ТЕОРИЯ ИНФОРМАЦИИ

Б.Д. Кудряшов

2009

Оглавление

Введение

Теория информации – наука с точно известной датой рождения. Ее появление на свет связывают с опубликованием Клодом Шенноном работы "Математическая теория связи" в 1948 г. [21]. С точки зрения Шеннона, теория информации – раздел математической теории связи. Поэтому круг задач теории информации мы поясним с помощью представленной на рис.1. структурной схемы типичной системы передачи или хранения информации. В этой схеме под источником понимается любое устройство

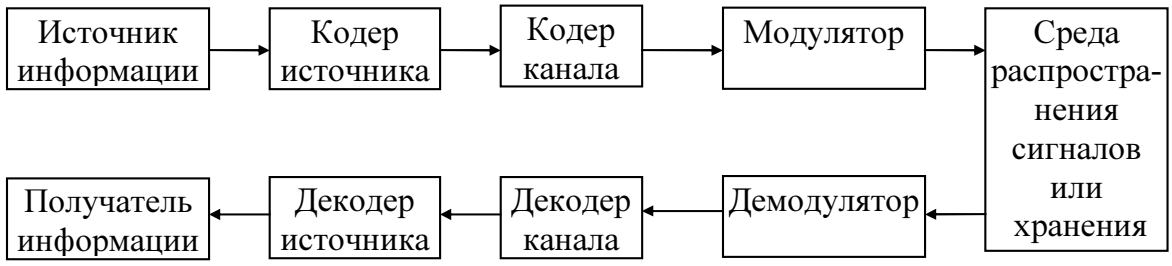


Рис. 1: Блок-схема системы связи.

или объект живой природы, порождающие сообщения, которые должны быть перемещены в пространстве или во времени. Это может быть клавиатура компьютера, человек, аналоговый выход видеокамеры и т. п. Поскольку, независимо от изначальной физической природы, все подлежащие передаче сообщения обычно преобразуется в форму электрических сигналов, именно такие сигналы мы и будем рассматривать как выход источника.

Цель кодера источника – представление информации в наиболее компактной форме. Это нужно для того, чтобы максимально эффективно использовать ресурсы канала связи либо запоминающего устройства.

Далее следует кодер канала, задачей которого является обработка информации с целью защиты сообщений от помех при передаче по каналу связи либо возможных искажений при хранении информации. Модулятор служит для преобразования сообщений, формируемых кодером

канала, в сигналы, согласованные с физической природой канала связи или средой накопителя информации.

Остальные блоки, расположенные на приемной стороне, выполняют обратные операции и предоставляют получателю информацию в удобном для использования виде.

Итак, в теории информации можно выделить следующие разделы:

- *Кодирование дискретных источников.* Иногда эту часть теории информации называют кодированием без потерь, кодированием для канала без шума, сжатием информации.

- *Кодирование информации для передачи по каналу с шумом.* Речь идет о защите информации от помех в каналах связи.

- *Кодирование с заданным критерием качества.* В некоторых системах связи искажения информации допустимы. Более того, информация аналоговых источников вообще не может быть представлена в цифровой форме без искажений. В данном разделе речь идет о методах кодирования, обеспечивающих наилучший компромисс между качеством (оцениваемым некоторой объективной мерой искажения) и затратами на передачу информации. Сегодня задачи такого типа стали особенно актуальны, поскольку они находят широкое применение для цифровой передачи речи, видео- и аудиоинформации.

- *Кодирование информации для систем со многими пользователями.* Здесь обсуждается оптимальное взаимодействие абонентов, использующих какой-либо общий ресурс, например, канал связи Примеры систем с множественным доступом – системы мобильной связи, локальные сети ЭВМ.

- *Секретная связь, системы защиты информации от несанкционированного доступа.* Здесь также можно указать широкий круг актуальных задач, лежащих на стыке теории информации, теории вычислительной сложности алгоритмов, исследования операций, теории чисел.

Из перечисленных разделов в данном пособии кратко рассматриваются первые три.

Глава 1

Измерение информации дискретных источников

1.1 Измерение информации. Собственная информация

Разумной мерой информации, содержащейся в сообщении является мера, монотонно связанная с затратами на передачу сообщения.

Условимся о том, что сообщения представляют собой случайные события. Рассмотрим в качестве источника произвольное дискретное множество X и каждой букве $x \in X$ припишем вероятность $p(x)$.

Сформулируем требования к некоторой мере $\mu(x)$, определенной для всех $x \in X$, которую следует принять как меру информации, содержащейся в сообщениях ансамбля $X = \{x, p(x)\}$.

- Поскольку предполагается, что эта мера будет определять затраты, связанные с передачей или хранением сообщений, мера должна быть неотрицательной.

- Поскольку для нас несущественно, каким образом будут интерпретированы и использованы передаваемые сообщения, мера должна однозначно определяться вероятностью сообщения. Поэтому вместо $\mu(x)$ будем писать $\mu(p(x))$.

