта, защищена от ошибок CRC-8 кодом из табл. 3.8. Восемь проверочных разрядов кода CRC записываются в последний пятый байт заголов-

ка и называются полем управления ошибками в заголовке (HEC – Heder Error Control).

Порождающий многочлен CRC-8 кода представляет собой произведение $g(X) = (1 + X)p(X)$, где $p(X)$ – примитивный многочлен степени m. Для CRC-8 кода степень этого многочлена $m = 7$, длина кода $n = 2^m - 1 = 127$ и код содержит $m + 1 = 8$ проверочных разрядов. Так как защищаемая CRC кодом информация составляет 32 бита, в стандарте ATM используется укороченный код длины $n = 40$ и скорость кода равна $R = 32/40 = 0,8$. Такой укороченный (40,32)-код обладает высокой корректирующей способностью.

Оценим вероятность необнаружимой ошибки декодирования. Укороченный (40,32)-код содержит 2^{32} кодовых слов и имеет $d_{\min} = 4$. Будем считать, что ошибки, происходящие в канале, независимы и происходят с вероятностью P_e. Воспользуемся достаточно грубой верхней оценкой вероятности необнаружимой ошибки для линейных блоковых кодов (2.30), получим .

$$P_{r,1} = 2^{32} P_e^4. \tag{3.105}$$

Эта верхняя оценка не учитывает особых свойств CRC кодов (см.раздел 3.11). Из этих свойств следует, что при наличии в принятом слове четырех и более ошибок, доля необнаружимых ошибок декодирования не превышает $2^{-m} = 2^{-7}$. Учитывая также, что все ошибки нечетной кратности обнаруживаются, эта доля ошибок снижается до 2^{-8}, и, с учетом свойств CRC (40,32)-кода, верхняя оценка вероятности необнаружимой ошибки декодирования равна

$$P_{r,2} = 2^{24} P_e^4. \tag{3.106}$$

На рис. 3.26 приведена зависимость оценок (3.105) и (3.106) от вероятности ошибки в двоичном символе P_e. При передаче информации по оптоволокну, среднее значение P_e равно 10^{-9} и вероятность необнаружимой ошибки декодирования не превышает $2 \cdot 10^{-29}$.

Для того, чтобы представить себе, насколько мала эта величина, приведем наглядный пример. Пусть информация передается со стандартной для ATM скоростью – 622,08 Мбит/сек, то есть каждую секунду в канал связи поступает $1,47 \cdot 10^6$ блоков, тогда, при непрерывной передаче, необнаружимая ошибка будет в среднем возникать один раз в 10^{15} лет.

Реальные системы передачи данных, конечно же, не всегда соответствуют рассмотренной модели. При использовании оптоволокна качество канала может резко ухудшаться в некоторые моменты

времени. Это вызывает появление в канале пакетов ошибок большой длины. Процесс передачи данных в такой системе можно описать с помощью диаграммы переходов с двумя состояниями (см. рис. 3.27). Этим состояниям соответствуют два режима работы декодера. В режиме исправления однократной ошибки декодер находится в том случае, если ранее, в течении определенного времени, ошибки в канале не фиксировались (то есть все принятые слова имели нулевой синдром). В этом случае предполагается «хорошее» состояние канала, в котором вероятность одиночной ошибки в блоке намного превышает вероятность многократных ошибок. Декодер остается в этом состоянии до тех пор, пока его синдром не становится отличным от нуля. В этом случае декодер переходит в режим обнаружения ошибок. Если, при этом, синдром соответствует одиночной ошибке, то она исправляется, если обнаруживается наличие ошибки большей кратности, то весь принятый блок стирается.

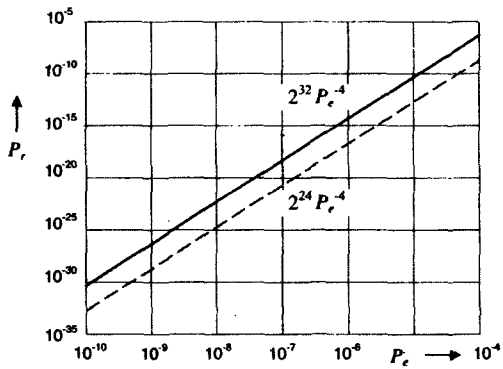


Рис. 3.26. Оценка вероятности необнаружимой ошибки.

В режиме обнаружения ошибок, попытки обнаружить и исправить ошибки уже не производятся, так как предполагается «плохое» качество канала. В этом режиме все ошибочные блоки АТМ стираются. Декодер может опять вернуться в режим исправления одиночных ошибок в блоках АТМ, если в течении некоторого времени он не будет обнаруживать ошибки в принятых символах.

Замечание. *Стертые блоки АТМ не пропадают. Согласно протоколу передачи данных, в случае стирания блоков, в канал обратной связи поступает запрос на их повторную передачу.*

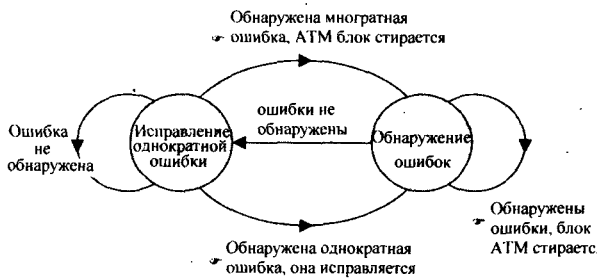


Рис. 3.27. Модель передачи блоков ATM по каналу с двумя состояниями.

В заключении рассмотрим еще одно применение укороченного (40,32)-кода для *управления ошибками в заголовке (HEC)*. В силу высокой способности этого кода к обнаружению ошибок, HEC может использоваться также для поддержания блоковой синхронизации. Опишем этот алгоритм с помощью диаграммы состояний рис. 3.28.

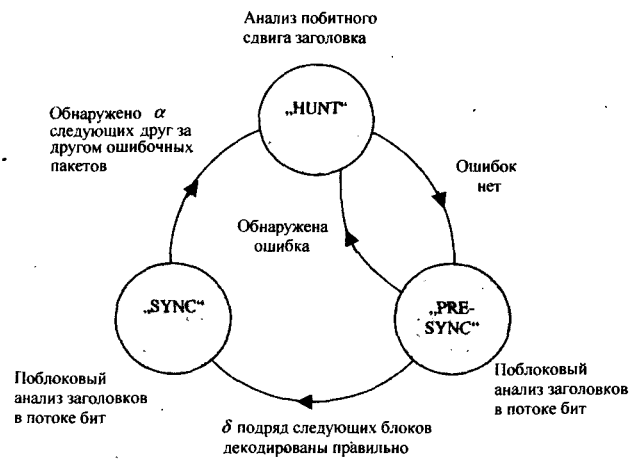


Рис. 3.28. Модель применения (40,32)-кода для синхронизации при передаче по технологии ATM.

В начале, в состоянии «HANT» или *поиска по времени*, декодер

делает попытки декодирования принимаемого потока бит, побитно сдвигая предполагаемое начало блока ATM. Если синдром указывает на возможно правильное декодирование заголовка блока, то декодер переходит в режим «PRESYNC» (*предсинхронизация*). Заголовки в принятом потоке блоков ATM декодируются, и, если при этом была обнаружена ошибка, декодер опять возвращается в состояние «HANT». Если же на протяжении δ блоков все заголовки были продекодированы правильно, декодер переходит в состояние «SYNC» (*синхронизация*). В этом состоянии обработка информации происходит согласно рис. 3.27. Синхронизация может быть утрачена из-за ошибок в канале, что обнаруживается декодером при наличии последовательных α ошибочных блоков. В этом случае декодер опять переходит в состояние поиска по времени «HANT».

Рассмотренные примеры показывают, что применение помехоустойчивых кодов может далеко выходить за рамки только обнаружения и исправления ошибок. Помехоустойчивые коды могут в корне менять принципы построения систем передачи данных и улучшать их технические характеристики, поэтому, знание принципов помехоустойчивого кодирования и областей его применения может оказать существенную помощь при разработке новых систем связи.

3.14. Упражнения

Задача 3.1: Код CRC.

1. Найдите порождающий многочлен CRC (7,3)-кода;
2. Приведите схему кодера систематического (7,3)-кода;
3. Рассмотрите процесс кодирования информационного вектора $\mathbf{u} = (0, 1, 0, 1)$ с помощью этой схемы. Найдите содержимое регистра кодера на каждом шаге кодирования;
4. С помощью соответствующего кодового многочлена проверьте правильность, кодового слова, полученного в п. 3;
5. Приведите схему декодера, обнаруживающего ошибки и объясните алгоритм обнаружения ошибки в принятом слове;
6. Проверьте работу декодера для случая, когда принято слово $\mathbf{r} = (0, 0, 1, 0, 1, 0, 0, 1)$;
7. Проведите декодирование принятого слова из п. 6 при помощи порождающей матрицы;
8. Найдите максимальную длину обнаруживаемого пакета ошибок;
9. Приведите пример концевого пакета ошибок максимальной длины.

Решение.

1. Рассмотрим примитивный многочлен третьей степени: $p(X) = X^3 + X + 1$ (заметим, что $p(X)$ является порождающим многочленом циклического (7,4)-кода Хэмминга). Порождающий многочлен CRC (7,3)-кода равен

$$g(X) = (1 + X)(1 + X + X^3) = X^4 + X^3 + X^2 + 1. \qquad (3.107)$$

2. Схема кодера CRC (7,3)-кода представлена на рис. 3.29.

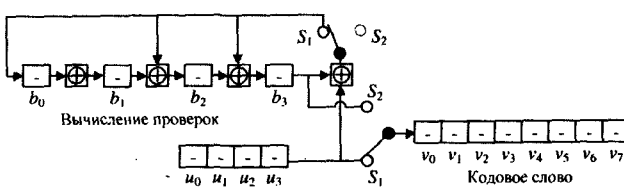


Рис. 3.29. Кодер систематического CRC (7,3)-кода.

3. Работа кодера при $\mathbf{u} = (0,1,0,1)$ представлена в табл. 3.15. Результатом является кодовое слово $\mathbf{v} = (0,1,1,0,0,1,0,1)$.

Таблица 3.15. Вычисление проверочных символов (7,3) CRC кода с помощью схемы рис.3.29.

Такт	Сообщение	b_0	b_1	b_2	b_3	Коментарий
0	0101	0	0	0	0	Инициализация
1	010	1	0	1	1	
2	01	1	1	1	0	
3	0	1	1	0	0	
4	-	0	1	1	1	Проверочные символы

4. Вычисление кодового многочлена $v(X)$ для информационного слова $\mathbf{u} = (0,1,0,1)$ в систематическом виде производится следующим образом:

$$\begin{aligned} u(X) &= X^3 + X \\ b(X) &= X^4 u(X) \mod g(X) = X^2 + X \\ v(X) &= X^4 u(X) + b(X) = X^7 + X^5 + X^2 + X, \end{aligned} \tag{3.108}$$

что соответствует кодовому слову $\mathbf{v} = (0,1,1,0,0,1,0,1)$.

5. Так как CRC (7,3)-код является циклическим, то при отсутствии ошибки в канале принятый многочлен $r(X)$ должен делиться на порождающий многочлен $g(X)$ без остатка. Если в результате работы декодера получен ненулевой синдром, то с уверенностью можно утверждать, что принятое слово является ошибочным. Схема вычисления синдрома представлена на рис. 3.30.

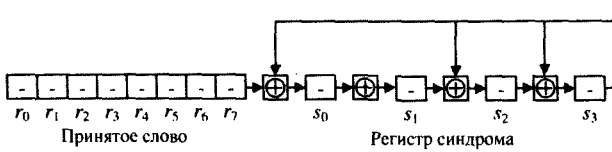


Рис. 3.30. Схема вычисления синдрома CRC (7,3)-кода.

6. Работа декодера для принятого слова $\mathbf{r} = (0,0,1,0,1,0,0,1)$ (см. рис. 3.30) представлена в табл. 3.16 по тактам.

Таблица 3.16. Вычисление синдрома для принятого слова $\mathbf{r} = (0, 0, 1, 0, 1, 0, 0, 1)$.

Такт	Сообщение	s_0	s_1	s_2	s_3	Коментарий
4	0010	1	0	0	1	Инициализация
5	001	1	1	1	1	
6	00	0	1	0	0	
7	0	0	0	1	0	
8	-	0	0	0	1	Ошибка !

7. Представим принятое слово $\mathbf{r} = (0, 0, 1, 0, 1, 0, 0, 1))$ в виде многочлена и найдем остаток от деления многочлена $r(X)$ на $g(X)$

$$\begin{aligned} r(X) &= X^7 + X^4 + X^2 \\ s(X) &= r(X) \mod g(X) = X^3, \end{aligned} \tag{3.109}$$

что соответствует синдрому, получаемому в предыдущем пункте.

8. Так как циклический CRC (7,3)-код образуется с помощью порождающего многочлена $g(X)$ четвертой степени, то все пакеты длины 4 и менее обнаруживаются.

9. «Концевой» пакет ошибок длины 4 $\mathbf{e} = (0, 1, 0, 0, 0, 0, 1, 1)$.

Задача 3.2: Укороченный (12,8)-код Хэмминга.

Рассмотрим (12,8)-код, являющийся укорочением циклического (15,11)-кода Хэмминга. Начертите блок-схему декодера (12,8)-кода оптимального с точки зрения сложности, исправляющего одну ошибку.

Решение.

Для минимизации сложности, выбернем схему декодера Меггитта, использующего вспомогательный многочлен $e'(X)$. Напомним, что $e'(X)$ образуется умножением многочлена ошибки $e(X)$ на $d(X)$, где $d(X) = X^i \mod g(X)$. В рассматриваемом примере

$$\begin{aligned} i &= n - k + l = 15 - 11 + 3 = 7 \\ g(X) &= 1 + X + X^4, \end{aligned} \tag{3.110}$$

следовательно, многочлен $d(X)$ равен

$$d(X) = X^i \mod g(X) = 1 + X + X^3. \tag{3.111}$$

Блок-схема декодера приведена на рис. 3.31.

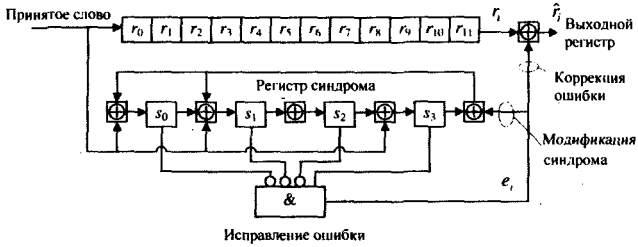


Рис. 3.31. Декодер Меггитта для укороченного (12,8)-кода Хэмминга.

ГЛАВА 4

СВЕРТОЧНЫЕ КОДЫ

4.1. Введение

В современной информационной технике сверточные коды играют такую же важную роль как и блоковые. Первые попытки сопоставления сверточных и блоковых кодов были сделаны в 50-е годы. С этого времени блоковые коды быстро нашли эффективное применение, в то время как сверточные коды оставались на заднем плане. Этот процесс продолжался до тех пор, пока в 1967 г. не был найден эффективный алгоритм декодирования сверточных кодов. Сегодня сверточные коды играют ведущую роль в современных системах связи. Это в первую очередь относится к цифровому радиовещанию и к мобильной связи сети GSM. Сверточные коды позволяют также эффективно бороться с помехами, вызванными многолучевым распространением радиоволн. В последние годы сверточные коды получили дальнейшее развитие в связи с открытием турбо-кодов. Достаточно сказать, что использование турбо-кодов в современных системах передачи данных позволило достичь скоростей передачи информации близких к пропускным способностям каналов. В связи с этим весьма неожиданным обстоятельством в мобильных сетях радиовещания был принят стандарт UMTS (Universal Mobile Telecommunication Sistem – англ.).

Важнейшими отличиями сверточных кодов от блоковых являются следующие:

1. Сверточные коды позволяют производить кодирование и декодирование потоков данных непрерывно во времени.

2. Сверточные коды не нуждаются в блоковой синхронизации.

3. Применение сверточных кодов позволяет достичь очень высокой надежности передаваемой информации.

4. «Хорошие» сверточные коды могут быть найдены путем моделирования.

Подробное изложение теории сверточных кодов и областей их применения выходит за рамки этой книги. В данной главе мы ограничимся изложением только самых необходимых теоретических основ и приведем типичные примеры применения сверточных кодов. Более подробное описание можно найти, например [5],[18],[16].

4.2. Кодер и импульсный отклик

Термин *«сверточные коды»* возник из теории инвариантных линейных систем LTI (Linear Time Invariant – англ). В теории систем LTI сверткой называют характерный признак некоторой линейной операции. С точки зрения этой теории, кодирование является отображением информационной последовательности символов в кодовую последовательность с помощью линейной схемы с параметрами, не меняющимися во времени. Такое отображение наглядно показано на рис. 4.1. Последовательность информационных символов поступает в демультиплексор, который разлагает входной поток на k самостоятельных подпоследовательностей. Схему рис. 4.1 можно также интерпретировать как совместное кодирование k независимых информационных последовательностей. Кодирование производится с помощью дискретной во времени схемы LTI с k входами и n выходами. Эта схема характеризуется тремя параметрами (n, k, m), причем, параметр m определяется внутренней конструкцией кодера.

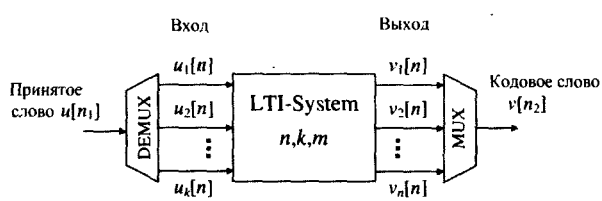


Рис. 4.1. Схема LTI с k входами и n выходами как кодер сверточного кода.

На практике, как правило, используются двоичные сверточные коды, поэтому, в дальнейшем мы будем говорить о последовательностях битов. В этом случае, под линейностью схемы мы подразумева-

ем выполнение этой схемой всех операций по правилам арифметики по модулю 2. (т.е. в поле $GF(2)$).