- Установим характер зависимости меры от вероятности сообщений. Пусть имеется множество из двух равновероятных сообщений и стоит задача кодирования сообщений этого множества двоичными символами алфавита $A = \{0, 1\}$. Интуитивно ясно, что неплохой способ кодирования состоит в том, чтобы сопоставить одному из

сообщений символ 0, другому – символ 1. Точно также для кодирования 4 равновероятных сообщений можно использовать последовательности длины 2, для 8 сообщений – длины 3 и т.д. В этих примерах вероятностям 1/2, 1/4, 1/8,... соответствовали затраты на передачу равные соответственно 1, 2, 3,... единицам информационной меры. Следовательно, разумно потребовать, чтобы мера информации удовлетворяла соотношению $\mu(p^m) = m\mu(p)$.

- Рассмотрим последовательность сообщений $x_1, ..., x_n$, выбираемых независимо из одного и того же множества X с одним и тем же распределением вероятностей $\{p(x)\}$. Из статистической независимости сообщений следует, что знание предыдущих сообщений не помогает предугадать, какими будут следующие. Поэтому затраты на передачу последовательности складываются из затрат на передачу каждого отдельного сообщения. Получаем еще одно разумное требование к информационной мере:

$$\mu(x_1, ..., x_n) = \mu(x_1) + ... + \mu(x_n)$$

для независимых сообщений $x_1, ..., x_n$.

Перечисленные требования к информационной мере приводят нас к следующему определению.

Собственной информацией $I(x)$ сообщения x, выбираемого из дискретного ансамбля $X = \{x, p(x)\}$, называется величина, вычисляемая по формуле

$$I(x) = -\log p(x). \tag{1.1}$$

В этой формуле не указано основание логарифма. Мы и дальше не будем его указывать, всякий раз подразумевая, что логарифмы вычисляются по основанию 2, если не оговорено другое. Это соответствует измерению информации в *битах*.

Из определения собственной информации и свойств логарифма непосредственно вытекают следующие свойства собственной информации.

- *Неотрицательность*: $I(x) \geq 0, x \in X$.
- *Монотонность*: если $x_1, x_2 \in X$, $p(x_1) \geq p(x_2)$, то $I(x_1) \leq I(x_2)$
- *Аддитивность*. Для независимых сообщений $x_1, ..., x_n$ имеет место равенство

$$I(x_1, ..., x_n) = \sum_{i=1}^{n} I(x_i).$$

Пример 1.1.1 Пусть $X = \{a, b, c, d\}$ и вероятности букв равны $p(a) = 1/2$, $p(b) = 1/4$, $p(c) = p(d) = 1/8$. Тогда собственные информацией букв равны $I(a) = 1$, $I(b) = 2$, $I(c) = I(d) = 3$ бит. Легко указать способ кодирования, при котором на каждую букву будет затрачено по два бита. Наша мера информации показывает, что на самом деле буквы несут различное количество информации, и, может быть, разумнее тратить различное число бит на передачу различных букв (как в азбуке Морзе). Позже мы вернемся к этому примеру и проверим наше предположение.

Пример 1.1.2 Пусть $X = \{a, b\}$ и вероятности букв равны $p(a) = 0.05$, $p(b) = 0.95$. Тогда $I(a) = 4,322$, $I(b) = 0,216$. Мы получили дробные числа. Более того, передача одного из двух символов может потребовать больше 4 бит информации! На второй символ предлагается тратить примерно $1/5$ бита. Это, на первый взгляд, кажется невозможным. Однако, как мы увидим позже, тратить целый бит на передачу каждой буквы рассматриваемого источника – непозволительная роскошь.

1.2 Энтропия

Определенная в предыдущем параграфе мера информации, содержащейся в сообщении, представляет собой случайную величину. Собственная информация сообщения x дискретного ансамбля $X = \{x, p(x)\}$ характеризует "информативность" или "степень неожиданности" конкретного сообщения. Естественно, среднее значение или математическое ожидание этой величины по ансамблю $X = \{x, p(x)\}$ будет характеристикой информативности всего ансамбля.

Энтропией дискретного ансамбля $X = \{x, p(x)\}$ называется величина

$$H(X) = \mathbf{M}\left[-\log p(x)\right] = -\sum_{x \in X} p(x) \log p(x) \quad .$$

Можно интерпретировать энтропию как количественную меру априорной неосведомленности о том, какое из сообщений будет порождено источником. Часто говорят, что энтропия является мерой неопределенности. Приведем несколько свойств энтропии. Эти свойства проясняют смысл понятия энтропии и позволяют оценивать ее, не выполняя точных вычислений.

Свойство 1.2.1 $H(X) \geq 0$.

Свойство 1.2.2 $H(X) \leq \log |X|$. *Равенство имеет место в том и только в том случае, когда элементы ансамбля X равновероятны.*

Свойство 1.2.3 *Если для двух ансамблей X и Y распределения вероятностей представляют собой одинаковые наборы чисел (отличающиеся только порядком следования элементов), то $H(X) = H(Y)$.*

Свойство 1.2.4 *Если ансамбли X и Y независимы, то*

$$H(XY) = H(X) + H(Y).$$

Свойство 1.2.5 *Энтропия – выпуклая $\cap$ функция распределения вероятностей на элементах ансамбля X.*

Свойство 1.2.6 *Пусть $X = \{x, p(x)\}$ и $A \subseteq X$. Введем ансамбль $X' = \{x, p'(x)\}$, задав распределение вероятностей $p'(x)$ следующим образом*

$$p'(x) = \begin{cases} \frac{P(A)}{|A|}, x \in A, \\ p(x), x \notin A. \end{cases}$$

Тогда $H(X') \geq H(X)$. Иными словами, "выравнивание" вероятностей элементов ансамбля приводит к увеличению энтропии.