Так как декодер представляет собой дискретную схему, для его описания будем использовать методы *теории обработки дискретных сигналов*. Придерживаясь терминологии этой теории, будем говорить об информационной последовательности $u[n]$, входных последовательностях $u_1[n]$, $u_2[n]$, ..., $u_k[n]$, выходных последовательностях $v_1[n]$, $v_2[n]$, ..., $v_k[n]$ и кодовой последовательности $u[n_2]$. Важнейшим параметром схемы LTI является импульсный отклик. Смысл этого параметра мы раскроем ниже.

В схеме, изображенной на рис. 4.1, все выходные последовательности объединяются в одну кодовую последовательность с помощью мультиплексора MUX. Это происходит путем поочередного считывания символов $v_1[n], v_2[n], \ldots, v_n[n]$ при каждом фиксированном значении n.

Для того, чтобы записать связь между параметрами кодера рис 4.1 в компактной форме и при этом избежать путаницы, введем единые обозначения. Фигурными скобками будем обозначать любую последовательность символов, причем, нижние индексы используются для нумерации элементов последовательностей. Для импульсного отклика системы первый нижний индекс определяет номер выходной последовательности, а второй – номер входной.

- *Входные последовательности* $\{u_j[n]\} = \{u_{j,0}, u_{j,1}, \ldots\}$, при $j = 1, \ldots, k$;
- *Выходные последовательности* $\{v_j[n]\} = \{v_{j,0}, v_{j,1}, \ldots\}$, при $j = 1, \ldots, n$;
- *Информационная последовательность* $\{u[n]\} = \{\underbrace{u_{1,0}, u_{2,0}, \ldots, u_{k,0}}_{k \text{ символов}}, \underbrace{u_{1,1}, u_{2,1}, \ldots, u_{k,1}}_{k \text{ символов}}, \ldots\}$;
- *Кодовая последовательность* $\{v[n]\} = \{\underbrace{v_{1,0}, v_{2,0}, \ldots, v_{n,0}}_{n \text{ символов}}, \underbrace{v_{1,1}, v_{2,1}, \ldots, v_{n,1}}_{n \text{ символов}}, \ldots\}$;
- *Импульсный отклик* $\{g_{ji}[n]\} = \{g_{ji,0}, g_{ji,1}, \ldots, g_{ji,m_i}\}$

С учетом введенных обозначений операцию *свертки*, выполняемую схемой рис. 4.1, можно записать в виде

$$v_j[n] = \sum_{i=1}^{k} g_{ji}[n] \circledast u_i[n] = \sum_{i=1}^{k} \sum_{m=0}^{m_i} g_{ji}[m] \cdot u_i[n-m]. \qquad (4.1)$$

Замечание. *Здесь символ «⊛» обозначает операцию линейной свертки. В дальнейшем мы будем рассматривать также циклическую свертку «∗».*

Эта операция выполняется для всех выходов j. Здесь индексы, стоящие в квадратных скобках, определяют нормированные переменные времени, то есть они указывают номера элементов соответствующих последовательностей. Нумерация элементов, как правило, начинается с нуля, поэтому $u_2[5]$, например, обозначает 6-ой элемент второй входной последовательности. Элементы с отрицательной временной переменной полагаются равными нулю.

Замечание. *В дальнейшем, если это не будет приводить к путанице, лишние индексы будем опускать.*

Пример: Свертка двоичных последовательностей.

Выберем импульсный отклик системы, равный

$$\{g[n]\} = \{1, 0, 1, 1\}. \tag{4.2}$$

Для входной последовательности

$$\{u[n]\} = \{1, 0, 1\} \tag{4.3}$$

получим свертку

$$v[n] = g[n] \circledast u[n] = \sum_{m=0}^{3} g[m]g[n-m]. \tag{4.4}$$

Найдем элементы $v[n]$

$$\begin{aligned}
v[0] &= g[0] \odot u[0] = 1 \odot 1 = 1 \\
v[1] &= (g[0] \odot u[1]) \oplus (g[1] \odot u[1]) = \\
&= (1 \odot 0) \oplus (0 \odot 1) = 0 \oplus 0 = 0 \\
v[2] &= (g[0] \odot u[2]) \oplus (g[1] \odot u[1]) \oplus (g[2] \odot u[0]) = \\
&= (1 \odot 1) \oplus (0 \odot 0) \oplus (1 \odot 1) = 1 \oplus 0 \oplus 1 = 0 \\
v[3] &= (g[1] \odot u[2]) \oplus (g[2] \odot u[1]) \oplus (g[3] \odot u[0]) = \\
&= (0 \odot 1) \oplus (1 \odot 0) \oplus (1 \odot 1) = 0 \oplus 0 \oplus 1 = 1 \\
v[4] &= (g[2] \odot u[2]) \oplus (g[3] \odot u[1]) = \\
&= (1 \odot 1) \oplus (1 \odot 0) = 1 \oplus 0 = 1 \\
v[5] &= g[3] \odot u[2] = 1 \odot 1 = 1.
\end{aligned} \tag{4.5}$$

Обратим внимание на подробное обозначение операций в GF(2). В дальнейшем такое обозначение может быть опущено, если смысл операций следует из контекста.

Рассмотрим теперь реальную схему кодера сверточного кода.

Пример: Кодер сверточного (2,1,3)-кода.

Изображенный на рис. 4.2 кодер имеет $n = 2$ выходов, $k = 1$ входов и $m = 3$ (так как регистр кодера содержит три разряда s_0, s_1, s_2). С точки зрения теории цифровой обработки сигналов – это нерекурсивная система с удвоением.

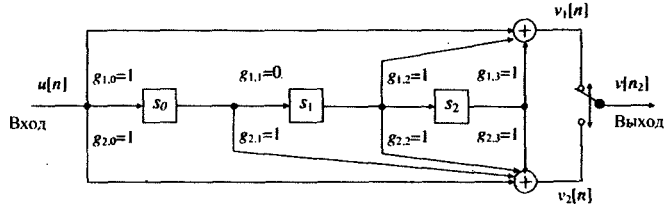


Рис. 4.2. Кодер сверточного (2,1,3)-кода.

Две выходные последовательности $v_1[n]$ и $v_2[n]$ можно генерировать с помощью двух фильтров с конечными импульсными откликами $g_1[n]$ и $g_2[n]$ из входной последовательности $u[n]$. Импульсные отклики $g_1[n]$ и $g_2[n]$ можно получить непосредственно из рис. 4.2. Учитывая, что при отсутствии связи между двоичными разрядами регистра и выходами кодера соответствующие коэффициенты равны нулю, имеем

$$\{g_1[n]\} = \{1, 0, 1, 1\} \text{ и } \{g_2[n]\} = \{1, 1, 1, 1\}, \tag{4.6}$$

что эквивалентно двум многочленам третьей степени.Задержка в регистре сдвига кодера также равна трем.

На выходе кодера две последовательности $v_1[n]$ и $v_2[n]$ объединяются в кодовую последовательность $v[n_2]$. Разность в обозначениях временных нормированных переменных n и n_2 объясняется удвоением частоты кодовой последовательности по сравнению с частотой информационной последовательности.

Работу схемы кодера объясним на следующем примере. Пусть

$$\{u[n]\} = \{1, 0, 1\}, \tag{4.7}$$

тогда

$$\{v_1[n]\} = \{1, 0, 0, 1, 1, 1\} \tag{4.8}$$

и

$$\{v_2[n]\} = \{1, 1, 0, 1, 1\}. \tag{4.9}$$

Объединяя эти последовательности между собой, на выходе кодера получим

$$\{v[n_2]\} = \{1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1\}. \tag{4.10}$$

Рассматривая схему кодера в общем случае определим важнейшие параметры сверточных кодов. Так как в основе любого кодера лежит регистр сдвига, то один бит входной информации оказывает влияние на процесс кодирования на протяжении нескольких тактов. Здесь возникают такие понятия, как память кодера и длина кодового ограничения.

Так как в общем случае кодер может иметь несколько входов, то *память кодера* определяется как

$$m = \max_{i=1,\ldots,k} m_i. \tag{4.11}$$

Кодовое ограничение является производным от памяти и определяет ширину области кодового слова, на которую влияет один входной бит, то есть

$$n_c = n(m + 1). \tag{4.12}$$

Как будет показано далее, параметры m и n_c оказывают решающее влияние на корректирующую способность кода и сложность алгоритма декодирования.

Следующим важнейшим параметром является *скорость кода* R. Так как в общем случае k входным битам соответствуют n выходных, то

$$R = \frac{k}{n}. \tag{4.13}$$

Как правило, число входов k невелико, поэтому, типичным являются кодовые скорости от $1/3$ до $7/8$.

В современных сетях связи режимы обмена информацией очень сильно зависят от ее характера, поэтому, необходимо учитывать, что отдельные реальные сообщения могут иметь различные конечные длины. В связи с этим, в некоторых случаях реальные кодовые скорости могут быть существенно меньшими, чем R из (4.13). Это вызвано тем обстоятельством, что для сохранения корректирующей способности кода мы вынуждены добавлять в конце сообщения

«*хвост*» из m нулей. Таким образом, если длина сообщения составляет L бит, то реальная (блоковая) скорость равна

$$R_B = \frac{kL}{n(L+m)} \approx R \text{ для } L \gg m. \tag{4.14}$$

Для сообщений небольшой длины приходится считаться с *относительной потерей скорости*, равной

$$\frac{m}{L+m}. \tag{4.15}$$

Замечание. *Для очень коротких сообщений имеет смысл использовать при декодировании алгоритм «нейтрализации хвоста». Этот алгоритм при незначительном снижении корректирующей способности кода позволяет полностью избежать вставки m нулей в конце информационных блоков.*

4.3. Полиномиальное представление

Аналогично блоковым кодам (глава 3), сверточные коды могут быть описаны с помощью многочленов. При этом становятся очевидными не только сходство этих кодов, но и их различия.

Будем рассматривать импульсные отклики кодеров сверточных кодов как *порождающие многочлены* степени m_i

$$g_{ji}(X) = g_{ji,0} + g_{ji,1}X + \cdots + g_{ji,m_i} + X^{m_i}. \tag{4.16}$$

Переменная X здесь играет роль «указателя сдвига» и никакой смысловой нагрузки больше не несет, X^n означает n-кратный сдвиг относительно некоторой точки отсчета (например, начала входной последовательности).

Замечание. *В литературе вместо переменной X очень часто используют переменную D (от слова delay – задержка – англ.).*

Так как мы по прежнему рассматриваем только двоичные кóды, все коэффициенты многочленов принадлежат $GF(2)$ и все операции над многочленами выполняются по правилам арифметики по модулю 2.

Аналогично блоковым кодам, процесс кодирования сверточных кодов может быть описан с помощью порождающих многочленов (4.16). Если входная последовательность имеет конечную длину, то

мы фактически имеем дело с блоковым кодированием, поэтому, j-ю выходную последовательность можно представить в виде

$$v_j(X) = \sum_{i=1}^{k} g_{ji}(X)u_i(X). \tag{4.17}$$

Так как имеется n выходов и *кодовый многочлен* образуется их перемешиванием, удобно записать

$$v(X) = \sum_{j=1}^{n} X^{j-1}v_j(X^n). \tag{4.18}$$

В этом случае

$$v(X) = \sum_{j=1}^{n} X^{j-1} \sum_{i=1}^{k} g_{ji}(X^n)u_i(X^n). \tag{4.19}$$

Поясним сказанное на примере сверточного (2,1,3)-кода.

Пример: Представление сверточного (2,1,3)-кода в виде многочлена.

В соответствии с рис. 4.2, имеем

$$g_1 = 1 + X^2 + X^3 \text{ и } g_2 = 1 + X + X^2 + X^3. \tag{4.20}$$

В качестве информационной последовательности выберем

$$\{u[n]\} = \{1, 0, 1, 1, 1\}. \tag{4.21}$$

Этой последовательности соответствует многочлен

$$u(X) = 1 + X^2 + X^3 + X^4, \tag{4.22}$$

поэтому,

$$\begin{aligned} v_1(X) &= u(X) \cdot g_1(X) = \\ &= (1 + X^2 + X^3 + X^4)(1 + X^2 + X^3) = 1 + X^7 \\ v_2(X) &= u(X) \cdot g_2 = 1 + X + X^3 + X^4 + X^5 + X^7. \end{aligned} \tag{4.23}$$

Согласно (4.8), кодовый многочлен определяется как

$$\begin{aligned} v(X) &= v_1(X^2) + Xv_2(X^2) = \\ &= 1 + X + X^3 + X^7 + X^9 + X^{11} + X^{14} + X^{15}, \end{aligned} \tag{4.24}$$

что соответствует кодовому слову

$$\{v[n_2]\} = \{1,1,0,1,0,0,0,1,0,1,0,1,0,0,1,1\}. \quad (4.25)$$

Для того, чтобы показать связь между блоковыми и сверточными кодами, представим процесс кодирования сверточных кодов в виде произведения информационного вектора $\mathbf{u} = (u_1, u_2, u_3 \cdots)$ на порождающую матрицу $\mathbf{G}$

$$\mathbf{v} = \mathbf{u} \odot \mathbf{G} = \mathbf{u} \begin{pmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \mathbf{G}_2 & \cdots & \mathbf{G}_m & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_{m-1} & \mathbf{G}_m & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{G}_0 & \cdots & \mathbf{G}_{m-2} & \mathbf{G}_{m-1} & \mathbf{G}_m & \mathbf{0} & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}. \quad (4.26)$$

Так как информационный вектор может иметь бесконечную длину, матрица $\mathbf{G}$ не ограничена справа и снизу, поэтому, ее часто называют полубесконечной. Если вектор $\mathbf{u}$ конечен, то мы имеем дело с несколько специфичным блоковым кодом.

Порождающая матрица $\mathbf{G}$ составлена из регулярных подматриц $\mathbf{G}_0, \mathbf{G}_1, \cdots, \mathbf{G}_m$, каждая из которых определяется импульсным откликом кодера

$$\mathbf{G}_l = \begin{pmatrix} g_{11,l} & g_{21,l} & \cdots & g_{n1,l} \\ g_{12,l} & g_{22,l} & \cdots & g_{n2,l} \\ \vdots & \vdots & \ddots & \vdots \\ g_{1k,l} & g_{2k,l} & \cdots & g_{nk,l} \end{pmatrix}. \quad (4.27)$$

Рассмотрим построение матрицы $\mathbf{G}$ на примере сверточного (2,1,3)-кода.

Пример: Порождающая матрица сверточного (2,1,3)-кода.

Рассмотрим уже знакомый нам сверточный код с параметрами $k = 1, n = 2, m = 3$, импульсные отклики которого равны

$$\{g_1[n]\} = \{1,0,1,1\} \text{ и } \{g_2[n]\} = \{1,1,1,1\}. \quad (4.28)$$

Этот код имеет $m + 1 = 4$ внутренних подматриц, которые определены как

$$\begin{aligned} \mathbf{G}_0 &= (g_{11,0} \;\; g_{21,0}) = (1 \;\; 1) \\ \mathbf{G}_1 &= (g_{11,1} \;\; g_{21,1}) = (0 \;\; 1) \\ \mathbf{G}_2 &= (g_{11,2} \;\; g_{21,2}) = (1 \;\; 1) \\ \mathbf{G}_3 &= (g_{11,3} \;\; g_{21,3}) = (1 \;\; 1). \end{aligned} \quad (4.29)$$

Тогда порождающая матрица равна

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}. \tag{4.30}$$

Самым простым способом проверки правильности (4.30) является кодирование информационного вектора $(100\cdots0)$. В этом случае, результатом должны быть чередующиеся друг с другом импульсные отклики кодера. Таким образом

$$\begin{aligned} (1\ 0\ 0\ \cdots) \cdot \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} &= \\ = (1\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0 \cdots), & \end{aligned} \tag{4.31}$$

что и совпадает с ожидаемым результатом.

С другой стороны, из свойства линейности схемы кодера следует, что все строки матрицы равны, и, начиная со второй, сдвинуты на n позиций относительно предыдущих. Это обстоятельство еще раз подтверждает правильность найденного решения.

4.4. Граф состояний

Регистры кодеров содержат ограниченное число двоичных разрядов, следовательно, число состояний, в котором может находиться кодер, всегда конечно. Именно поэтому процесс кодирования можно описать как последовательность смены состояний кодера. Такой подход является ключевым и ведет к более глубокому пониманию свойств сверточных кодов. Более того, он способствует разработке эффективных алгоритмов кодирования и декодирования. Мы разовьем эту идею на нескольких примерах.

Пример: Описание состояний сверточного (2,1,3)-кода.

Скопируем схему кодера рис. 4.2 с единственной разницей: вместо разрядов регистра сдвига поставим переменные X_1, X_2 и X_3. Такая модифицированная схема представлена на рис. 4.3.

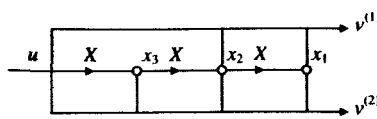


Рис. 4.3. Граф потока состояний кодера сверточного (2,1,3)-кода.

Так как X_1, X_2, X_3 - двоичные переменные, то кодер, изображенный на рис. 4.3, может находиться в одном из $2^3 = 8$ состояний. Эти состояния пронумерованы в лексикографическом порядке и приведены в таблице 4.1. Нумерация состояний может быть произвольной, однако, лексикографический порядок наиболее удобен для программной реализации.

Таблица 4.1. Таблица состояний.

Состояние регистра			Состояние S_i
x_3	x_2	x_1	$i = x_3 + 2x_2 + 2^2x_1$
0	0	0	0
1	0	0	1
0	1	0	2
1	1	0	3
0	0	1	4
1	0	1	5
0	1	1	6
1	1	1	7

Смена состояний регистра кодера происходит следующим образом: переменные X_1 и X_2 заменяются на X_2 и X_3 соответственно, а в разряд X_3 загружается новый информационный бит, поэтому, каждое состояние может переходить только в два последующих в зависимости от того, «0» или «1» были поданы на вход регистра сдвига. Все возможные смены состояний, их зависимость от загружаемого бита, а также кодовые символы для этих переходов показаны на рис. 4.4. Процесс кодирования начинается с S_0, т.е. состояния, в котором $X_1 = X_2 = X_3 = 0$.