Свойство 1.2.7 *Пусть задан ансамбль X и на множестве его элементов определена функция $g(x)$. Введем ансамбль $Y = \{y = g(x)\}$. Тогда $H(Y) \leq H(X)$. Равенство имеет место тогда и только тогда, когда функция $g(x)$ обратима.*

Смысл последнего утверждения состоит в том, что обработка информации не приводит к увеличению энтропии.

Доказательства свойств 1.2.1, 1.2.3 и 1.2.4 мы опускаем как очень простое упражнение для читателя. Доказательства свойств 1.2.5 и 1.2.6 будут даны в следующем параграфе. Доказательство свойства 1.2.7 отложим до знакомства с понятием условной энтропии (параграф 1.4).

Доказательство свойства 1.2.2

Рассмотрим разность левой и правой частей доказываемого неравенства:

$$\begin{aligned} H(X) - \log|X| &\overset{(a)}{=} -\sum_{x\in X} p(x)\log p(x) - \sum_{x\in X} p(x)\log|X| = \\ &\overset{(b)}{=} \sum_{x\in X} p(x)\log\frac{1}{p(x)|X|} \leq \\ &\overset{(c)}{\leq} \log e\left[\sum_{x\in X} p(x)\left(\frac{1}{p(x)|X|} - 1\right)\right] = \\ &= \log e\left(\sum_{x\in X}\frac{1}{|X|} - \sum_{x\in X} p(x)\right) = 0 \quad . \end{aligned}$$

Поясним приведенные выкладки. Переход (a) основан на свойстве нормировки вероятностей, (b) – на свойствах логарифма. Переход (c) использует хорошо известное неравенство

$$\ln x \leq x - 1,$$

эквивалентное неравенству

$$\log x \leq (x-1)\log e. \tag{1.2}$$

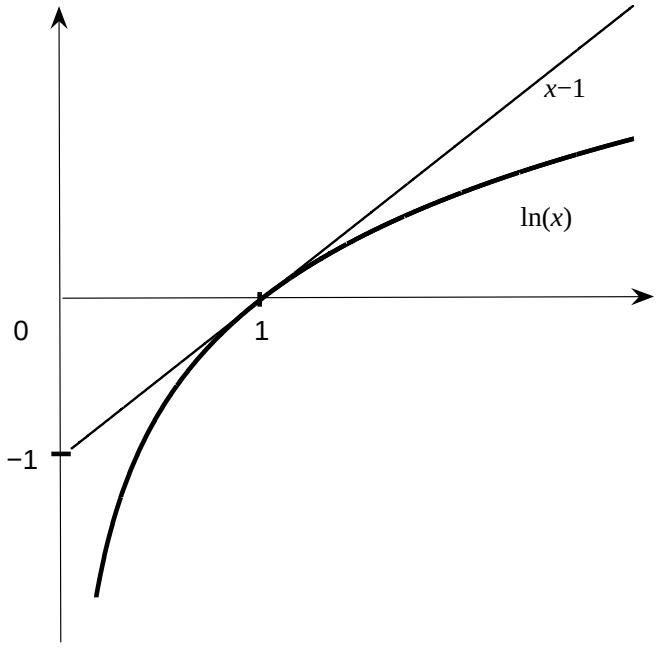


Рис. 1.1: Графическая интерпретация неравенства $\ln(x) \leq x - 1$

Графики функций $\ln x$ и $(x-1)$ приведены на рис.1.1. Из рисунка видно, что неравенство превращается в равенство только в одной точке $x = 1$. Поэтому в нашем случае необходимым и достаточным условием равенства будет условие $p(x) = 1/|X|$, $x \in X$. Последующие переходы снова используют условие нормировки вероятностей. Итак, разность оказалась неотрицательной и, следовательно, свойство 1.2.2 доказано. □

Пример 1.2.1 Рассмотрим двоичный ансамбль $X = \{0, 1\}$. Положим $p(1) = p$, $p(0) = 1 - p = q$. Энтропия этого ансамбля равна

$$H(X) = -p\log p - q\log q \triangleq \eta(p). \tag{1.3}$$

Мы ввели специальное обозначение $\eta(p)$ для энтропии двоичного ансамбля, в котором один из элементов имеет вероятность p. Эта функция будет часто использоваться. Построим график зависимости $\eta(p)$ от p при $p \in [0,1]$. Начнем с крайних точек $p = 0$ и $p = 1$. В обоих случаях имеет место неопределенность. Тем не менее, используя правило Лопиталя, легко вычислить предельные значения $\eta(p)$ при $p \to 0$ и $p \to 1$. Получим $\eta(0) = \eta(1) = 0$. Далее, заметим, что $\eta(p) = \eta(1-p)$, то есть функция симметрична относительно оси $p = 1/2$. Это можно было бы также утверждать, опираясь на свойство энтропии 1.2.3. Чтобы найти экстремальные точки, следует вычислить производную функции $\eta(p)$. Производная равна

$$\eta'(p) = -\log p + \log(1-p).$$

Точкой экстремума оказывается точка $p = 1/2$. Вычисляя вторую производную

$$\eta''(p) = -\log e/p - \log e/(1-p) < 0,$$

убеждаемся в том, что это точка максимума. Этот результат можно было предсказать на основе свойства 1.2.2. Кроме того, нетрудно видеть, что вторая производная строго отрицательна. Отсюда следует, что энтропия двоичного ансамбля – строго выпуклая вверх функция параметра p. График функции $\eta(p)$ представлен на рис.1.2.

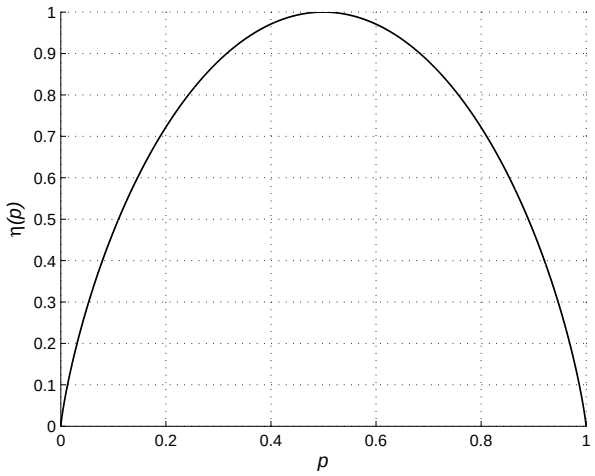


Рис. 1.2: Энтропия двоичного ансамбля

1.3 Выпуклые функции многих переменных

В математической литературе на русском языке различают выпуклые и вогнутые функции. В теории информации принято различать функции выпуклые вверх и функции выпуклые вниз. В первом случае для краткости пишут, что функция выпукла $\cap$, а во втором – что она выпукла $\cup$.

Множество аргументов функции запишем в виде вектора. Рассмотрим произвольную функцию $f(\boldsymbol{x})$, определенную для всех векторов из некоторого множества $R = \{\boldsymbol{x}\}$. Само множество R мы считаем подмножеством m-мерного евклидова пространства векторов с вещественными компонентами.

Множество вещественных векторов R *в*ыпукло, если для любых $\boldsymbol{x}, \boldsymbol{x}' \in R$ и любого $\alpha \in [0,1]$ вектор $\boldsymbol{y} = \alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{x}'$ принадлежит R.

Приведем важный для нас содержательный пример выпуклой области.

Теорема 1.1 *Множество вероятностных векторов длины M выпукло.*

Доказательство. Напомним, что компоненты вероятностных векторов неотрицательны и сумма компонент равна 1. Будем записывать распределение вероятностей на множестве $X = \{1, 2, \ldots, M\}$ в виде вектора из M элементов. Для двух распределений вероятностей $\boldsymbol{p} = (p_1, ..., p_M)$ и $\boldsymbol{p}' = (p'_1, ..., p'_M)$ и параметра $\alpha \in [0,1]$ рассмотрим множество векторов вида

$$\boldsymbol{q} = \alpha\boldsymbol{p} + (1-\alpha)\boldsymbol{p}'.$$

Нужно доказать, что этот вектор тоже стохастический. Все элементы вектора $\boldsymbol{q} = (q_1, ..., q_M)$ неотрицательны, поскольку в правой части имеем сумму двух неотрицательных векторов. Сумма компонент вектора $\boldsymbol{q}$ равна

$$\sum_{i=1}^{M} q_i = \alpha \sum_{i=1}^{M} p_i + (1-\alpha) \sum_{i=1}^{M} p'_i = \alpha + 1 - \alpha = 1.$$

Тем самым утверждение доказано. □

Теперь мы готовы к тому, чтобы конкретизировать понятие выпуклой функции. Функция $f(\boldsymbol{x})$ векторного аргумента $\boldsymbol{x}$, определенная на выпуклой области R, называется *выпуклой* $\cap$ на этой области, если для любых $\boldsymbol{x}, \boldsymbol{x}' \in R$ и любого $\alpha \in [0,1]$ имеет место неравенство

$$f(\alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{x}') \geq \alpha f(\boldsymbol{x}) + (1-\alpha) f(\boldsymbol{x}') \quad . \tag{1.4}$$

Отметим, что если в (1.4) возможно равенство при некоторых значениях $\alpha \in (0,1)$, функцию называют *нестрого выпуклой*. В противном случае она называется *строго выпуклой*. Выпуклость области определения функции нужна для того, чтобы выражение в левой части (1.4) имело смысл.

Понятно, что определение функции выпуклой $\cup$ получается из (1.4) изменением знака неравенства на противоположный. Мы в дальнейшем рассматриваем только выпуклые $\cap$ функции, считая, что перенос результатов на второй случай не составляет труда.