1. Если первый информационный бит равен «0», кодер остается

в состоянии S_0. На выход выдаются кодовые символы «00». Такой исход на рис. 4.4 обозначен через «0/00».

2. Если первый информационный бит равен «1», кодер переходит в состояние S_1 (см табл. 4.1). Выход в этом случае равен «11». Этот переход обозначен «1/11». Продолжая этот процесс вправо, можно построить всю диаграмму состояний (рис. 4.4).

Диаграмма рис. 4.4 содержит всю информацию о сверточном коде. Кодирование информационной последовательности эквивалентно движению по некоторому неразрывному пути по диаграмме состояний. При программной реализации, например, кодирование может быть наиболее эффективно осуществлено исключительно с помощью заранее записанных в памяти таблиц переходов между состояниями. Рассмотрим такое альтернативное кодирование на примере.

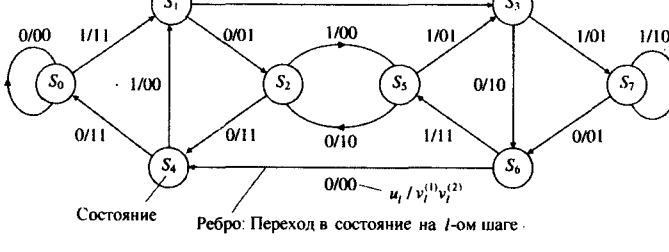


Рис. 4.4. Диаграмма состояний сверточного (2,1,3)-кода.

Пример: Кодирование сверточного (2,1,3)-кода с помощью таблиц переходов.

Выпишем для каждого состояния два его последующих, в зависимости от значения очередного бита («0» или «1»). Для этих путей выпишем также соответствующие пары кодовых бит. Полученные результаты сведены в таблицу 4.2.

Рассмотрим кодирование информационной последовательности $u[n] = 1, 0, 1, 1, 1$. Процесс начнем из состояния S_0 и в нем же и закончим, добавив к $u[n]$ «хвост» из трех нулей. В результате получим последовательность состояний

$$\{S[n]\} = \{S_0, S_1, S_2, S_5, S_3, S_7, S_6, S_4, S_0\} \tag{4.32}$$

и кодовое слово

$$\{v[n]\} = \{1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1\}. \tag{4.33}$$

Таблица 4.2. Таблица переходов для сверточного $(2, 1, 3)$-кода.

Состояние	Новое состояние $S[n+1]$		Кодовые биты	
	0	1	0	1
0	0	1	00	11
1	2	3	01	10
2	4	5	11	00
3	6	7	10	01
4	0	1	11	00
5	2	3	10	01
6	4	5	00	11
7	6	7	01	10

Таблица переходов в некоторых случаях может помочь оценить качество выбранного сверточного кода. Сравним, например, построчно пары кодовых бит в последних двух столбцах таблицы 4.2. Мы увидим, что различие между этими парами (расстояние Хэмминга) в каждой строчке максимально и равно 2. Это значит, что на начальных отрезках любых двух альтернативных путей («0» и «1») всегда достигается максимальное расстояние Хэмминга и, поэтому, мы можем сказать, что, на первый взгляд, выбранный код является хорошим. Конечно же, для более точного анализа необходимо рассматривать длинные отрезки альтернативных путей. Здесь возникает понятие свободного кодового расстояния d_{free}, о чем пойдет речь ниже.

Из диаграммы состояний рис. 4.4 становится очевидным, что сложность кодирования (а также и декодирования) возрастает с ростом числа состояний. Число состояний, в свою очередь, однозначно определяется количеством двоичных разрядов памяти кодера m.

Для сверточного (n, k, m)-кода *полная память кодера* определяется как

$$M = \sum_{i=1}^{k} m_i. \tag{4.34}$$

Заметим, что именно M последних информационных разрядов оказывают влияние на процесс кодирования. Этим разрядам соответствуют в точности 2^M различных состояний схемы кодера.

Пример: Сверточный (3,1,2)-код.

Заданы порождающие многочлены

$$g_1(X) = 1 + X, g_2(X) = 1 + X^2, g_3(X) = 1 + X + X^2 \tag{4.35}$$

и информационная последовательность

$$\{u[n]\} = \{1, 1, 0, 0, 1\}. \tag{4.36}$$

Найдите:

1. Схему кодера.
2. Состояния кодера.
3. Диаграмму состояний.
4. Таблицу переходов между состояниями.
5. Кодовую последовательность для $u[n]$, если кодирование начинается и заканчивается в нулевом состоянии.

Решение.

1.

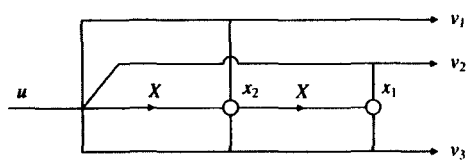


Рис. 4.5. Схема кодера сверточного (3,1,2)-кода.

2.

Таблица 4.3. Состояния кодера.

Состояние регистра		Состояние S_i
x_2	x_1	$i = x_2 + 2x_1$
0	0	0
1	0	1
0	1	2
1	1	3

3.

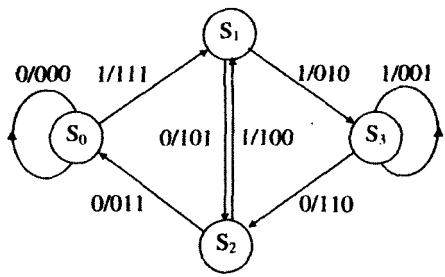


Рис. 4.6. Диаграмма состояний сверточного (3,1,2)-кода.

4.

Таблица 4.4. Таблица переходов между состояниями сверточного (3,1,2)-кода.

Состояние	Новое состояние $S[n+1]$		Кодовые биты	
	0	1	0	1
0	0	1	000	111
1	2	3	101	010
2	0	1	011	100
3	2	3	110	001

5. Так как кодирование должно заканчиваться в нулевом состоянии кодера, добавим к информационной последовательности два нуля и получим

$$\{v[n_2]\} = \{1,1,1,0,1,0,1,1,0,0,1,1,0,1,0,1,0,1,0,1,1\}. \qquad (4.37)$$

Графическое отображение состояний и переходов между ними дает возможность показать весь процесс кодирования непрерывно во времени в виде *сетевой диаграммы* (иногда ее называют решетчатой или треллисной диаграммой).

Пример: Сетевая диаграмма кодирования сверточного (3,1,2)-кода.

На сетевой диаграмме рис. 4.7 для каждого временного шага показаны все возможные состояния кодера. На переходах между ними

(ребрах), в качестве веса, представлены соответствующие информационные и кодовые биты. Так как в рассматриваемом примере $k = 1$, из каждого состояния выходят и в каждое состояние ведут в точности $2^1 = 2$ ребра.

Замечание. *Расположение состояний по вертикали специально подобрано таким образом, чтобы все ребра, содержащие нулевой информационный бит, вели вниз.*

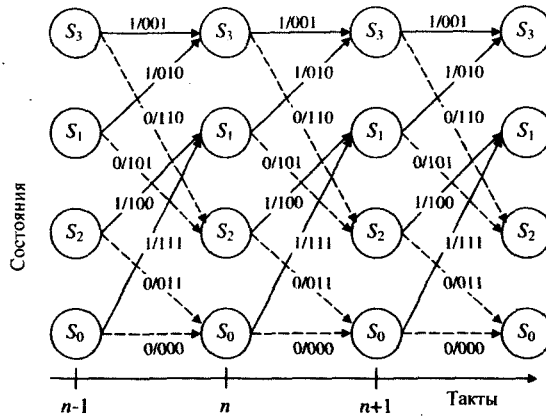


Рис. 4.7. Граф потока сигналов сверточного (3,1,2)-кода.

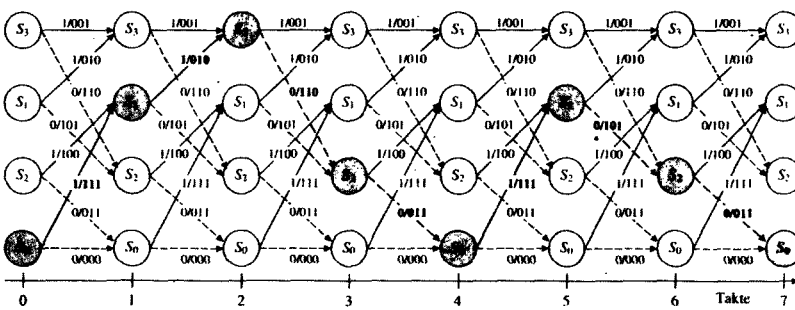


Рис. 4.8. Кодирование информационной последовательности.

На рис 4.8 показано кодирование информационной последова-

тельности (4.36). Это кодирование начинается из нулевого состояния S_0. Так как первый информационный бит равен $u_0[0] = 1$, мы попадаем в состояние S_1. Этому переходу соответствует первая тройка кодовых символов «111». Второй информационный бит $u_0[1] = 1$ приводит нас в состояние S_3 с выработкой следующей кодовой тройки «010» и т.д. В конце кодирования мы попадаем в нулевое состояние S_0.

4.5. Структура сверточных кодов

В предыдущем разделе было проведено описание сверточных кодов на основе диаграммы состояний и сетевой диаграммы. Ясно, что обе диаграммы содержат полную информацию об исследуемом коде. Описание сверточных кодов с помощью диаграмм состояний и переходов между ними открывает возможность более глубокого исследования их структуры и свойств.

До сих пор вопросы декодирования не обсуждались. Декодированию сверточных кодов будет посвящен следующий раздел этой книги. Однако, на основании сетевой диаграммы рис. 4.8, основные требования к хорошим кодам можно наметить уже сейчас.

Каждой кодовой последовательности соответствует свой путь на сетевой диаграмме, поэтому, ясно, что чем больше различий между путями, тем легче распознать истинное кодовое слово, тем меньше вероятность ошибки декодирования.

Как следует измерять различие между путями?

Когда мы обсуждали свойства двоичных линейных блоковых кодов (главы 2, 3), основным критерием различия было число несовпадающих символов, т.е. расстояние Хэмминга. Для блоковых кодов минимальное расстояние Хэмминга d_{min} играло особую роль, но также не менее важным было распределение весов слов линейного блокового кода. Эти же рассуждения можно применить и к сверточным кодам. В этом случае нулевым словом будем считать кодовый вектор, состоящий из одних нулей.

Если рассматривать сверточные коды с конечной длиной информационных векторов, то можно считать их линейными блоковыми кодами и к ним применимы все рассуждения, проведенные в главах 2 и 3.

Расстояния между кодовыми словами или расстояния между путями на сетевой диаграмме будем искать в метрике Хэмминга. В качестве базы используем нулевую последовательность, т.е. последовательность нулевых состояний. Без ограничения общности считаем,

что кодирование начинается и заканчивается в нулевом состоянии, поэтому, ненулевой кодовой последовательности на сетевой диаграмме соответствует путь, ответвляющийся от нулевого состояния, а затем сливающийся с ним.

Рассмотрим ненулевую последовательность, изображенную на рис. 4.8. Ей соответствует путь, состоящий из двух участков, ответвляющихся, а затем сливающихся с нулевым состоянием. В дальнейшем такие участки будем называть базовыми путями. Исследование структуры кода сводится к анализу базовых путей.

Методику такого анализа покажем на конкретном примере.

Пример: Модифицированная диаграмма состояний сверточного (3,1,2)-кода.

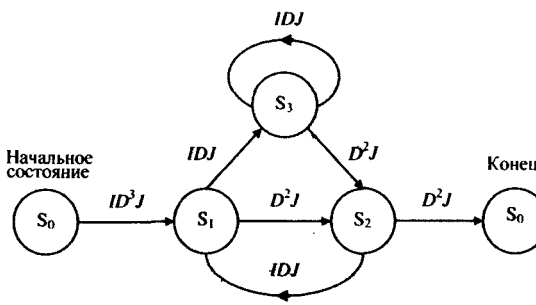


Рис. 4.9. Модифицированная диаграмма состояний сверточного (3,1,2)-кода.

Возьмем диаграмму состояний, изображенную на рис. 4.6. Будем исходить из того факта, что кодирование начинается и заканчивается в нулевом состоянии S_0, поэтому, модифицируем эту диаграмму таким образом, чтобы состояние S_0 было только или начальным или конечным (рис. 4.9). Для дальнейшего анализа промаркируем приросты весов Хэмминга информационной или кодовой последовательностей, а также прирост длины проходимого пути. С этой целью введем следующие переменные: для каждой информационной «1» – переменную I, для возрастания веса Хэмминга кодовой последовательности на l единиц – переменную D^l, и, наконец, для каждой смены состояний – переменную J. Промаркированные таким образом переходы нанесены на диаграмму состояний рис. 4.9. В этом случае каждому пути соответствует произведение маркеров всех переходов, из которых этот путь состоит. Например, информационной

последовательности

$$\{u[n]\} = \{1, 1, 0, 0\} \tag{4.38}$$

с последовательностью состояний

$$\{s[n]\} = \{S_0, S_1, S_2, S_0\} \tag{4.39}$$

в качестве весовой функции будет приписано выражение

$$ID^3J \cdot D^2J \cdot D^2J = ID^7J^3. \tag{4.40}$$

Таким образом, весовая функция любого базового пути, начинающегося и заканчивающегося нулевым состоянием содержит следующие переменные:

1. I с показателем, равным весу Хэмминга информационной последовательности;
2. D с показателем, равным весу Хэмминга кодовой последовательности;
3. J с показателем, равным числу переходов, осуществленных на этом пути.

Пример: Базовые пути и веса сверточного (3,1,2)-кода.

В таблице 4.5 представлены четыре кратчайших базовых пути и их весовые функции сверточного (3,1,2)-кода. Самый короткий из них имеет вес Хэмминга, равный 7, то есть соответствующая ему кодовая последовательность имеет единицы в семи разрядах. Из таблицы 4.5 может создаться впечатление, что с ростом длины путей их вес Хэмминга также возрастает. Проверим, так ли это для сверточного (3,1,2)-кода.

Для ответа на этот вопрос рассмотрим модифицированную диаграмму состояний рис. 4.9 и табл. 4.5. Каждое последующее удлинение пути требует прохождения некоторой «петли» на диаграмме состояний. Так как любая петля содержит минимум один переход веса d^1 или больше, то вес Хэмминга будут расти с увеличением длины путей, поэтому, не существует путей, отличных от нулевой последовательности, с весом Хэмминга, меньшим 7.

Анализ кода с помощью модифицированных диаграмм состояний базируется на математической теории графов, топологии и теории систем.

Таблица 4.5. Базовые пути и их весовые функции(N – вес Хэмминга информационной последовательности; C – вес Хэмминга кодовой последовательности; K – длина пути

Состояния $S[n]$	Весовые функции	N	C	K
S_0, S_1, S_2,S_0	$ID^3J \cdot D^2J \cdot D^2J = ID^7J^3$	1	7	3
S_0, S_1, S_3, S_2,S_0	$ID^3J \cdot IDJ \cdot D^2J \cdot D^2J = I^2D^8J^4$	2	8	4
S_0, S_1, S_3, S_3, S_2,S_0	$ID^3J \cdot IDJ \cdot IDJ \cdot D^2J \cdot D^2J = I^3D^9J^5$	3	9	5
S_0, S_1, S_2, S_1, S_2, S_0	$ID^3J \cdot D^2J \cdot IDJ \cdot D^2J \cdot D^2J = I^2D^{10}J^5$	2	10	5
и. т. д.				

Каждому состоянию рис. 4.9 поставим в соответствие переменные Z_e – начальному, Z_1,Z_2,Z_3 соответственно S_1,S_2,S_3 и Z_a – конечному состояниям.

Определим производящую функцию как

$$T(I, D, J) = \frac{Z_a}{Z_e}. \tag{4.41}$$

Для узлов графа можно записать следующие уравнения состояний:

$$Z_1 = ID^3J \cdot Z_e + IDJ \cdot Z_2 \tag{4.42}$$

$$Z_2 = D^2J \cdot Z_1 + D^2J \cdot Z_3 \tag{4.43}$$

$$Z_3 = IDJ \cdot Z_1 + IDJ \cdot Z_3 \tag{4.44}$$

$$Z_a = D^2J \cdot Z_2. \tag{4.45}$$

Для решения этой системы уравнений требуется несколько шагов. Из (4.44) следует

$$Z_3 = \frac{IDJ}{1 - IDJ} Z_1. \tag{4.46}$$

Используя (4.43), имеем

$$Z_2 = D^2J \left(1 + \frac{IDJ}{1 - IDJ}\right) Z_1 = \frac{D^2J}{1 - IDJ} Z_1 \tag{4.47}$$

или

$$Z_1 = \frac{1 - IDJ}{D^2J} Z_2. \tag{4.48}$$

Подставляя в (4.42), получаем

$$\frac{1-IDJ}{D^2J}Z_2 = ID^3J\cdot Z_e + IDJ\cdot Z_2, \tag{4.49}$$

что приводит к

$$Z_2 = \frac{ID^5J^2}{1-IDJ-ID^3J}Z_e. \tag{4.50}$$

Теперь уже все готово для того, чтобы непосредственно связать Z_a и Z_e

$$Z_a = D^2J\frac{ID^5J^2}{1-IDJ-ID^3J}Z_e = ID^7J^3\frac{1}{1-IDJ(1+D^2)}Z_e \tag{4.51}$$

и передаточная функция равна

$$T(I,D,J) = \frac{Z_a}{Z_e} = ID^7J^3\frac{1}{1-IDJ(1+D^2)}. \tag{4.52}$$

С помощью разложения

$$\frac{1}{1-X} = 1+X+X^2+X^3+\cdots \text{ для } |X|<1 \tag{4.53}$$

имеем

$$\begin{aligned} T(I,D,J) &= ID^7J^3(1+IDJ(1+D^2)+ \\ &\quad +I^2D^2J^2(1+D^2)^2+\cdots) = \\ &= ID^7J^3+I^2D^8J^4+I^2D^{10}J^4+I^3D^9J^5+\cdots. \end{aligned} \tag{4.54}$$

Разложение (4.54) определяет характеристики сверточного (3,1,2)-кода. Члены разложения содержат всю информацию о кодовых словах любой произвольной длины, первые члены соответствуют табл. 4.4.