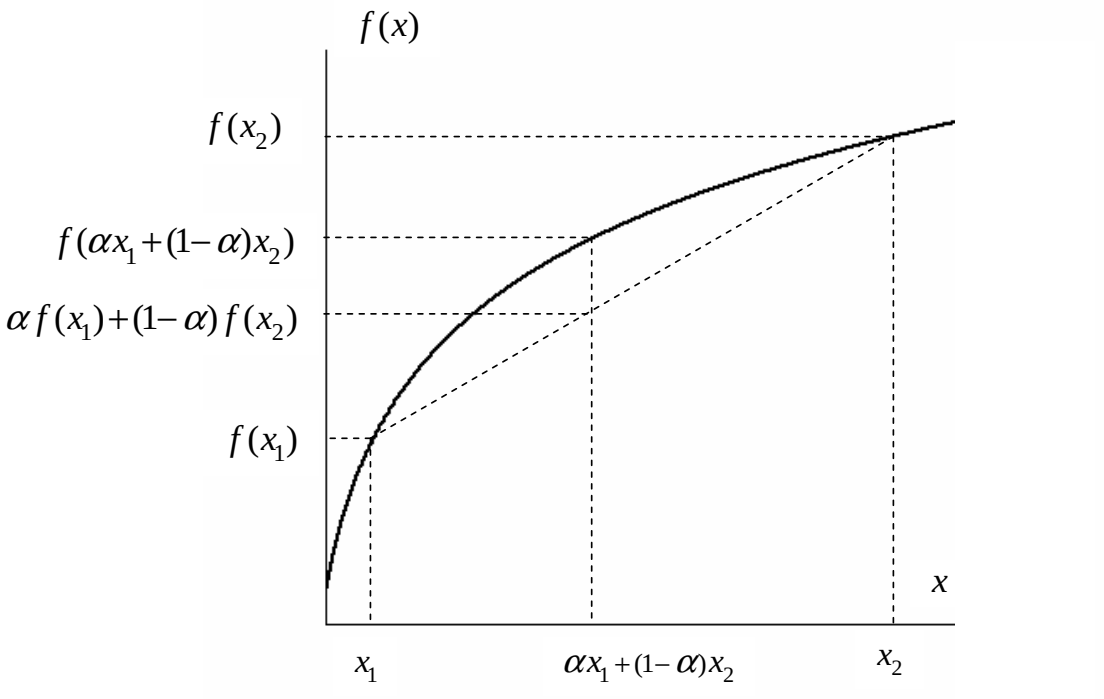


Рис. 1.3: Определение выпуклой функции

Рис.1.3 поясняет понятие выпуклой функции многих переменных на примере функции одной переменной. Из элементарных геометрических соображений следует, что правой части (1.4) при различных значениях α соответствуют точки отрезка, соединяющего точки с координатами $(x_1, f(x_1))$ и $(x_2, f(x_2))$. Левой части соответствуют точки на кривой. Из рисунка видно, что

$$f(\alpha x_1 + (1-\alpha)x_2) \geq \alpha f(x_1) + (1-\alpha) f(x_2),$$

и определение выпуклости функции многих переменных хорошо согласуется с привычным понятием выпуклости функции одной переменной.

Непосредственно из определения выпуклой функции, используя метод математической индукции, легко вывести следующее свойство выпуклых функций.

Теорема 1.2 *Пусть* $f(\boldsymbol{x})$ *– выпуклая* $\cap$ *функция векторного аргумента* $\boldsymbol{x}$*, определенная на выпуклой области* R *и пусть константы* $\alpha_1, ..., \alpha_M \in [0,1]$ *таковы, что* $\sum_{m=1}^{M} \alpha_m = 1$*. Тогда для любых* $\boldsymbol{x}_1, ..., \boldsymbol{x}_M \in R$

$$f\left(\sum_{m=1}^{M} \alpha_m \boldsymbol{x}_m\right) \geq \sum_{m=1}^{M} \alpha_m f(\boldsymbol{x}_m). \tag{1.5}$$

Это свойство имеет следующую важную для дальнейшего применения интерпретацию. Константы α_m, использованные в формулировке этого свойства, можно интерпретировать как вероятности соответствующих векторов $\boldsymbol{x}_m$. Тогда суммы в правой и левой части (1.5) можно интерпретировать как математические ожидания, если считать, что случайный вектор $\boldsymbol{x}$ принимает значение $\boldsymbol{x}_m$ с вероятностью α_m. Получаем полезное неравенство

$$f\left(\mathbf{M}\left[\boldsymbol{x}\right]\right) \geq \mathbf{M}\left[f(\boldsymbol{x})\right]. \tag{1.6}$$

Именно это простое неравенство, часто называемое *неравенством Йенсена*, в дальнейшем сократит выкладки, связанные с анализом информационных характеристик алгоритмов кодирования. Однако, чтобы пользоваться им, нужно быть уверенным, что функция выпукла. Поэтому остаток параграфа посвятим выводу признаков выпуклости функций многих переменных.

В формулировках следующих утверждений, вытекающих из определения выпуклых функций, предполагается, что все функции определены на одной и той же выпуклой области.

Свойство 1.3.1 *Сумма выпуклых функций выпукла.*

Свойство 1.3.2 *Произведение выпуклой функции и положительной константы представляет собой выпуклую функцию.*

Прямым следствием этих утверждений является еще одно свойство.

Свойство 1.3.3 *Линейная комбинация выпуклых функций с неотрицательными коэффициентами – выпуклая функция.*

Теперь мы знаем, что доказать выпуклость функции многих переменных можно, представив функцию в виде линейной комбинации других функций, выпуклость которых доказать легко. Если, например, функция

представлена в виде линейной комбинации функций одной переменной, то анализ составляющих можно выполнить с помощью вычисления производных. Опираясь на эти результаты, докажем выпуклость энтропии ансамбля как функции распределения вероятностей на множестве его элементов.

Теорема 1.3 *Энтропия $H(\boldsymbol{p})$ дискретного ансамбля с распределением вероятностей $\boldsymbol{p}$ является выпуклой $\cap$ функцией аргумента $\boldsymbol{p}$.*

Доказательство. Прежде всего, заметим, что постановка вопроса о выпуклости энтропии корректна, поскольку ее область определения выпукла в соответствии с Теоремой 1.1, доказанной выше. По определению энтропии

$$H(\boldsymbol{p}) = -\sum_{m=1}^{M} p_m \log p_m = \sum_{m=1}^{M} f_m(\boldsymbol{p}). \tag{1.7}$$

Рассмотрим слагаемые $f_m(\boldsymbol{p})$. Каждое из них представляет собой функцию одной переменной. Вторая производная этой функции равна $-(\log e)/p_m$ и, следовательно, отрицательна при всех $p_m \in (0,1)$. Таким образом, энтропия выпукла в силу свойства 1.3.3. Тем самым теорема доказана. □

Заметим, что слагаемые $f_m(\boldsymbol{p})$ в (1.7) – *строго* выпуклые функции. Следовательно, энтропия также *строго* выпукла.

В заключение параграфа мы приведем пример применения свойств выпуклых функций как инструмента исследования свойств информационных мер. Докажем свойство энтропии 1.2.6.

Доказательство свойства 1.2.6. Пусть имеется ансамбль X, $|X| = M$. Запишем вероятности букв в виде вектора $\boldsymbol{p} = (p_1, ..., p_M)$. Для энтропии ансамбля будем использовать обозначение $H(\boldsymbol{p})$. Докажем, частный случай утверждения, а именно, докажем, что выравнивание вероятностей первого и второго элементов множества X приводит к увеличению энтропии. Обобщение на случай произвольного подмножества $A \subseteq X$ оставим читателю в качестве упражнения. Обозначим $\tilde{\boldsymbol{p}} = ((p_1 + p_2)/2, (p_1 + p_2)/2, p_3, ..., p_M)$. Нужно доказать неравенство

$$H(\tilde{\boldsymbol{p}}) \geq H(\boldsymbol{p}). \tag{1.8}$$

Для этого введем обозначения

$$\begin{aligned} \boldsymbol{p}' = \boldsymbol{p} &= (p_1, p_2, p_3, ..., p_M), \\ \boldsymbol{p}'' &= (p_2, p_1, p_3, ..., p_M). \end{aligned}$$

Заметим, что по свойству энтропии 1.2.3 два распределения имеют одинаковую энтропию, т.е. $H(\boldsymbol{p}') = H(\boldsymbol{p}'') = H(\boldsymbol{p})$. Справедливо тождество $\tilde{\boldsymbol{p}} = (\boldsymbol{p}' + \boldsymbol{p}'')/2$. Из выпуклости энтропии следует, что

$$\begin{aligned} H(\tilde{\boldsymbol{p}}) &= H\left(\frac{\boldsymbol{p}' + \boldsymbol{p}''}{2}\right) \geq \\ &\geq \frac{1}{2}H(\boldsymbol{p}') + \frac{1}{2}H(\boldsymbol{p}'') = H(\boldsymbol{p}). \end{aligned}$$

□

1.4 Условная энтропия

Как уже отмечалось, для эффективного кодирования информации необходимо учитывать статистическую зависимость сообщений. Наша ближайшая цель – научиться подсчитывать информационные характеристики последовательностей зависимых сообщений. Начнем с последовательности из двух сообщений.

Рассмотрим ансамбли $X = \{x\}$ и $Y = \{y\}$ и их произведение $XY = \{(x, y), p(x, y)\}$. Для любого фиксированного $y \in Y$ можно построить условное распределение вероятностей $p(x|y)$ на множестве X и для каждого $x \in X$ подсчитать собственную информацию

$$I(x|y) = -\log p(x|y),$$

которую называют *условной собственной информацией* сообщения x при фиксированном y.

Ранее мы назвали энтропией ансамбля X среднюю информацию сообщений $x \in X$. Аналогично, усреднив условную информацию $I(x|y)$ по $x \in X$, получим величину

$$H(X|y) = -\sum_{x \in X} p(x|y) \log p(x|y), \tag{1.9}$$

называемую *условной энтропией X при фиксированном $y \in Y$*. Заметим, что в определении (1.9) имеет место неопределенность в случае, когда $p(x|y) = 0$. В параграфе 1.3 отмечалось, что выражение вида $z \log z$ стремится к нулю при $z \to 0$ и на этом основании мы считали равными нулю слагаемые энтропии, соответствующие буквам x вероятность которых $p(x) = 0$. Точно также в (1.9) мы считаем равными нулю слагаемые, для которых $p(x|y) = 0$.

Вновь введенная энтропия $H(X|y)$ – случайная величина, поскольку она зависит от случайной переменной y. Чтобы получить неслучайную информационную характеристику пары вероятностных ансамблей, нужно выполнить усреднение в (1.9) по всем значениям y. Величина

$$H(X|Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(x|y)$$

называется *условной энтропией ансамбля X при фиксированном ансамбле Y*. Отметим ряд свойств условной энтропии.

Свойство 1.4.1

$$H(X|Y) \geq 0.$$

Свойство 1.4.2

$$H(X|Y) \leq H(X),$$

причем равенство имеет место в том и только в том случае, когда ансамбли X и Y независимы.