Вычисленная в примере производящая функция задает весовую структуру кода, так как отражает влияние весов переходов на весь сверточный код

$$T(I,D,J) = \sum_{d=d_{free}}^{\infty}\sum_{i=1}^{\infty}\sum_{j=m+1}^{\infty} t(i,d,j)I^iD^dJ^j, \tag{4.55}$$

коэффициент $t(i,d,j)$ определяет число базовых путей с параметрами $I^iD^dJ^j$.

Таблица 4.6. Порождающие многочлены (в восьмеричной форме) оптимальных сверточных кодов со скоростями R и кодовым ограничением n_c.

	$R = 1/2$			$R = 1/3$				$R = 1/4$				
n_c	g_1	g_2	d_{free}	g_1	g_2	g_3	d_{free}	g_1	g_2	g_3	g_4	d_{free}
3	5	7	5	5	7	7	8	5	7	7	7	10
4	15	17	6	13	15	17	10	13	15	15	17	15
5	23	35	7	25	33	37	12	25	27	33	37	16
6	33	75	8	47	53	75	13	53	67	71	75	18
7	133	171	10	133	145	175	15	135	135	147	163	20
8	247	371	10	225	331	367	16	235	275	313	357	22
9	561	753	12	557	663	711	18	463	535	733	745	24

Минимальный выходной вес пути, начинающегося в нулевом состоянии, называется *свободным расстоянием* d_{free}. При анализе корректирующей способности кода d_{free} играет такую же роль, как минимальное кодовое расстояние для блоковых кодов.

Сверточный код считается оптимальным, если при заданной скорости кода и памяти он имеет максимальное d_{free}. Существует довольно много оптимальных кодов, но для их построения не существует никаких аналитических методов. Оптимальные коды находятся путем компьютерного моделирования. В литературе приведены многочисленные таблицы порождающих многочленов оптимальных сверточных кодов.

Пример: Порождающие многочлены оптимального сверточного (3,1,2)-кода.

Для построения оптимального сверточного (3,1,2)-кода воспользуемся табл. 4.6. Для заданных параметров кода коэффициенты порождающих многочленов равны $g_1 = 5_{(8)} = 101$, $g_2 = 7_{(8)} = 111$ и $g_3 = 7_{(8)} = 111$, поэтому, порождающие многочлены этого кода имеют следующий вид

$$g_1(X) = 1 + X^2,\ g_2(X) = 1 + X + X^2,\ g_3 = 1 + X + X^2. \qquad (4.56)$$

На рис. 4.10 показана зависимость свободного расстояния d_{free} от длины кодового ограничения для скоростей $R = 1/2$, $R = 1/3$

и $R = 1/4$. При увеличении длины кодового ограничения d_{free} возрастает, т.е. выигрыш от применения кодирования тем больше, чем больше сложность используемого кода. Свободное расстояние увеличивается также при снижении скорости кода. Здесь, однако, необходимо принимать во внимание реальный выигрыш от кодирования, измеренный в дБ отношения сигнал/шум. Этот выигрыш зависит от конкретных схем применения сверточных кодов и не всегда может быть достигнут снижением скорости.

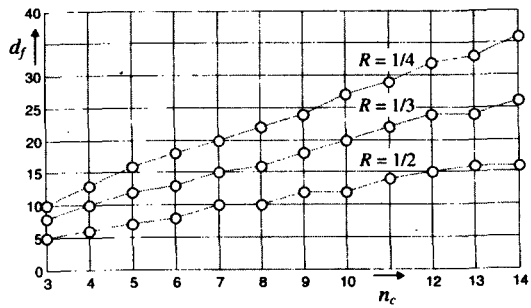


Рис. 4.10. Зависимость свободного расстояния d_{free} от длины кодового ограничения n_c.

Замечание. *В [7] для каждого кода дополнительно приводится выигрыш от кодирования в дБ. Там показано, что при малых скоростях кода выигрыш от кодирования для кодов с различным d_{free} приблизительно одинаков. При этом, однако, предполагается согласование алгоритма декодирования и канала, и, поэтому, этот результат не может быть распространен на общий случай.*

Особый класс сверточных кодов образуют так называемые *катастрофические сверточные коды*. Они не пригодны для использования, так как приводят к катастрофическому размножению ошибок. Рассмотрим пример катастрофического кода, а заодно научимся распознавать класс катастрофических кодов.

Пример: Катастрофический сверточный (2,1,2)-код.

Пусть сверточный код задан следующими многочленами

$$g_1(X) = 1 + X,\ g_2(X) = 1 + X^2. \tag{4.57}$$

1. Постройте схему кодера;

2. Постройте диаграмму состояний;

3. Закодируйте нулевую последовательность, т.е

$$\{u[n]\} = \{0, 0, 0, \ldots\};$$

4. Закодируйте последовательность $\{u[n]\} = \{1, 1, 1, \ldots\}$;

5. Предположим, что при передаче последовательности единиц произошли ошибки в первом, втором и четвертом битах кодовой последовательности. Как много ошибочных бит в информационной последовательности появится при декодировании?

6. Как по диаграмме состояний можно определить, что код является катастрофическим?

Решение.

1.

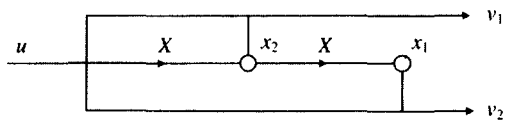


Рис. 4.11. Схема кодера сверточного (2,1,2)-кода.

2.

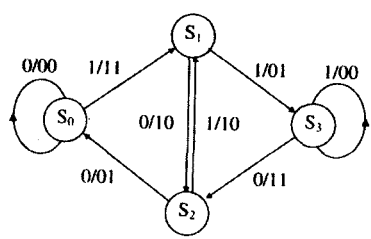


Рис. 4.12. Диаграмма состояний сверточного (2,1,2)-кода.

3. Так как кодирование начинается из нулевого состояния и мы все время в нем остаемся, кодовым словом будет нулевая последовательность.

4. Информационной последовательности единиц соответствует последовательность состояний

$$\{s[n]\} = \{S_0, S_1, S_3, S_3, \ldots\}, \tag{4.58}$$

что приводит к кодовой последовательности

$$\{v[n_2]\} = \{1,1,0,1,0,0,0,0,\ldots\}. \tag{4.59}$$

5. Если первый, второй и четвертый биты в (4.59) приняты ошибочно как «0», то декодер примет решение, что была передана нулевая информационная последовательность. Здесь мы имеем граничный случай – размножение ошибок.

6. Признаком катастрофичности кода является появление «петли» в модифицированной диаграммные состояний, в которой соответствующие кодовые символы являются нулевыми. На диаграмме состояний рис. 4.12 при переходе из состояния S_3 в состояние S_3 вес соответствующих кодовых символов равен нулю.

В заключении рассмотрим *систематические сверточные коды*. Эти коды используются достаточно редко, так как в них не достигается оптимальное свободное кодовое расстояние d_{free}. Тем не менее, в некоторых приложениях, например, при треллисной модуляции, применение систематических сверточных кодов весьма желательно [12].

Пример: Систематический сверточный (2,1,1)-код.

Систематический код задан порождающими многочленами

$$g_1 = 1 \text{ и } g_2(X) = 1 + X. \tag{4.60}$$

1. Постройте схему кодера;
2. Постройте диаграмму состояний;
3. Определите d_{free} с помощью весовой функции кода;
4. Является ли код катастрофическим?

Решение.

1.

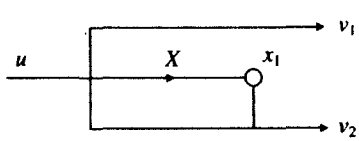


Рис. 4.13. Схема кодера систематического сверточного (2,1,1)-кода.

2.

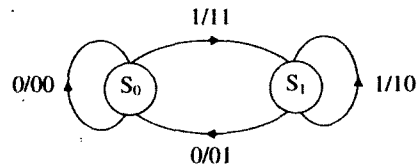


Рис. 4.14. Диаграмма состояний систематического сверточного (2,1,1)-кода.

3. Вычисление d_{free} с помощью весовой функции.

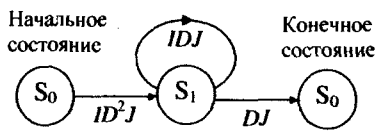


Рис. 4.15. Модифицированная диаграмма состояний сверточного (2,1,2)-кода.

Для состояний в узлах рис. 4.15 имеют место два равенства

$$Z_1 = ID^2JZ_e + IDJZ_1 \tag{4.61}$$

$$Z_a = DJ \cdot Z_1. \tag{4.62}$$

Связь между входом и выходом определяется как

$$Z_a = DJ\frac{ID^2J}{1 - IDJ}Z_e \tag{4.63}$$

и, следовательно, получаем весовую функцию

$$T(I,D,J) = ID^3J^2(1 + IDJ + I^2D^2J^2 + I^3D^3J^3 + \cdots) = \tag{4.64}$$
$$= ID^3J^2 + I^2D^4J^3 + I^3D^5J^4 + I^4D^6J^5 + \cdots.$$

Учитывая, что d_{free} определяется наименьшим показателем D в весовой функции, имеем

$$d_{free} = 3. \tag{4.65}$$

4. Код не является катастрофическим, так как единственная «петля» на модифицированной диаграмме состояний обладает весом IDJ и, следовательно, повышает вес кодовой последовательности на единицу.

4.6. Декодирования по максимуму правдоподобия

Для декодирования сверточных кодов имеются две альтернативы: последовательное декодирование и алгоритм Витерби.

Алгоритмы *последовательного декодирования* – стек-алгоритм или алгоритм Фано могут рассматриваться как методы проб и ошибок при поиске правильного пути на кодовом дереве. В ряде приложений использование последовательного декодирования может быть весьма эффективным. Для ознакомления с алгоритмами последовательного декодирования см. например, [5].

В настоящее время для декодирования сверточных кодов используют, как правило, *алгоритм Витерби*, который является частным случаем динамического программирования. Динамическое программирование применяется при решении задач математической оптимизации. Одной из таких задач является, например, прокладка шоссейных дорог на местности. В этой задаче требуется проложить дороги таким образом, чтобы общая длина была бы минимальной. Аналогичная задача возникает при декодировании сообщений. Здесь роль местности играют состояния на сетевой диаграмме, а роль общей длины пути берет на себя некоторая метрика. Идея алгоритма Витерби интуитивно возникает при внимательном рассмотрении сетевой диаграммы, поэтому, прежде всего рассмотрим пример, а затем займемся теоретическими обобщениями.

Пример: Декодирование сверточного (3,1,2)-кода с использованием сетевой диаграммы.

Исходным пунктом декодирования служит сетевая диаграмма рис. 4.8. Мы предполагаем, что кодирование начинается и заканчивается в состоянии S_0. Поиск альтернативных кодовых последовательностей сводится к поиску альтернативных путей на сетевой диаграмме (рис. 4.16.).

Предположим, что в канале не произошло ошибок, тогда на декодер поступает кодовая последовательность

$$\{r[n]\} = \{v[n]\} = \{1,1,1,0,1,0,1,1,0,0,1,1,1,1,1,1,0,1,0,1,1\}. \qquad (4.66)$$

Каким образом декодер может определить, какая последовательность была передана?

Декодер сравнивает биты принятой последовательности с битами возможных кодовых последовательностей и выбирает из всех кодовых последовательностей ту, которая наиболее «похожа» на принятую. Для независимых ошибок в канале мерой «похожести» является расстояние Хэмминга.

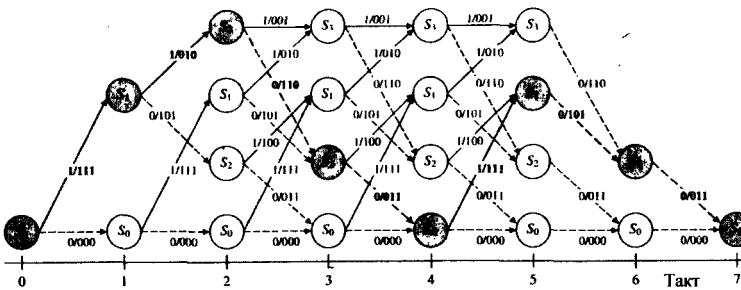


Рис. 4.16. Сетевая диаграмма сверточного $(3, 1, 2)$-кодера, соответствующая (4.36).

Сравнение расстояний Хэмминга всех кодовых последовательностей может быть эффективно реализовано с помощью сетевой диаграммы. На рис. 4.17 показан алгоритм декодирования и его результаты.

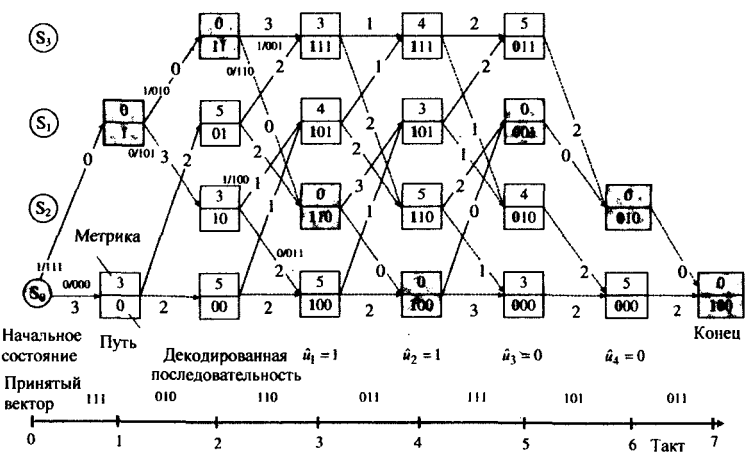


Рис. 4.17. Сетевая диаграмма декодера.

На первом шаге возможны два пути из нулевого состояния S_0. Декодер, прежде всего, сравнивает принятую тройку бит с тройками бит двух возможных путей. Найденные при этом расстояния Хэмминга являются метриками этих путей. Так как один путь ве-

дет в состояние S_0, а второй — в состояние S_1, в текущих регистрах метрик состояний S_0 и S_1 записываются метрики соответствующих им путей (см. рис. 4.17). Одновременно в текущие регистры путей записываются первые разряды соответствующих информационных последовательностей («0» – для S_0 и «1» — S_1).

На втором шаге декодирования из каждого состояния (S_0 и S_1) также возможны два перехода. Декодер прибавляет новые расстояния Хэмминга к текущим метрикам прежних состояний S_0 и S_1. Полученные таким образом новые метрики заносятся в регистры метрик новых состояний S_0, S_1 и S_2, S_3. В регистры путей этих состояний заносятся соответствующие им теперь уже вторые информационные разряды. В конце второго шага декодирования все возможные состояния сетевой диаграммы оказываются достигнутыми.

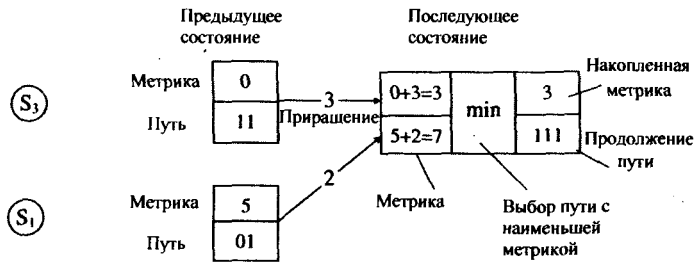


Рис. 4.18. Выбор пути с наилучшей метрикой.

На третьем шаге декодирования производится аналогичная процедура. Здесь, однако, впервые оказывается, что в каждое новое состояние ведут два пути. Вот тут то и раскрывается сущность динамического программирования. В качестве примера на рис. 4.18 рассмотрена процедура декодирования нового состояния S_3. В новое состояние S_3 могут переходить два прежних состояния S_1 и S_3 с приращениями метрик, равными 2 и 3 соответственно. С учетом прежних значений, новые метрики путей равны: от состояния S_1 – 7 и от состояния S_3 – 3, поэтому, на третьем шаге декодирования, в качестве предшествующего новому состоянию S_3, мы выбираем прежнее состояние S_3. Метрику нового состояния S_3 полагаем равной трем и, в качестве третьего информационного разряда, выбираем бит перехода $S_3 \rightarrow S_3$, равный «1». Дальнейшее приращение метрик путей, выходящих из состояния S_3, не зависят от метрики S_3, накопленной на третьем шаге, поэтому, на третьем шаге декодирования, мы мо-

жем смело отбросить переход $S_1 \to S_3$, как обладающий большей метрикой, чем переход $S_3 \to S_3$.

Таким образом, сущность *динамического программирования* заключается в том, что на каждом шаге декодирования по сетевой диаграмме для каждого состояния мы выбираем единственный, втекающий в него путь с минимальной метрикой (в случае равных метрик выбор одного из двух путей осуществляется произвольно).

Дальнейшие шаги декодирования производятся аналогично. При практической реализации процесс декодирования в данном примере можно существенно упростить. Из рис. 4.7 видно, что уже на третьем шаге всем состояниям предшествует первый бит информационной последовательности, равный «1», поэтому, первый информационный символ уже можно выдать потребителю и в дальнейшем не учитывать. Таким образом снижается длина регистров пути декодера и уменьшается время задержки декодирования. Окончательно, на 7-ом шаге декодирования мы выбираем в состоянии S_0 путь, обладающий наименьшей метрикой. Содержимое регистра пути состояния S_0 дописывается к ранее продекодированным информационным символам и, тем самым, процесс декодирования заканчивается.

Рассмотренный пример раскрывает основы алгоритма Витерби. Процесс декодирования по максимуму правдоподобия обобщает рис. 4.19. Для наглядной интерпретации представим кодовую последовательность в виде вектора $\mathbf{v} = (v_0, v_1, v_2, \ldots)$.