Свойство 1.4.3

$$H(XY) = H(X) + H(Y|X) = H(Y) + H(X|Y).$$

Свойство 1.4.4

$$\begin{aligned} H(X_1...X_n) = H(X_1) &+ H(X_2|X_1) + \\ &+ H(X_3|X_1X_2) + ... + \\ &+ H(X_n|X_1,...,X_{n-1}). \end{aligned}$$

Свойство 1.4.5

$$H(X|YZ) \leq H(X|Y)$$

причем равенство имеет место в том и только в том случае, когда ансамбли X и Z условно независимы при всех $y \in Y$.

Свойство 1.4.6

$$H(X_1...X_n) \leq \sum_{i=1}^{n} H(X_i),$$

причем равенство возможно только в случае совместной независимости ансамблей X_1,. . . ,X_n.

Свойство 1.4.1 доказывается так же как свойство энтропии 1.2.1.

Доказательство свойства 1.4.2. По аналогии с доказательством свойства энтропии 1.2.2:

$$\begin{aligned} H(X|Y) - H(X) &= -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(x|y) + \\ &\quad + \sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(x) = \\ &= \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\frac{p(x)}{p(x|y)} \leq \\ &\leq \sum_{x\in X}\sum_{y\in Y} p(x,y)\left(\frac{p(x)}{p(x|y)} - 1\right)\log e = \\ &= \left(\sum_{x\in X}\sum_{y\in Y} p(y)p(x) - \sum_{x\in X}\sum_{y\in Y} p(x,y)\right)\log e = \\ &= 0. \end{aligned}$$

□

Доказательство свойств 1.4.3 и 1.4.4. По формуле умножения вероятностей

$$p(x,y) = p(x)p(y|x) = p(y)p(x|y),$$

$$p(x_1,...,x_n) = p(x_1)p(x_2|x_1)...p(x_n|x_1,...,x_{n-1}).$$

Обе стороны этих тождеств нужно прологарифмировать, а затем усреднить по всем случайным переменным. □

Доказательство свойства 1.4.5. Рассмотрим ансамбль $XYZ = \{(x,y,z), p(x,y,z)\}$. При каждом $y \in Y$ определены условные распределения $p(x,z|y)$ и $p(x|y)$. Для этих распределений запишем энтропии

$$H(X|y,Z) = \mathbf{M}_{XZ|y}[-\log p(x|yz)],$$

$$H(X|y) = \mathbf{M}_{X|y}[-\log p(x|y)].$$

По свойству 1.5.2

$$H(X|y,Z) \leq H(X|y).$$

После усреднения по всем y получим требуемое неравенство. □

Из свойств 1.4.1–1.4.5 следует 1.4.6.

Применим выявленные свойства условной энтропии для доказательства свойства энтропии 1.2.7, устанавливающего невозрастание энтропии при обработке информации.

Доказательство свойства 1.2.7. Имеем ансамбль $X = \{x, p(x)\}$, определенную на нем функцию $g(x)$ и ансамбль $Y = \{y = g(x), x \in X\}$. Нужно доказать, что

$$H(Y) \leq H(X). \tag{1.10}$$

По свойствам энтропии

$$H(XY) = \underbrace{H(X|Y)}_{\geq 0} + H(Y) = \underbrace{H(Y|X)}_{=0} + H(X). \tag{1.11}$$

Поскольку значения $g(x)$ однозначно определены при заданном x, имеем $H(Y|X) = 0$. В то же время $H(X|Y) \geq 0$. С учетом этих обстоятельств из (1.11) следует (1.10). □

1.5 Энтропия на сообщение дискретного стационарного источника

Рассмотрим произвольный дискретный стационарный источник, порождающий последовательность $(x_1, x_2, \ldots, x_t, \ldots)$, $x_t \in X_t = X$. Из предположения о стационарности следует, что распределение вероятностей для буквы x_t, порождаемой в момент времени t, не зависит от t. Следовательно, величина энтропии этого распределения $H(X_t) = H(X)$ также не зависит от времени. Назовем ее *одномерной энтропией источника* (или соответствующего случайного процесса). Обозначим ее как $H_1(X)$.

Как уже было отмечено, величина $H_1(X)$ не определяет полностью информационные характеристики процесса, поскольку не учитывает зависимости букв.

Рассмотрим последовательность из n последовательных букв источника $\boldsymbol{x} = (x_1, ..., x_n) \in X_1X_2...X_n = X^n$. Для стационарного процесса энтропия распределения вероятностей на таких блоках $H(X_1...X_n) = H(X^n)$ не зависит от расположения блока во времени, ее называют *n-мерной энтропией* источника.

Величина $H(X^n)$ определяет среднее количество информации в последовательности из n букв. Нормированную величину

$$H_n(X) = \frac{H(X^n)}{n},$$

называют *энтропией на букву последовательности длины n*. Интуиция подсказывает, что значения $H_n(X)$ при больших n могли бы служить адекватной мерой информативности источника.