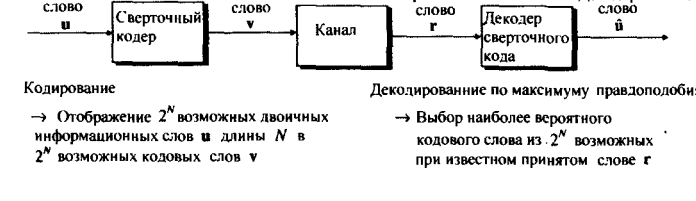


Рис. 4.19. Декодирование по максимуму правдоподобия.

Задачу *декодера максимального правдоподобия* (МП) можно сформулировать следующим образом: имея принятый вектор $\mathbf{r}$ из всех возможных слов $\mathbf{v}$, принадлежащих коду, выбрать такое $\hat{\mathbf{v}}$, для которого

$$P(\mathbf{r}/\hat{\mathbf{v}}) = \max_{\mathbf{v} \in \text{коду}} P(\mathbf{r}/\mathbf{v}). \tag{4.67}$$

Решение задачи декодирования по МП зависит от выбранной модели канала. Для источников и каналов без памяти (например, для

канала с АБГШ) задача упрощается. В этом случае передача отдельных бит кодовой последовательности длины J происходит независимо. Вместо того, чтобы для вычисления $P(\mathbf{r}/\mathbf{v})$ рассматривать всю последовательность, мы можем свести вычисление $P(\mathbf{r}/\mathbf{v})$ к произведению условных вероятностей отдельных бит

$$P(\mathbf{r}/\mathbf{v}) = \prod_{j=0}^{J-1} P(r_j/v_j). \tag{4.68}$$

С точки зрения технических затрат, произведение условных вероятностей удобнее свести к сумме их логарифмов. Логарифмическая функция является монотонной и не изменяет соотношение между величинами условных вероятностей кодовых последовательностей. В этом случае, для *логарифмической функции правдоподобия*, (4.67) преобразуется в равенство

$$\log P(\mathbf{r}/\hat{\mathbf{v}}) = \max_{\mathbf{v}\in\text{коду}} \prod_{j=0}^{J-1} \log P(r_j/v_j). \tag{4.69}$$

Для каналов без памяти декодирование по максимуму правдоподобия может быть реализовано с помощью алгоритма Витерби с наименьшими затратами. Введем следующие вспомогательные величины:

1. *Метрику* кодовой последовательности v_i

$$M(\mathbf{r}/\mathbf{v}_i) = \log M(\mathbf{r}/\mathbf{v}_i); \tag{4.70}$$

2. *Приращение метрики*, как вклад j-ой компоненты

$$M(r_j/v_{i,j}) = \log P(r_j/v_{i,j}); \tag{4.71}$$

3. *Частичную метрику*, как промежуточную сумму

$$M_k(\mathbf{r}/\mathbf{v}_i) = \sum_{j=1}^{k-1} M(r_j/v_{i,j}). \tag{4.72}$$

Работа декодера Витерби показана на рис. 4.17 и 4.18. Для каждого состояния вычисляются метрики всех вливающихся в него путей. Величина приращений метрик зависит от модели канала. Ниже будут представлены два примера вычисления метрик. Для каждого конкретного состояния величина приращений метрик выходящих из него путей не зависит от метрик путей, вливающихся в него. На

каждом такте для каждого состояния из всех путей, в него вливающихся, декодер выбирает для продолжения единственный путь, обладающий наибольшей метрикой.

После того, как алгоритм Витерби описан в общих чертах, можно оценить его сложность.

1. Для декодера с памятью M существует 2^M возможных состояний.

2. На каждом шаге декодирования определяются 2^{M+1} приращений метрик. Частичные метрики подсчитываются и сравниваются.

3. На каждом шаге декодирования в память заносятся 2^M указателей путей с частичными метриками этих путей.

Можно заметить, что сложность декодера Витерби экспоненциально возрастает с ростом памяти декодера.

Пример: Метрика декодера при передаче информации по двоичному симметричному каналу (ДСК).

Двоичный канал без памяти (ДСК), по определению, является каналом, в котором передаваемые биты искажаются независимо друг от друга с вероятностью ε (см. рис. 4.20).

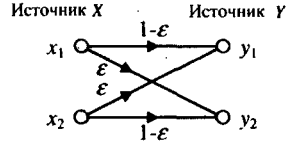


Рис. 4.20. Диаграмма передачи информации по двоичному симметричному каналу.

При декодировании по максиму правдоподобия сравниваются условные вероятности кодовых слов (4.67). Условная вероятность события, при котором при передаче слова $\mathbf{v}_i$ принимается слово $\mathbf{r}$, для ДСК определяется только расстоянием Хэмминга $d_H(\mathbf{r}, \mathbf{v}_i)$. Если длина кодовой последовательности равна J, то эта условная вероятность равна произведению вероятностей искажения $d_H(\mathbf{r}, \mathbf{v}_i)$ двоичных символов и правильного приема $J - d_H(\mathbf{r}, \mathbf{v}_i)$ бит. Переходя к

логарифмической функции правдоподобия, имеем

$$\log P(\mathbf{r}/v_i) = \log(\varepsilon^{d_H(\mathbf{r},\mathbf{v}_i)}(1-\varepsilon)^{J-d_H(\mathbf{r},\mathbf{v}_i)}) = \\ = d_H(\mathbf{r},\mathbf{v}_i)\log\frac{\varepsilon}{1-\varepsilon} + J\log(1-\varepsilon). \tag{4.73}$$

Результат может быть существенно упрощен. Параметры J и ε не зависят от передаваемого сообщения и, поэтому, не оказывают никакого влияния на решение декодера. Это значит, что второе слагаемое в (4.73) может быть просто опущено. В оставшемся произведении сомножитель $\log\frac{\varepsilon}{1-\varepsilon}$ является константой и имеет отрицательное значение при $\varepsilon < 0,5$. Если его отбросить, то декодер должен искать кодовое слово $\hat{\mathbf{v}}$ не с максимальной условной вероятностью $p(\mathbf{r}/\hat{\mathbf{v}})$, а с минимальным расстоянием Хэмминга $d_H(\mathbf{r},\hat{\mathbf{v}})$.

Таким образом, правило решения декодера максимального правдоподобия для ДСК можно сформулировать следующим образом: декодер ищет такое кодовое слово $\hat{\mathbf{v}}$, для которого

$$d_H(\mathbf{r},\hat{\mathbf{v}}) \le d_H(\mathbf{r},\mathbf{v}) \quad \forall \quad \mathbf{v} \in \text{ коду}. \tag{4.74}$$

Если имеется несколько таких кодовых слов, то из них произвольно выбирается любое.

Замечание. *Рассматривая в предыдущем примере работу декодера Витерби, мы интуитивно правильно использовали метрику Хэмминга. Теперь мы убедились в том, что декодирование по критерию максимального правдоподобия в ДСК сводится к поиску кодового слова с минимальным расстоянием Хэмминга до принятой последовательности.*

Пример: Декодирование Витерби сверточного (3,1,2)-кода при передаче информации по ДСК.

Рассмотрим декодирование по максимуму правдоподобия с помощью алгоритма Витерби для ДСК. Данный пример аналогичен предыдущему за исключением того, что в принятое сообщение внесены ошибки.

Процесс декодирования зашумленного кодового слова показан на рис. 4.21. В принятую последовательность (нижняя часть рис. 4.21) внесены пять ошибок (на рис. они выделены жирным шрифтом). Декодирование происходит аналогично предыдущему примеру. Разница заключается в том, что на третьем шаге декодирования в состоянии S_1 для продолжения выбирается путь, ведущий на втором

шаге из состояния S_0 в S_1 и, поэтому, на третьем шаге декодирования не происходит совпадения первого принятого бита для всех состояний (как это было в предыдущем примере). Таким образом, первый информационный бит не может быть выдан потребителю на третьем шаге. Как видно, в нашем случае наличие шума приводит к задержке процесса декодирования. Из рис. 4.21 следует, что первый информационный бит выдается потребителю только на пятом шаге. Далее, на шестом шаге выдаются сразу три бита, совпадающих для всех состояний, и, наконец, на седьмом, последнем шаге декодирования к ним добавляются недостающие биты. Заметим, что несмотря на наличие 5 ошибочных бит в кодовом слове, все ошибки исправляются (слово продекодировано правильно).

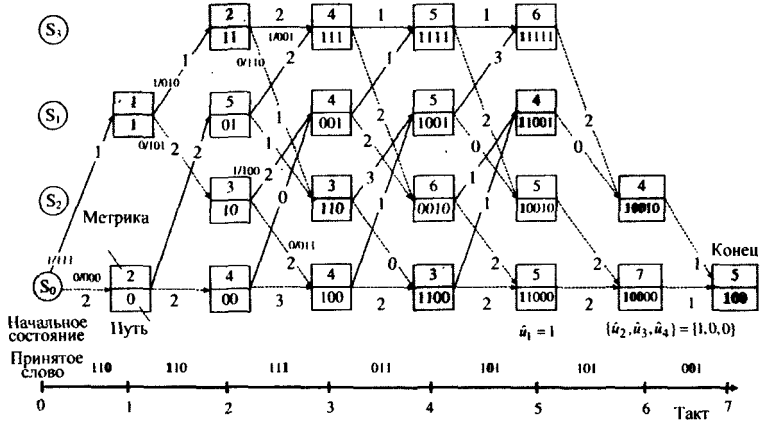


Рис. 4.21. Декодирование по максимуму правдоподобия с помощью алгоритма Витерби.

Заметим также, что при правильном декодировании, метрика найденного пути показывает число ошибок в принятом слове. Эта информация может быть использована для оценки надежности продекодированного сообщения. Наконец, число исправленных ошибок может служить индикатором качества канала. Так, например, метрика продекодированного слова позволяет своевременно обнаружить резкое ухудшение качества канала и принять соответствующие контрмеры.

Пример: Метрика декодера при передаче информации по каналу с аддитивным белым гауссовским шумом (АБГШ).

При передаче информации противоположными сигналами по каналу с АБГШ и приеме с помощью согласованных фильтров используется модель канала связи, представленная на рис. 4.22 (см. Часть I, раздел 7.5.2). Согласно рис. 4.22, отдельный бит кодового слова «1» или «0» после передачи по каналу без шума и детектирования принимает аналоговое значение r_0 или $-r_0$. Из-за наличия шума в канале продетектированный сигнал $\mathbf{r}$ на выходе приемника является непрерывной стохастической переменной с условными плотностями распределения вероятностей, равными $f(r/1)$ $f(r/0)$

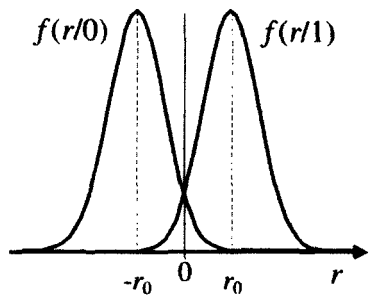


Рис. 4.22. Условные плотности распределения вероятностей продетектированного сигнала $\mathbf{r}$ при передаче противоположными сигналами по каналу с АБГШ.

$$f(r/0) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r+r_0)^2}{2\sigma^2}\right) \qquad (4.75)$$

$$f(r/1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r-r_0)^2}{2\sigma^2}\right).$$

Замечание. *Значения r_0 и σ^2 зависят от канала передачи информации. Приведенные в примере рассуждения можно обобщить и на случай M-ичных сигналов и использовать там аналогичную метрику.*

Будем предполагать, что «1» и «0» передаются независимо друг от друга с вероятностями, равными 1/2. Канал с АБГШ также является каналом без памяти, поэтому, для декодера максимального правдоподобия приращение метрики можно определить как

$$M'(r_j/v_j) = \log P(r_j/v_j) = \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r \pm r_0)^2}{2\sigma^2}\right)\right], \qquad (4.76)$$

где $+r_0$ при $v_j = 0$ и $-r_0$ при $v_j = 0$. Выражение (4.76) можно упростить

$$\log P(r_j/v_j) = -\frac{(r \pm r_0)^2}{2\sigma^2} \log e - \frac{1}{2} \log 2\pi\sigma^2. \qquad (4.77)$$

Так как для декодера МП все константы несущественны, их можно отбросить, поэтому, без потери общности, в качестве приращения метрики можно использовать расстояние Евклида между продетектированным сигналом $\mathbf{r}$ и значениями $\pm r_0$

$$M''(r_j/v_j) = -(r \pm r_0)^2. \qquad (4.78)$$

Можно произвести дальнейшие упрощения. Из

$$(r \pm r_0) = r^2 \pm 2rr_0 + r_0^2 \qquad (4.79)$$

следует, что для декодера МП приращение метрики можно свести к

$$M(r_j/v_j) = \begin{cases} -r & \text{для } v_j = 0 \\ +r & \text{для } v_j = 1 \end{cases}. \qquad (4.80)$$

Таким образом, при вычислении приращений метрик нужно только учитывать знак переменной $\mathbf{r}$.

Заметим, что метрика декодера является непрерывной величиной. В этом случае говорят о мягком решении декодера, в отличии от жесткого решения, при котором продетектированный сигнал квантуется на два уровня «0» и «1». Ниже приводится пример практической реализации декодера Витерби с мягким решением, при этом, в частности, учитывается сложность реализации. Во многих практических применениях, например, в мобильной связи, предъявляются повышенные требования к стоимости декодера. Очень часто декодер представляет собой интегральную схему, размещенную в одном чипе. Важную роль играют так же надежность в эксплуатации и минимальная мощность принимаемого сигнала. В связи с указанными требованиями, разработчики декодеров стремяться минимизировать их сложность с минимальными потерями эффективности, при этом, особую роль играют целочисленное представление метрики и простота арифметических операций декодирования.

Упрощенные декодеры позволяют реализовать субоптимальные решения. Во многих применениях достигнут разумных компромисс между сложностью декодера Витерби с мягким решением и минимальной мощностью принимаемого сигнала.

Пример: Декодер Витерби с мягким решением для канала с АБГШ.

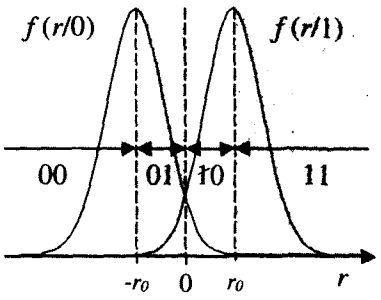


Рис. 4.23. Квантование принятого сигнала.

Пусть информация источника с равномерным распределением символов «0» и «1» передается по каналу с АБГШ. В этом случае на выходе согласованных фильтров приемника мы получаем продетектированный сигнал $\mathbf{r}$ с нормальным распределением плотностей условных вероятностей $f(r/0)$ и $f(r/1)$ (рис. 4.23). Вместо жесткого решения, при котором условные вероятности квантуются на два уровня, рассмотрим реализацию декодером простейшего мягкого решения. При таком решении на вход декодера поступают символы кодового слова «0» и «1», имеющие длину, равную двум битам. Такая информация соответствует 4 уровням квантования текущих значений переменной $\mathbf{r}$ (см. рис. 4.23). Дополнительный двоичный разряд входных символов позволяет различать «хорошие» и «плохие» биты. Смысл такой градации состоит в том, что если переменная $\mathbf{r}$ имеет положительное значение, очень близкое к нулю, то вероятность «1» лишь немного превышает вероятность «0». В этом случае имеет смысл говорить о «ненадежной» или «плохой» единице. Если значение $\mathbf{r}$ превышает r_0, то с большой долей уверенности можно утверждать, что была принята единица.

Замечание. *Интервалы квантования переменной* $\mathbf{r}$ *были выбраны интуитивно, для того, чтобы продемонстрировать реализацию декодера Витерби с мягким решением.*

При мягком решении с минимальными техническими затратами для подсчета приращений метрик будем пользоваться таблицей 4.7. В этом случае текущее значение метрики рассматриваемого пути

Таблица 4.7. Таблица приращений метрик.

Принятый символ	Приращение метрики		Коментарий
	0	1	
$00 \to 0g$	3	0	«хороший» ноль
$01 \to 0b$	2	1	«плохой» ноль
$10 \to 1b$	1	2	«плохая» единица
$11 \to 1g$	0	3	«хорошая» единица

будет складываться с некоторым целым числом из интервала]0,3[. Выбор этого числа зависит от уровня квантования, в который попадает продетектированный сигнал **r** и текущего значения кодового символа анализируемого участка пути.

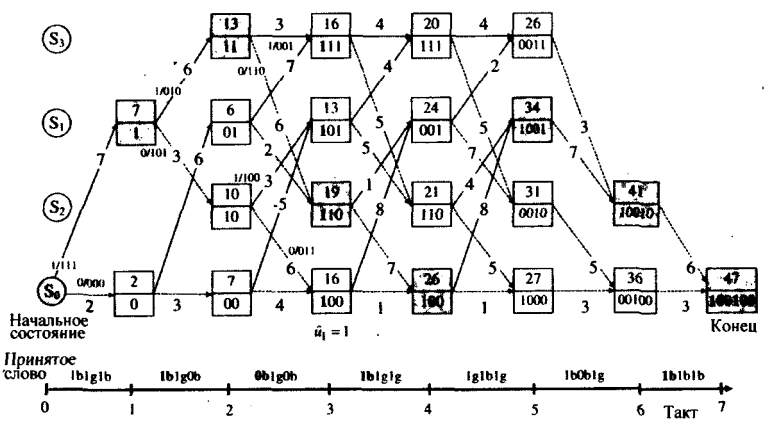


Рис. 4.24. Декодирование с помощью алгоритма Витерби с мягким решением.

Рассмотрим числовой пример работы декодера с мягким решением. Исходные данные для этого примера получены моделированием на компьютере канала с $r_0 = 1$ и $\sigma^2 = 1/2$ и сведены в таблицу 4.8. В первой строке этой таблицы приведена передаваемая кодовая последовательность, идентичная рассмотренной в предыдущем примере. Вторая строка транспонирует эту последовательность в биполярные сигналы «1» и «-1». Третья строка содержит значения продетектиро-

Таблица 4.8. Передача информации по каналу с АБГШ и прием с квантованием принятого сигнала на 4 уровня.

n	0	1	2	3	4	5	6	7	8	9	10	11
Кодовые биты	1	1	1	0	1	0	1	1	0	0	1	1
Биполярные символы	1	1	1	-1	1	-1	1	1	-1	-1	1	1
Принято	0,3	2,0	0,4	0,4	1,2	-0,7	-0,5	1,0	-0,7	0,4	1,4	2,2
Мягкий вход	1b	1g	1b	**1b**	1g	0b	**0b**	1g	0b	**1b**	1g	1g
n	12	13	14	15	16	17	18	19	20			
Кодовые биты	1	1	1	0	1	0	1	1	0			
Биполярные символы	1	1	1	1	-1	1	-1	1	1			
Принято	1,4	0,5	1,3	0,3	-0,0	1,0	0,0	0,8	1,8			
Мягкий вход	1g	1b	1g	1b	0b	1g	**1g**	1b	1b			

ванного сигнала **r**, полученные моделированием, и, наконец, в четвертой строке содержатся входные символы декодера из табл. 4.7, соответствующие переменной **r** из предыдущей строки.

Процесс декодирования представлен на рис. 4.24. Входные символы последовательно сопоставляются с двоичными кодовыми символами исследуемых путей и, в соответствии с табл. 4.7, вычисляются метрики этих путей.

На первом шаге декодирования на вход декодера поступает последовательность символов «1b, 1g, 1b». В этом случае метрика пути, содержащего кодовые символы «0, 0, 0», равна $1+0+1=2$, а метрика пути «1, 1, 1» равна $2+3+2=7$. Чем больше метрика, тем выше степень соответствия принятой последовательности кодовым символам исследуемого пути и для дальнейшего продолжения выбирается путь с наибольшей метрикой.

Рассмотренный пример показывает, что декодер Витерби с мягким решением может быть реализован с помощью целочисленной арифметики. Выигрыш от применения мягкого решения может быть довольно существенным. Практика показывает, что в реальных системах связи переход от жесткого решения к мягкому с квантованием продетектированного сигнала на $2^3 = 8$ уровней позволяет получить дополнительный выигрыш в отношении сигнал/шум, равный 2,5 дБ.

Следующим важным параметром декодера является длина за-

поминаемого пути. Как правило, эта длина задается равной 3 – 5 длинам кодового ограничения выбранного сверточного кода. Если происходит переполнение памяти, равной произведению числа состояний декодера на длину запоминаемого пути, то принимается принудительное решение о старших декодированных символах. Такое решение может быть принято, например, выбором пути с наилучшей, на момент принятия решения, метрикой. В нашем примере при длине кодового ограничения $n_c = 3$, длина запоминаемого пути должна составлять 9 - 15 бит. При небольшой глубине декодирования снижение этой длины приводит к резкому снижению корректирующей способности кода.

Мягкое решение используется также при декодировании *турбокодов*, обладающих уникальными свойствами. Модифицированный декодер Витерби осуществляет декодирование каждого принятого бита по максимуму апостериорной вероятности (MAP). Для этого для каждого бита вычисляются апостериорные вероятности его идентичности «0» и «1». Таким образом, модифицированный декодер Витерби имеет мягкое решение на выходе. При многоступенчатом декодировании это мягкое решение используется при декодировании сверточных кодов следующих ступеней (обычно в турбо-кодах используется 2 - 3 ступени кодирования). Путем нескольких итераций этой процедуры, турбо-коды позволяют достичь скорости передачи информации, близкой к пропускной способности канала [2], [13].

В заключение упомянем еще о треллисных кодах, использующих концепцию сверточных кодов в пространстве сигналов с цифровой модуляцией. Вместо расстояния Хэмминга, эти коды характеризуются расположением последовательностей символов в пространстве сигналов [12], [13].

4.7. Детектор Витерби

Прицины сверточного кодирования и декодирования по алгоритму Витерби могут с успехом применяться при детектировании узкополосных сигналов (*мобильные сети связи*) и при считывании информации с магнитных носителей. Дело заключается в том, что структура таких каналов соответствует структуре сверточного кодирования с одним входом и одним выходом.

На рис. 4.25 представлена модель канала, в которой имеет место *многолучевое распространение* радиоволн. В этой модели каждый символ дискретной последовательности $d[n]$ проходит через линию

задержки (содержащую ряд элементов задержки D) и многократно складывается с некоторым весом с предыдущими и последующими символами. Таким образом выход незашумленного канала в каждый дискретный момент времени зависит от символа, поступающего на вход приемника, состояния $x_i[n]$ и коэффициента $f_i[n]$. На рис. 4.25 к этому сигналу добавлен аддитивный белый гауссовский шум.

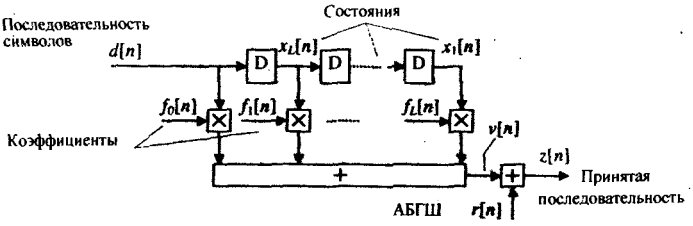


Рис. 4.25. Передача последовательности $d[n]$ по дискретному во времени каналу с элементами задержки D, выходом $v[n]$, гауссовским шумом $r[n]$ и принятой последовательностью $z[n]$.

Из-за наличия цепи, содержащей элементы задержки D, в таком канале происходит «наслоение» соседних символов, что приводит к межсимвольной интерференции (МСИ).

Качество принимаемого сигнала может быть существенно улучшено, если при детектировании принимать во внимание наличие МСИ. Для этого детектор должен знать коэффициенты $f_i[n]$ в каждый момент времени n. В сети мобильной связи GSM по каналам периодически передается фиксированная последовательность символов и производятся необходимые измерения.

Если внимательно проанализировать работу детектора компенсирующего МСИ (его обычно называют экволайзером), то можно обнаружить, что алгоритм его работы почти в точности совпадает с алгоритмом Витерби. Продемонстрируем работу детектора Витерби на простом примере.

Пример: Детектор Витерби.

Примем в качестве исходной модель канала, представленную на рис. 4.25. Ограничимся рассмотрением канала с одним элементом задержки D. В этом случае возможны два состояния канала S_0 и S_1.

Таблица 4.9. Состояние канала с МСИ и уровни сигналов.

Символ на входе	Значение	Состояние	Уровень
$d[n]$	$x_1[n]$	$S[n]$	$v[n]$
-1	-1	S_0	$-0,7-0,3=-1,0=d_0$
+1			$+0,7-0,3=+0,4=d_1$
-1	1	S_1	$-0,7+0,3=-0,4=d_2$
+1			$+0,7+0,3=+1,0=d_3$

Пусть последовательность сигналов $d[n]$ состоит из биполярных символов

$$d[n] \in \{-1, 1\}. \tag{4.81}$$

Выберем коэффициенты f_0 и f_1 равными

$$f_0 = 0,7 \text{ и } f_1 = 0,3. \tag{4.82}$$

В таблице 4.9 для входных символов $d[n]$ в зависимости от значения $x_1[n]$, определяющей состояние канала S_0 и S_1, приведены уровни незашумленных сигналов $v[n]$. Согласно модели канала рис. 4.25 к этим сигналам добавляется гауссовский шум $r[n]$. На входе детектора мы имеем сигнал $z[n]$ с нормальным распределением и средними значениями из $\{d_0, d_1, d_2, d_3\}$ (табл. 4.9).

На каждом такте декодер сравнивает величину $z[n]$ с уровнями сигнала из $\{d_0, d_1, d_2, d_3\}$ и вычисляет метрики соответствующих переходов между состояниями канала. Исходя из принципа максимального правдоподобия и нормального распределения $z[n]$, в качестве метрик выбираются величины

$$\lambda_i = (z - d_i)^2. \tag{4.83}$$

На сетевой диаграмме (рис. 4.26) показаны состояния канала, переходы между ними и приращения текущих метрик, соответствующие этим переходам.

После предварительных замечаний рассмотрим численный пример работы детектора Витерби. В таблице 4.10 приведены последовательность входных сигналов, уровни принимаемых незашумленных символов и последовательность сигналов на входе детектора. Работа детектора Витерби представлена на рис. 4.27.

Детектор Витерби на сетевой диаграмме рис. 4. 27 начинает декодирование на первом такте из состояния S_1 (на нулевом такте канал

принудительно переводится в это состояние путем передачи «+1»). На первом такте возможны только два перехода. Найдем приращения метрик на этих переходах и нанесем их на сетевую диаграмму

$$\begin{aligned} \lambda_3 &= (z-d_3)^2 = (2{,}2-1)^2 = 1{,}44 \\ \lambda_2 &= (z-d_2)^2 = (2{,}2+0{,}4)^2 = 6{,}76. \end{aligned} \tag{4.84}$$

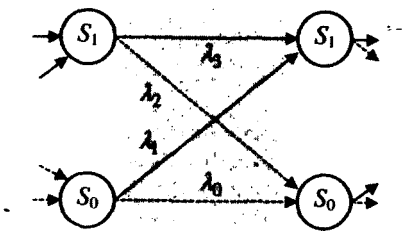


Рис. 4.26. Переходы между состояниями и приращения метрик.

Таблица 4.10. Передача информации по каналу с МСИ.

Такт n	0	1	2	3	4	5	6	7	8	9	10	11	12
Символ $d[n]$	1[1]	1	-1	1	-1	1	1	1	-1	-1	-1	1	[2]
Канал $v[n]$	0,7	1,0	-0,4	0,4	-0,4	0,4	1,0	1,0	-0,4	-1,0	-1,0	0,4	0,3
Принято $z[n]$	1,1	2,2	0,02	-0,06	-0,13	-0,31	0,99	0,97	-0,40	-1,2	-0,23	-0,93	0,6

[1] Инициализация (посылаемый бит известен приемнику).
[2] Замирание.

Так как на первом такте не происходит слияния альтернативных путей, приращения метрик непосредственно заносятся в регистры метрик оптимальных путей для состояний S_0 и S_1. В регистры путей для состояний S_0 и S_1 заносятся соответственно «0» и «1».

На втором такте декодирования возможны уже все переходы. Найдем приращения метрик для всех четырех переходов и для продолжения выберем путь с наименьшей метрикой. Так как для первого входного бита произошло слияние путей для состояний S_0 и S_1, будем считать первый принятый бит продетектированным.

Продолжая этот процесс далее, продетектируем всю последовательность входных символов.

Несмотря на значительное воздействие на детектируемый сигнал шумовой компоненты, все символы продетектированы правильно.

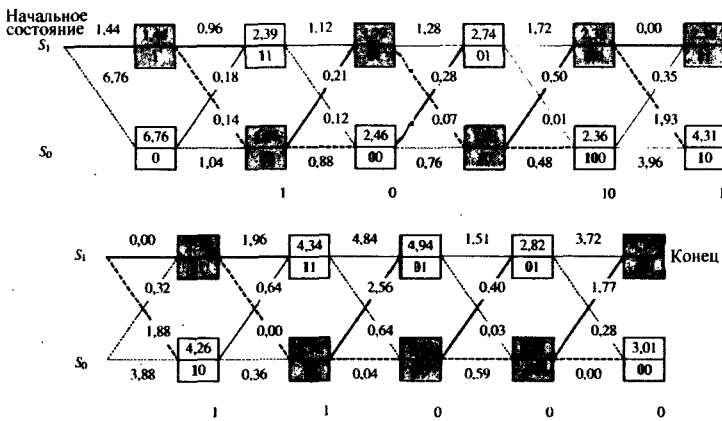


Рис. 4.27. Сетевая диаграмма детектора Витерби.

4.8. Упражнения

Упражнение 4.1: Сверточный код.

1. Найдите порождающий многочлен двоичного сверточного кода, обладающего максимальным свободным расстоянием, со скоростью $R = 1/2$ и длиной кодового ограничения $n_c = 6$;

2. Приведите схему кодера;

3. Нарисуйте диаграмму состояний;

4. Проведите кодирование информационной последовательности $\{u[n]\} = \{1, 0, 0, 1, 1, 0, 1\}$ с помощью диаграммы состояний. Для этого, прежде всего, найдите последовательность состояний, считая, что процедура кодирования начинается и заканчивается в нулевом состоянии;

5. Предположим, что принята последовательность $\{r[n_2]\} = \{11, 01, 11, 01, 10, \ 10, 10, 01, 11\}$. Проведите декодирование данной последовательности по максимуму правдоподобия с помощью

сетевой диаграммы состояний; Указание: при декодировании учтите, что кодирование начиналось и заканчивалось нулевым состоянием.

6. Как может быть интерпретирована метрика декодированной информационной последовательности?

Упражнение 4.2: Сверточный код, используемый в сети мобильной связи GSM.

Для передачи речи в сетях мобильной связи GSM используется оптимальный сверточный код с производящими многочленами $g_1(x) = 1 + x^3 + x^4$ и $g_2(x) = 1 + x + x^3 + x^4$, имеющий свободное расстояние, равное $d_{free} = 7$

1. Приведите схему кодера, содержащую переменные x_1, x_2, x_3, x_4.

2. Каждый речевой блок, продолжительностью 20 мсек, содержащий 185 бит, кодируется сверточным кодом. Сколько нулей нужно добавить к информационной последовательности для того, чтобы кодирование закончилось в нулевом состоянии?

3. Определите значения следующих величин: памяти кодера, кодового ограничения, блоковой скорости, относительной потери скорости, полной памяти кодера.

4. Оцените сложность реализации декодера Витерби.

Упражнение 4.3: Сверточный код, используемый в сети мобильной связи GSM.

Для передачи данных в сети GSM со скоростью 4,8 кбит/сек по каналу TCH/F4,8 используется сверточный код с порождающими многочленами $g_1(x) = 1 + x + x^3 + x^4$, $g_2(x) = 1 + x^2 + x^4$ и $g_3(x) = 1 + x + x^2 + x^3 + x^4$ с $d_{free} = 12$.

1. Приведите схему кодера, содержащую переменные x_1, x_2, x_3, x_4.

2. Из технических соображений каждый информационный блок, содержащий 60 бит, кодируется словом, длина которого составляет 228 бит. Сколько нулевых бит добавляется в конце информационного блока? Каким состоянием заканчивается декодирование?

3. Определите значение памяти кодера, кодового ограничения, блоковой скорости кода, относительной потери кодовой скорости, полной памяти кодера.

4. Оцените сложность реализации декодера Витерби.

Упражнение 4.4: Детектор Витерби с субоптимальной метрикой.

Повторите ранее рассмотренный пример работы детектора Витерби при субоптимальной метрике

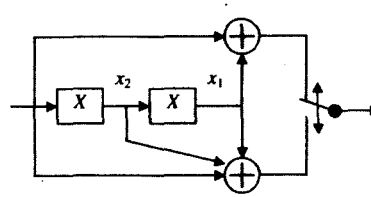
$$\lambda_i = \mid z - d_i \mid .$$

Решения заданий.

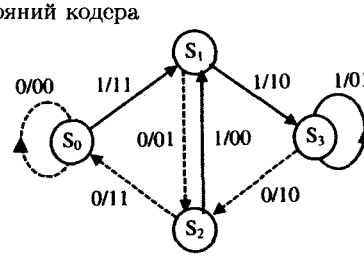
Решение задания 4.1. Сверточный код.

1. Для заданных параметров из таблицы находим порождающие многочлены $g_1(x) = 1 + x^2$; $g_2(x) = 1 + x + x^2$.

2. Схема кодера.



3. Диаграмма состояний кодера



4. Кодирование информационной последовательности.

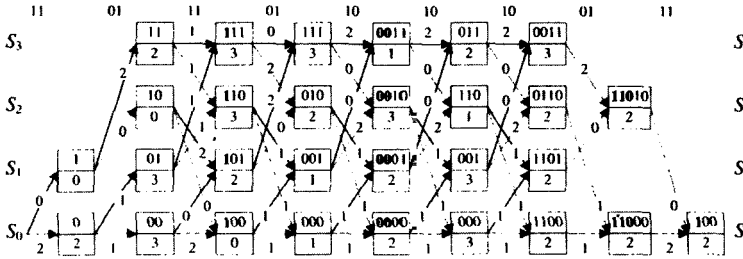
Из диаграммы состояний получаем:

Последовательность состояний (старт S_0) $- S_1 - S_2 - S_0 - S_1 - S_3 - S_2 - S_1 (-S_2 - S_0$ «хвост»).

Кодовое слово $\{v[n]\} = \{11, 01, 11, 11, 10, 10, 00, 01, 11\}$

5. Декодирование по сетевой диаграмме с минимальным расстоянием Хэмминга

   Принятое слово $\{r[n]\} = \{11, 01, 11, 01, 10, 10, 01, 10, 11\}$



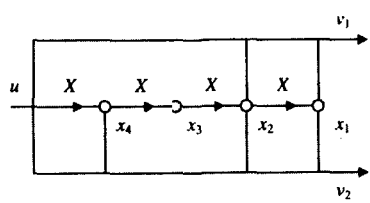
   Продекодированная информационная последовательность

$$\{u'[n]\} = \{1, 0, 0, 1, 1, 0, 1\}$$

6. Если переданное слово продекодировано правильно, то метрику полученного кодового слова можно интерпретировать как число ошибочных бит и в этом случае метрика может быть использована для оценки качества канала.

Решение задания 4.2. Сверточный код, используемый для кодирования речи в сетях мобильной связи GSM.

1. Схема кодера



2. К информационной последовательности необходимо добавить 4 нуля для того, чтобы кодирование заканчивалось нулевым состоянием.

3. Память кодера $m = 4$.

   Кодовое ограничение $n_c = 10$.

Блоковая скорость $185/(2[185+4]) = 0,489$

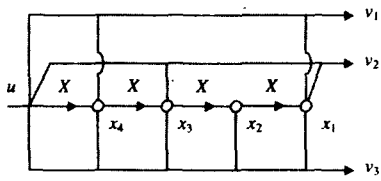
Относительная потеря скорости $4/(185+4) = 2,11 \cdot 10^{-2}$

Полная память кодера $M = 4$.

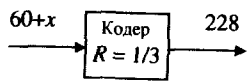
4. Существует $2^M = 16$ состояний. На каждом шаге декодер вычисляет 32 приращения метрик, исходя из которых вычисляются тридцать две частичных метрики и сравниваются попарно; 16 выживших путей выбираются для продолжения и их частичные метрики запоминаются.

Решение задания 4.3. Сверточный код для передачи данных в мобильных сетях связи GSM.

1. Схема кодера



2. Из равенства $3 \cdot (60+x) = 228$ следует, что $x = 16$, т.е. к информационной последовательности добавляются 16 нулей и кодирование заканчивается нулевым состоянием.



3. Память кодера $m = 4$. Кодовое ограничение $n_c = 3 \cdot (4+1) = 15$.

Блоковая скорость $60/(3 \cdot [60+4]) = 0,3125$

– учитывая 16 нулевых бит получаем $60/228 = 0,263$

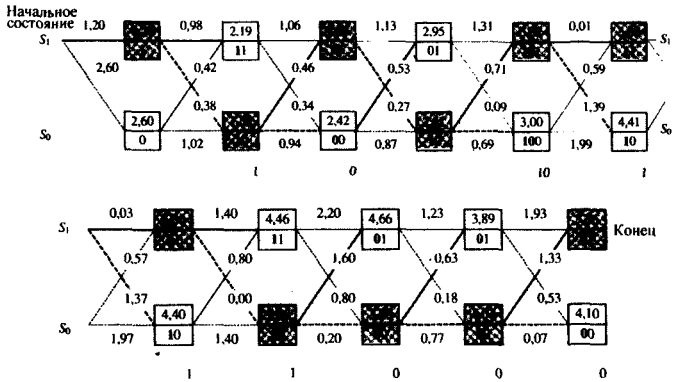
Относительная потеря скорости $4/64 = 6,25 \cdot 10^{-2}$

– учитывая 16 нулевых бит получаем $16/76 = 0,211$

Полная память кодера $M = 4$.

4. Существует $2^M = 16$ состояний. В каждом такте декодер вычисляет 32 приращения метрик, исходя из которых вычисляются и попарно сравниваются 32 частичные метрики; 16 полученных путей выбираются для продолжения и их частичные метрики запоминаются.

Решение задания 4.4. Детектор Витерби с субоптимальной аддитивной метрикой.



Таким образом, при аддитивной метрике информационная последовательность также детектируется без ошибок.

ГЛАВА 5

ДИСКРЕТНЫЕ ПРЕОБРАЗОВАНИЯ ФУРЬЕ И КОДЫ РИДА – СОЛОМОНА

5.1. Введение

В предыдущих главах мы рассматривали только двоичные коды, то есть коды с символами из $GF(2)$. Между тем, коды над алфавитом $GF(q)$, где $q > 2$, так называемые q-ичные коды, помимо чисто теоретического интереса, имеют в настоящее время огромное прикладное значение. Мы ограничимся только рассмотрением кодов Рида – Соломона, как наиболее ярких представителей q-ичных кодов, без которых немыслима современная теория и практика помехоустойчивого кодирования.

Остановимся, прежде всего, на прикладном значении кодов Рида – Соломона. Как было показано в предыдущих главах, простейшие алгебраические двоичные коды могут успешно применяться для передачи данных в компьютерных сетях связи, обеспечивая необходимую надежность передаваемой информации. Это объясняется достаточно высоким качеством проводных и оптоволоконных каналов связи и возможностью переспроса передаваемых блоков данных. Задача кодирования для таких каналов, в основном, сводится только к обнаружению независимых ошибок и пакетов ошибок.

Все многообразие современных коммуникационных линий связи не исчерпывается только компьютерными сетями. В качестве примеров достаточно привести спутниковые каналы и мобильную цифровую связь. Здесь уже переспросы не допустимы и корректирующая способность кода определяется максимальным числом исправляемых ошибок. Предельная мощность передаваемого сигнала в таких каналах жестко ограничивается, что приводит к высокой вероятности появления как независимых ошибок, так и пакетов ошибок. В таких каналах использование рассмотренных нами простейших двоичных кодов не имеет смысла. Так, например, коды Хэмминга с $d_{\min} = 3$ и $d_{\min} = 4$ при наличии в блоке более, чем одной ошибки, вместо исправления ошибок будут вносить новые. Проблему, в

какой-то степени, могут решить низкоскоростные двоичные коды Боуза – Чоудхури – Хоквингема, однако, их реализация и декодирование аналогично *кодам Рида – Соломона* осуществляется с помощью операций над $GF(2^m)$ при $m > 1$.

С другой стороны, турбо-коды, построенные на базе двоичных сверточных кодов, хотя и позволяют обеспечить требуемую надежность при кодовых скоростях, близких к пропускной способности каналов с независимыми ошибками, не расчитаны на борьбу с длинными пакетами ошибок. Кроме этого, для их эффективной реализации требуются блоки очень большой длины (десятки и даже сотни кбит), что не всегда технически приемлемо.

Решению подобных задач в немалой степени способствовало появление в 1968 г. кодов Рида – Соломона с символами из $GF(q)$, где $q > 2$ [20]. Коды Рида – Соломона с параметрами (n, k) имеют минимальное расстояние $d_{\min} = n-k+1$ и способны исправлять $](n-k)/2[$ ошибок. После открытия Берлекэмпом в 1968 г. простого и эффективного алгоритма декодирования кодов Рида – Соломона, эти коды прочно вошли в практику помехоустойчивого кодирования.

В первую очередь в 60$^{\text{ых}}$ годах коды Рида – Соломона стали применяться в качестве внешних кодов в каскадных конструкциях, используемых в спутниковых линиях связи. В таких конструкциях [21] q-ичные символы кодов Рида – Соломона (один или несколько) кодируются внутренними двоичными сверточными кодами. При декодировании сверточных кодов используется мягкое решение, особенно эффективное в каналах с АБГШ. Так как шум в реальных каналах всегда отличается от гауссовского, в спутниковых каналах возможно появление пакетов ошибок. Такие пакеты могут привести к ошибочному декодированию внутренними сверточными кодами одного или нескольких довольно длинных блоков. Для внешних кодов Рида – Соломона это, в основном, эквивалентно появлению ошибочных q-ичных символов небольшой кратности, лежащих в пределах корректирующей способности внешнего кода. Таким образом, весь каскадный код, даже при наличии пакетов ошибок, в подавляющем большинстве случаев декодируется правильно, что обеспечивает необходимую надежность передаваемой информации. Самое удивительное заключается в том, что четкой альтернативы каскадным кодам с внешними кодами Рида – Соломона для спутниковых линий связи до сих пор найти не удалось и коды Рида – Соломона являются неотъемлемой частью большинства стандартов (например Intelsat IESS-308). Кроме этого коды Рида – Соломона имеют самостоятельное прак-

тическое применение. Они, например, являются практически оптимальными при записи информации на носители аудио-CD, что обусловлено техническими характеристиками и характером ошибок при записи.

Структура кодов Рида – Соломона и алгоритм декодирования наиболее просто описывается с помощью спектральных методов [5]. Арифметика полей Галуа $GF(q)$ описана в приложении 2.5.

5.2. Дискретные преобразования Фурье в поле Галуа

Определение. Пусть задан вектор $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ над $GF(q)$, где $q = 2^m$, а $n = q - 1$ и пусть α – примитивный элемент поля Галуа $GF(q)$ характеристики 2. Преобразование Фурье в поле Галуа вектора $\mathbf{v}$ определяется как вектор $\mathbf{V} = (V_0, V_1, \ldots, V_{n-1})$, задаваемый равенствами

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i, \qquad j = 0, \ldots, n-1. \tag{5.1}$$

По аналогии с преобразованиями Фурье непрерывных сигналов, дискретный индекс i принято называть временем и говорить о том, что вектор $\mathbf{v}$ принадлежит временной области. Естественно, что в этом случае индекс j называется частотой и говорят, что вектор $\mathbf{V}$ определен в частотной области.

Теорема 5.2.1. Над полем $GF(q)$ характеристики 2 вектор $\mathbf{v}$ во временной области и вектор $\mathbf{V}$ в частотной области связаны соотношениями

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i, \tag{5.2}$$

$$v_i = \sum_{j=0}^{n-1} \alpha^{-ij} V_j. \tag{5.3}$$

По аналогии с непрерывными сигналами будем называть преобразование (5.2) прямым преобразованием Фурье, а (5.3) – обратным преобразованием.

Вектор $\mathbf{v}$ иногда задается многочленом $v(X)$. С помощью преобразования Фурье в поле Галуа многочлен

$$v(X) = v_0 + v_1 X + \cdots + v_{n-1} X^{n-1} \tag{5.4}$$

может быть преобразован в многочлен

$$V(X) = V_0 + V_1 X + \cdots + V_{n-1} X^{n-1}, \tag{5.5}$$

который называется *спектральным многочленом* или многочленом в частотной области вектора $\mathbf{v}$.

Теорема 5.2.2.

1. j-ая частотная компонента V_j равна нулю тогда и только тогда, когда элемент α^j является корнем многочлена $v(X)$;

2. i-ая временная компонента v_i равна нулю тогда и только тогда когда элемент α^{-i} является корнем многочлена $V(X)$.

Доказательство. Доказательство утверждений 1. и 2. очевидны, так как

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i = v(\alpha^j) \tag{5.6}$$

и

$$v_i = \sum_{j=0}^{n-1} \alpha^{-ij} V_j = V(\alpha^{-i}). \tag{5.7}$$

■

Преобразование Фурье обладает многими важными свойствами, которые переносятся на случай конечных полей. Помимо линейности преобразований Фурье для нас важным является свойство свертки. Прежде чем сформулировать теорему о свертке, введем некоторые определения.

Пусть в поле $GF(q)$ заданы векторы $\mathbf{g} = (g_0, g_1, \ldots, g_{n-1})$ и $\mathbf{d} = (d_0, d_1, \ldots, d_{n-1})$, многочлены которых имеют вид $g(X) = g_0 + g_1(X) + \cdots + g_{n-1}X^{n-1}$ и $d(X) = d_0 + d_1(X) + \cdots + d_{n-1}X^{n-1}$. Тогда компоненты вектора линейной свертки $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ векторов $\mathbf{g}$ и $\mathbf{d}$ определяются как

$$c_i = \sum_{k=0}^{n-1} g_{i-k} d_k, \qquad i = 0, \ldots, n-1$$

и многочлен линейной свертки $c(X)$ может быть записан в виде произведения

$$c(X) = g(X)d(X).$$

Операцию линейной свертки двух векторов мы уже рассматривали в разделе 4.2.

Циклическая свертка может быть записана в виде

$$c_i = \sum_{k=0}^{n-1} g_{((i-k))} d_k, \qquad i = 0, \ldots, n-1,$$

где двойные скобки означают вычисление индексов по модулю $n = q - 1$ (заметим, что в арифметике по $\bmod n$ имеет место равенство $-k = n - k$).

Многочлен циклической свертки имеет вид

$$c(X) = g(X)d(X)(\mod X^n - 1).$$

Для того, чтобы избежать путаницы, будем обозначать операцию циклической свертки символом «$*$», а операцию линейной свертки – символом «$\circledast$». Наконец, операцию покомпонентного умножения двух векторов $\mathbf{c} = \mathbf{gd}$ определим как

$$c_i = g_i d_i.$$

Заметим, что все введеные выше операции обладают свойством линейности.

Теорема 5.2.3. Теорема о свертке.

Пусть во временной области заданы векторы $\mathbf{f} = (f_0, f_1, \ldots, f_{n-1})$ и $\mathbf{g} = (g_0, g_1, \ldots, g_{n-1})$, прямые преобразования Фурье которых в частотной области имеют вид $\mathbf{F} = (F_0, F_1, \ldots, F_{n-1})$ и $\mathbf{G} = (G_0, G_1, \ldots, G_{n-1})$, тогда покомпонентному произведению векторов в частотной области $\mathbf{E} = \mathbf{FG}$ взаимно однозначно соответствует их циклическая свертка во временной области $\mathbf{e} = \mathbf{f} * \mathbf{g}$, т.е. если $\mathbf{E} = \mathbf{FG}$, то

$$e_i = \sum_{k=0}^{n-1} f((i-k))g_k, \qquad i = 0, \ldots, n-1.$$

Справедлива так же обратная теорема. Для ее формулировки нужно только поменять местами временную и частотные области.

Изложенных выше сведений из теории дискретных преобразований Фурье вполне достаточно для изучения структуры кодов Рида – Соломона и алгоритма их декодирования.

5.3. Коды Рида – Соломона

Прежде, чем приступить к изложению теории кодов Рида - Соломона, введем понятие кода с максимальным расстоянием.

Теорема 5.3.1. Граница Синглтона.

Минимальное расстояние $d_{\min}$ любого (n, k)-кода (необязательно линейного) удовлетворяет неравенству

$$d_{\min} \leq n - k + 1.$$

Доказательство. Пусть символы кода принадлежат полю $GF(q)$. В множестве q^k кодовых слов выделим и зафиксируем $k-1$ разрядов. Эти разряды могут содержать самое большее q^{k-1} различных q-ичных чисел. Следовательно, во всем множестве кодовых слов в выделенных $k-1$ разрядах имеет место по крайней мере $q^k - q^{k-1}$ совпадений. Рассмотрим любые два кодовых слова, совпадающие между собой в $k-1$ разрядах. Так как эти слова могут иметь различие только в $n-k+1$ компонентах, расстояние между ними не может превышать величины $n-k+1$. ■

Определение. Любой код с минимальным расстоянием, удовлетворяющим равенству

$$d_{\min} = n - k + 1$$

называется кодом с максимальным расстоянием.

Определение. Код, который может быть приведен к систематическому виду путем операций, не изменяющих дистанционный профиль кодовых слов, называется разделимым.

Так как линейный код может быть приведен к систематическому виду элементарными преобразованиями порождающей матрицы, любой линейный код является разделимым.

Определение. Разделимый код с максимальным расстоянием называется МДР кодом (разделимым кодом с максимальным расстоянием).

Теперь вернемся к кодам Рида – Соломона.

Определение. Кодом Рида – Соломона называется линейный циклический $(n, n-d+1)$-код над $GF(q)$, где $q = p^m$, длины $n = q-1$, порождающий многочлен которого $g(X)$ имеет своими корнями $d-1$ последовательных степеней примитивного элемента α из поля $GF(q)$.

В качестве порождающего многочлена кода Рида Соломона можно выбрать, например

$$g(X) = (X - \alpha)(X - \alpha^2)\cdots(X - \alpha^{d-1}).$$

Замечание. *Так как мы ограничиваемся только полями характеристики 2, то будем в дальнейшем вместо операций вычитания использовать операцию сложения.*

В теории помехоустойчивого кодирования доказывается, что свойства (n, k)-кода Рида – Соломона с символами из $GF(q)$ и параметрами $n = q - 1$, $k = n - d + 1$ не зависят от метода его построения и определяются только выбранными значениями q и d. Наиболее просто код Рида – Соломона, а так же алгоритм его декодирования реализуется на основе дискретных преобразований Фурье, рассмотренных нами в предыдущем разделе. Процедуры кодирования и декодирования такого кода могут быть значительно ускорены с помощью техники быстрых преобразований Фурье (БПФ).

Выберем и зафиксируем некоторое поле Галуа $GF(q)$ с $q = 2^m$, примитивный элемент $\alpha \in GF(q)$ и параметр d. Рассмотрим информационный вектор $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ длины $k = n - d + 1 = q - d$ с компонентами из $GF(q)$. Поставим в соответствие вектору $\mathbf{u}$ вектор

$$\mathbf{v} = (v_0, v_1, \ldots, v_{k-1}, \underbrace{0, 0, \ldots, 0}_{n-k})$$

длины $n = q - 1$, у которого первые k компонент совпадают с компонентами вектора $\mathbf{u}$, а остальные компоненты – нулевые. Рассмотрим прямое преобразование Фурье вектора $\mathbf{v}$ в вектор $\mathbf{V}$, определяемое (5.1) и удовлетворяющее теореме 5.2.1. Тогда справедлива следующая теорема:

Теорема 5.3.2. Множество q^k векторов $\mathbf{V}$ в частотной области образует (n, k)-код Рида Соломона.

Доказательство. Представим векторы $\mathbf{v}$ и $\mathbf{V}$ в виде многочленов

$$v(X) = v_0 + v_1 X \cdots + v_{k-1} X^{k-1} + 0 \cdot X^k + 0 \cdot X^{k+1} + \cdots + 0 \cdot X^{n-1}$$

$$V(X) = V_0 + V_1 X \cdots + V_{n-1} X^{n-1}.$$

Так как i-ые временные компоненты вектора $\mathbf{v}$ равны нулю при $k \le i \le n - 1$, то, согласно теореме 5.2, многочлен $V(X)$ имеет корни $\alpha^{-k} = \alpha^{d-1}$, $\alpha^{-(k+1)} = \alpha^{d-2}, \ldots, \alpha^{-(n-1)} = \alpha$. Таким образом, много-

члены $V(X)$ образуют (n, k)-код Рида – Соломона с порождающим многочленом

$$g(X) = (X + \alpha)(X + \alpha^2) \cdots (X + \alpha^{d-1}).$$

■

Спектральный подход позволяет легко доказать следующую теорему.

Теорема 5.3.3. Любой код Рида – Соломона является МДР кодом.

Доказательство. Так как код Рида – Соломона является линейным, $d_{\min}$ равно минимальному весу кодового слова. Любое кодовое слово $\mathbf{V} = (V_0, V_1, \ldots V_j, \ldots, V_{n-1})$ является прямым преобразованием Фурье некоторого вектора

$$\mathbf{v} = (v_0, v_1, \ldots, v_{k-1}, \underbrace{0, 0, \ldots, 0}_{n-k}).$$

В силу теоремы 5.2.2, j-ая частотная компонента равна нулю тогда и только тогда, когда α^j является корнем многочлена $v(X) = v_0 + v_1 X + \cdots + v_{k-1} X^{k-1}$. Согласно основной теореме алгебры, которая справедлива и для конечных полей Галуа, многочлен $v(X)$ степени $k-1$ может иметь в поле $GF(q)$ не более $k-1$ корней. Следовательно, вес любого слова (n, k)-кода Рида – Соломона не может быть меньше, чем $n-(k-1) = n-k+1$ и, в силу границы Синглтона $d_{\min} = n-k+1$, то есть код обладает максимальным расстоянием. Так как коды Рида – Соломона линейны, они являются МДР кодами. ■

Спектральный подход позволяет также достаточно просто интерпретировать процедуру декодирования кодов Рида – Соломона.

5.4. Декодирование кодов Рида – Соломона

Не трудно заметить, что при образовании кодов Рида – Соломона информационные символы можно размещать в любых k рядом стоящих разрядах вектора $\mathbf{v}$. Из методических соображений отведем под информацию старшие компоненты $\mathbf{V}$. В этом случае многочлен $v(X)$ будет иметь вид

$$v(X) = v_{n-1} X^{n-1} + v_{n-2} X^{n-2} + \cdots + v_{n-k} X^{n-k} + 0 \cdot X^{n-k-1} + \cdots + 0 \cdot X^0,$$

а порождающий многочлен соответствующего кода Рида – Соломона имеет вид

$$g(X) = (X + \alpha^{k+1})(X + \alpha^{k+2}) \cdots (X + \alpha^n).$$

Как уже отмечалось ранее, слова рассматриваемого кода Рида - Соломона являются прямым преобразованием Фурье множества векторов $\mathbf{V}$, то есть $\mathbf{v} \rightleftarrows \mathbf{V}$. В канале кодовому слову $\mathbf{V}$ добавляется вектор ошибки $\mathbf{E}$ кратности $l \leq d-1$.

Рассмотрим обратное преобразование Фурье принятого из канала слова

$$\mathbf{R} = \mathbf{V} + \mathbf{E}, \tag{5.8}$$

В силу свойства линейности обратного преобразования Фурье имеем

$$(v_{n-1} + e_{n-1}, \ldots, v_{n-k} + e_{n-k}, e_{n-k-1}, e_{n-k-2}, \ldots, e_0) \leftrightarrows \mathbf{V} + \mathbf{E}, \tag{5.9}$$

где $\mathbf{v} = (v_{n-k}, v_{n-k-1}, \ldots, v_{n-1})$ - информационный вектор, лежащий во временной области, $\mathbf{e} = (e_0, e_1, \ldots, e_{n-1})$ - обратное преобразование вектора ошибок. Так как $n-k$ правых компонент вектора обратного преобразования Фурье от $\mathbf{V}+\mathbf{E}$ не зависят от кодового слова $\mathbf{V}$, эти компоненты образуют синдром ошибок. Запишем этот синдром в виде

$$s_{d-2}, s_{d-1}, \ldots, s_1, s_0, \tag{5.10}$$

где $s_0 = e_0, s_1 = e_1, \ldots, s_{d-2} = e_{n-k-1}$. Заметим, что синдром является некоторым «окном», через которое можно наблюдать обратное преоразование Фурье вектора ошибки $\mathbf{E}$.

Обозначим индексы l ненулевых компонент вектора $\mathbf{E}$ через $j_1, j_2, \ldots, j_l$. Определим вектор во временной области, прямое преобразование Фурье которого содержит нулевые компоненты для всех частот j, для которых $E_j \neq 0$. Проще всего такой вектор задать в виде многочлена локаторов ошибок

$$\sigma(X) = \prod_{k=1}^{l} (1 - X\alpha^{-j_k}) = \sigma_0 + \sigma_1 X + \ldots + \sigma_l X^l. \tag{5.11}$$

Покомпонентное произведение прямого преобразования Фурье от многочлена $\sigma(X)$ на вектор ошибки $\mathbf{E}$ в частотной области равно нулю, следовательно циклическая свертка во временной области вектора $\boldsymbol{\sigma}$ с векторм $\mathbf{e}$ также равна нулю

$$\boldsymbol{\sigma} * \mathbf{e} = \mathbf{0}. \tag{5.12}$$

Из (5.9) и (5.10) следует, что для определения компонент $\sigma_1, \sigma_2, \ldots, \sigma_l$

(согласно (5.11) $\sigma_0 = 1$), используя (5.12), мы можем составить систему $d-1-l$ линейных уравнений с l неизвестными

$$\begin{aligned}
&\sigma_0 s_l + \sigma_1 s_{l-1} + \ldots + \sigma_l s_0 = 0\\
&\sigma_0 s_{l+1} + \sigma_1 s_l + \ldots + \sigma_l s_1 = 0\\
&\vdots\\
&\sigma_0 s_{d-2} + \sigma_1 s_{d-3} + \ldots + \sigma_l s_{d-2-l} = 0.
\end{aligned} \tag{5.13}$$

Сложность, возникающая при решении системы уравнений (5.13) состоит в том, что кратность ошибки l нам заранее не известна. Таким образом, в процессе решения должна еще осуществляться и минимизация значения l, при котором возможно выполнение всех $d-1-l$ уравнений из (5.13). Такой простой и эффективный алгоритм был найден Берлекэмпом в 1967 г. Без преувеличения можно сказать, что этот алгоритм произвел настоящую революцию в теории и практике помехоустойчивого кодирования. Он будет рассмотрен в следующем разделе этой главы.

Предположим, что все коэффициенты многочлена $\sigma(X)$ определены. В этом случае остальные k компонент вектора $\mathbf{e}$, то есть компоненты $e_{n-k}, e_{n-k+1}, \ldots, e_{n-1}$ во временной области, могут быть рекурентно определены, исходя из (5.12).

Для определения компоненты e_{k-1} нам достаточно решить уравнение

$$\sigma_0 e_{n-k} + \sigma_1 s_{d-2} + \ldots + \sigma_{l-1} s_{d-2-l} + \sigma_l s_{d-1-l} = 0$$

относительно e_{k-1}, так как все остальные компоненты уже определены.

На втором шаге мы уже можем составить уравнение для определения e_{k-2}. Это уравнение имеет вид

$$\sigma_0 e_{n-k+1} + \sigma_1 e_{n-k} + \sigma_2 s_{d-2} + \ldots + \sigma_l s_{d-l} = 0.$$

Рекурентно продолжая описанную процедуру, мы найдем все оставшихся компоненты вектора $\mathbf{e}$.

Для определения информационного вектора $\mathbf{v}$ нам достаточно вычесть найденные компоненты $e_{n-k}, e_{n-k+1}, \ldots, e_{n-1}$ из полученных обратным преобразованием Фурье от $\mathbf{R}$ значений $v_{n-k} + e_{n-k}, v_{n-k+1} + e_{n-k+1}, \ldots, v_{n-1} + e_{n-1}$.

Самое замечательное в рассмотренном алгоритме состоит в том, что при его реализации не приходится находить ни корни многочлена $\sigma(X)$ (локаторы ошибок), ни значения ошибок.

Теперь общая картина процедуры декодирования ясна и ее можно сформулировать в виде пяти шагов.

Шаг 1. Вычислить обратное преобразование Фурье принятого вектора $\mathbf{R} = \mathbf{V} + \mathbf{E}$. Выделить $s_0, \ldots, s_{d-2}$ во временной области и зашумленные компоненты информационного вектора $\mathbf{v}$;

Шаг 2. Найти $\sigma(X)$ из (5.13);

Шаг 3. С помощью рекурентной процедуры вычислить компоненты $e_{n-k}, e_{n-k+1}, \ldots, e_{n-1}$;

Шаг 4. Найти информационный вектор $\mathbf{v}$;

Замечание. *Рассмотренный алгоритм всегда исправляет* $\lfloor \frac{d-1}{2} \rfloor$ *и менее ошибок. В следующем разделе будет рассмотрен также случай, когда число канальных ошибок превышает* $\lfloor \frac{d-1}{2} \rfloor$.

Для контроля правильности декодирования часто проводится следующий шаг.

Шаг 5. Вычислить прямое преобразование Фурье вектора $\mathbf{v}$, получив при этом кодовай вектор $\mathbf{V}'$ (не всегда совпадающий с $\mathbf{V}$).

5.5. Итеративный алгоритм для нахождения $\sigma(X)$

Будем руководствоваться блок-схемой алгоритма, приведенной на рис. 5.1 ([10] стр. 271). Отметим, что эта блок-схема содержит всю необходимую информацию для программной или схемной реализации алгоритма. Если число ошибок не превышает величины $\lfloor \frac{d-1}{2} \rfloor$, то алгоритм находит многочлен $\sigma(X)$ минимальной степени l, при котором выполняется система уравнений (5.13). Доказательству этого факта в [10] посвящено несколько довольно сложных теорем и заинтересованному читателю мы советуем обратится к [10]. Ограничимся описанием основных принципов работы алгоритма.

В работе алгоритма используются три регистра. В регистре динамических связей R_1 на каждом такте декодирования содержится очередное итеративное значение $\sigma_n(X)$. Компоненты многочлена $\sigma_n(X)$ на n-ом такте декодирования $C_{n,1}, C_{n,2}, \ldots, C_{n,j}$, где j – степень многочлена.

Рассмотрим процесс вычисления $\sigma_n(X)$ более подробно. В регистр сдвига R_2 последовательно, на каждом такте декодирования

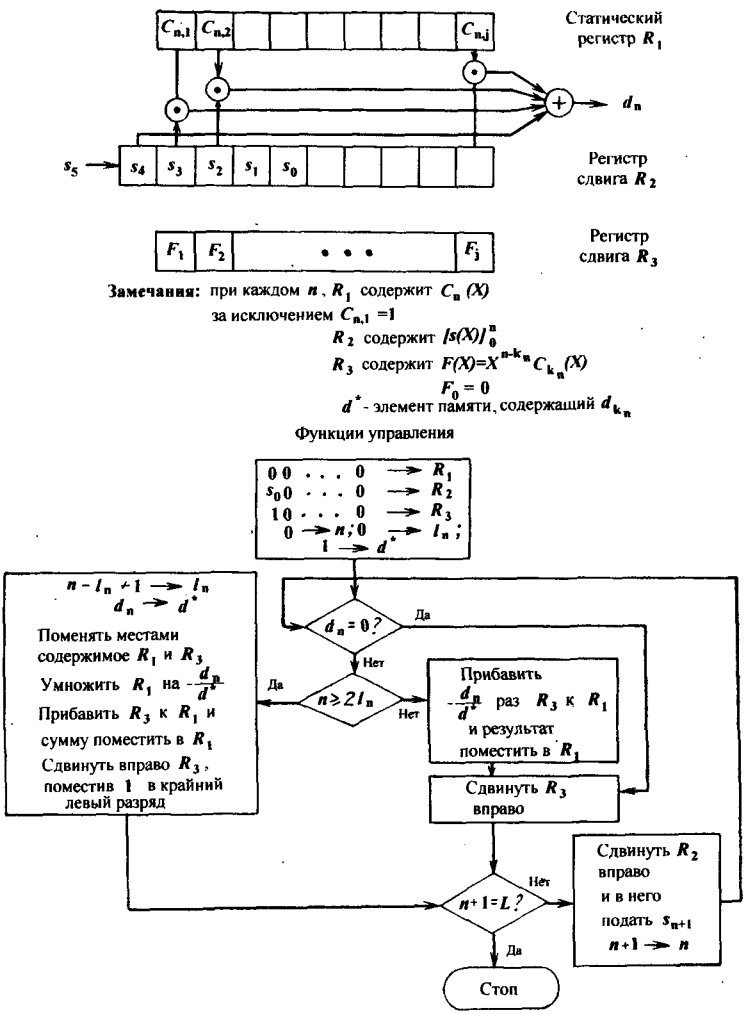


Рис. 5.1. Алгоритм Берлекэмпа Месси [10].

подаются компоненты синдрома $s_0, s_1, \ldots, s_{d-2}$. Пусть l_n – итеративная длина регистра R_1 на такте n, тогда на этом же такте на выходе вычисляется

$$d_n = C_{n,1}s_{l_n} + C_{n,2}s_{l_n-1} + \cdots + C_{n,l}s_0$$

и декодер стремится так скорректировать $\sigma_n(X)$, чтобы реализовать первое уравнение из системы (5.13). Далее декодер добивается выполнения второго уравнения из (5.13) при сохранении первого и т.д. Оказывается, что процесс коррекции можно начинать прямо с первого такта, то есть с $d_0 = s_0$. В результате мы получим $\sigma(X)$ минимальной степени, при котором выполняются все уравнения из (5.13).

Контроль правильности всего процесса декодирования происходит на пятом шаге декодирования. Пусть на этом шаге вычислен кодовый вектор $\mathbf{V}'$. Сравнивая $\mathbf{V}'$ с $\mathbf{R} = \mathbf{V} + \mathbf{E}$ можно найти число исправлений. Если в канале произошло не более $\lfloor \frac{d-1}{2} \rfloor$ ошибок, то число исправлений должно совпадать со степенью многочлена $\sigma(X)$.

Если число канальных ошибок превышает $\lfloor \frac{d-1}{2} \rfloor$, то может произойти одно из трех событий. Во-первых, степень многочлена $\sigma(X)$ может оказаться выше $\lfloor \frac{d-1}{2} \rfloor$. Тогда процесс декодирования прерывается уже на втором этапе декодирования. Во-вторых, число исправлений может не совпадать со степенью $\sigma(X)$, такая ошибка обнаруживается в конце декодирования. И, наконец, вполне возможно, что число исправлений совпадает со степенью $\sigma(X)$ и не превышает $\lfloor \frac{d-1}{2} \rfloor$. Здесь имеет место необнаружимая ошибка декодирования.

Заметим, что для реализации этого алгоритма требуются незначительные технические затраты и в настоящее время декодеры кодов Рида - Соломона работают на скоростях до 40 Гбит/сек.

Литература

[1] C. E. Shenon: A mathematical theory of communication Bell Sys. Tech. J., vol 27, 1948, S.379-423, (Имеется перевод: Шеннон К. Математическая теория связи.-В сб.: Работы по теории информации и кибернетики. -М.: ИЛ, 1963)

[2] C. Berrou, A. Glavieux: "Near Optimum Error-Correcting Coding and Decoding: Turbo Codes."IEEE Trans. Commun. Vol. 44, 1996, S.1261-1270

[3] C. Berrou, A. Glavieux, P.Thitimajshima: "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes."Proc. IEEE Int. Conf. Commun., May 1993, S. 1064-1070

[4] R. E. Blahut: Principles and practice of information theory, Reading, Mass.: Addison-Wesley Publishing Company,1987

[5] R. E. Blahut: Principles and practice of error control coding, Addison-Wesley Publishing Company,1984 (Имеется перевод: Блейхут. Р. Теория и практика кодов, контролирующих ошибки. -М.: Мир, 1986)

[6] D. A. Huffman: "A method for the Construction of Minimum Redunduncy Codes."Proc. IRE, vol. 40,1952,S.1098-1101, (Имеется перевод: Хаффман Д. А. Методы построения кодов с минимальной избыточностью.-Кибернетический сб. вып. 3.-М.: ИЛ, 1961)

[7] S. Lin, D. J. Costello: Error control coding: Fundamentals and Applications, Englewood Cliffs, NJ,: Prentice- Hall Inc., 1983

[8] F. J. MacWilliams, N. J. Sloane: The Theory of Error-Correcting Codes. Amsterdam: North-Holland, 1977, (Имеется перевод: Мак-Вильямс Ф. Дж., Слоэн Н. Дж. А. Теория кодов исправляющих ошибки. -М.: Связь, 1979)

[9] N. Wiener: Cybernetics or Control and Communication in the Animal and the Mashine. Paris: Hermann, 1948

[10] R. G. Gallager: Information Theory and Reliable Communication. New York: John Willey and Sons Inc. 1968, (Имеется перевод: Галагер Р. Теория информации и надежная связь.-М.: Сов. радио, 1974)

[11] W. W. Peterson, E. J. Weldon: Error- Correcting Codes. 2nd ed. -Cambrige (Mass.): MIT Press, 1971 (Имеется перевод: Питерсен У., Уэлдон Э. Коды, исправляющие ошибки. -М.: Мир, 1976)

[12] G. Ungerböck: Channel Coding with Multilevel/phase Signals. IEEE Trans. Inform. Theory, IT-28,1982, S.55-67.

[13] J. G.Proakis: Digital Communications. New York: McGraw-Hill, 2000.

Литература, добавленная при переводе

[14] I. Chisar, J. Kërner: Information theory, Coding theorems for discrete memoryless systems. Akademiai Budapest, 1981 (Имеется перевод: Чисар И., Кернер Я. Теория информации, теоремы кодирования для дискретных систем без памяти. -М.: Мир, 1985)

[15] W. Feller: An introduction to probabilistic theory and its application. John Willey & Sons, NY, 1970 (Имеется перевод: Феллер В. Введение в теорию вероятностей и ее приложения. -М.: Мир, 1984)

[16] J. M. Wozencraft, I. Jacobs: Principles of communication engineering. John Willey & Sons, NY, 1965 (Имеется перевод: Возенкрафт Дж., Джекобс И. Теоретические основы техники связи. -М.: Мир, 1969)

[17] J. J. Stiffler: Theory of synchronous communications. NJ: Prentice Hall Inc., 1971, (Имеется перевод: Стиффлер Дж. Теория синхронной связи. -М.: Связь, 1978)

[18] G. C. Klark, J. B. Cain: Error correction coding for digital communication, Plenum press, 1982 (Имеется перевод: Кларк Дж., Кейн Дж. Кодирование с исправлением ошибок в системах цифровой связи. -М.: Радио и связь, 1987)

[19] Д. Сэломон: Практическое руководство по методам сжатия данных. М.: Техносфера, 2003

[20] E. R. Berlekamp: Algebraic Coding Theory. NY, McGraw-Hill,1968 (Имеется перевод: Берлекэмп Э. Алгебраическая теория кодирования. -М.: Мир, 1971)

[21] G. D. Jr. Forney: Concantenated Codes, Cambrige, Mass., MIIT Press,1963 (Имеется перевод: Форни Д. Каскадные коды. -М.: Мир, 1970)

Предметный указатель