Другой подход к измерению информации, порождаемой произвольным стационарным источником, состоит в том, что при передаче буквы x_n все предыдущие буквы $x_1, ..., x_{n-1}$ можно считать известными декодеру. Среднее количество подлежащей передаче информации об x_n определяется величиной условной энтропии $H(X_n|X_1, ..., X_{n-1})$. В силу стационарности конкретные значения индексов не играют роли, важна лишь длина предыстории. Поэтому используется обозначение

$$H(X_n|X_1, \ldots, X_{n-1}) = H(X|X^{n-1}).$$

Следующая теорема устанавливает некоторые свойства двух информационных мер $H_n(X)$ и $H(X|X^{n-1})$ для стационарных источников.

Теорема 1.4 *Для дискретного стационарного источника*
A. $H(X|X^n)$ *не возрастает с увеличением* n*;*
B. $H_n(X)$ *не возрастает с увеличением* n*;*
C. $H_n(X) \geq H(X/X^{n-1})$*;*
D. $\lim\limits_{n\to\infty} H_n(X) = \lim\limits_{n\to\infty} H(X|X^n)$.

Доказательство. Утверждение A следует из невозрастания энтропии с увеличением числа условий (свойство 1.4.5).

Из свойства 1.4.4 и стационарности источника имеем

$$H(X^n) = H(X) + H(X|X^1) + ... + H(X|X^{n-1}).$$

Заметим, что в правой части n слагаемых, из которых последнее – наименьшее (в силу свойств 1.4.2 и 1.4.5). Поделив обе части на n, убеждаемся в справедливости утверждения C.

Чтобы убедиться в справедливости утверждения B, выполним преобразования

$$\begin{aligned}
H(X^{n+1}) &\overset{(a)}{=} H(X_1...X_nX_{n+1}) = \\
&\overset{(b)}{=} H(X_1...X_n) + H(X_{n+1}|X_1, ..., X_n) = \\
&\overset{(c)}{=} H(X^n) + H(X|X^n) \leq \\
&\overset{(d)}{\leq} H(X^n) + H(X|X^{n-1}) \leq \\
&\overset{(e)}{\leq} H(X^n) + H_n(X) = \\
&\overset{(f)}{=} (n+1)H_n(X).
\end{aligned}$$

В этой цепочке переход (a) использует предположение о стационарности, (b) – свойство условной энтропии, (c) вновь использует стационарность, а (d) и (e) опираются на уже доказанные утверждения A и C. Наконец, (f) следует из определения величины $H_n(X)$. Если теперь поделить правую и левую часть на $(n+1)$, убеждаемся в справедливости утверждения B.

Перейдем к доказательству последнего утверждения. Прежде всего отметим, что последовательности $H_n(X)$ и $H(X|X^n)$ не возрастают и ограничены снизу (поскольку неотрицательны). Значит, оба предела существуют. Из свойства C следует, что

$$\lim_{n\to\infty} H_n(X) \geq \lim_{n\to\infty} H(X|X^n). \tag{1.12}$$

В то же время, при любых натуральных $m < n$ имеют место соотношения

$$\begin{aligned} H(X^n) &= H(X_1 \dots X_n) = \\ &\overset{(a)}{=} H(X_1 \dots X_m) + H(X_{m+1} \dots X_n | X_1, \dots, X_m) = \\ &\overset{(b)}{=} mH_m(X) + H(X_{m+1}|X_1, \dots, X_m) + \dots \\ &\quad + H(X_n|X_1, \dots, X_{n-1}) \leq \\ &\overset{(c)}{\leq} mH_m(X) + (n-m)H(X|X^m). \end{aligned}$$

Здесь использованы: в (a) и (b) – свойства условной энтропии, в (c) – стационарность источника и невозрастание последовательности $H(X|X^m)$ с увеличением числа условий m. Поделим обе части полученного неравенства на n и перейдем к пределу при $n \to \infty$. Получим неравенство

$$\lim_{n\to\infty} H_n(X) \leq H(X|X^m),$$

справедливое при любых m. Устремляя m к бесконечности, получаем

$$\lim_{n\to\infty} H_n(X) \leq \lim_{m\to\infty} H(X|X^m). \tag{1.13}$$

Из (1.12) и (1.13) вытекает доказываемое утверждение. □

Введем обозначения

$$H_\infty(X) = \lim_{n\to\infty} H_n(X), \quad H(X|X^\infty) = \lim_{n\to\infty} H(X|X^n).$$

Основной результат теоремы состоит в том, что $H_\infty(X) = H(X|X^\infty)$. В дальнейшем, изучая конструктивные методы кодирования, мы убедимся в том, что именно эта величина определяет минимально возможные

удельные затраты бит на передачу одной буквы стационарного источника.

Пример 1.5.1 *Энтропия на сообщение дискретного постоянного источника.* Дискретным постоянным источником мы назвали дискретный стационарный источник без памяти. По свойствам энтропии для источника без памяти имеем

$$H(X_1...X_n) = H(X_1) + ... + H(X_n).$$

$$H(X^n) = nH(X).$$

Поделив обе стороны тождества на n, получим, что при всех n справедливо равенство

$$H_n(X) = H(X),$$

и, следовательно,

$$H_\infty(X) = H(X).$$

Таким образом, энтропия на сообщение дискретного постоянного источника в точности равна его одномерной энтропии. К тому же результату можно было прийти с другой стороны. В силу стационарности и отсутствия зависимости между сообщениями

$$H(X|X^n) = H(X_{n+1}|X_1, ..., X_n) = H(X),$$

следовательно,

$$H(X|X^\infty) = H(X).$$

Таким образом, для рассматриваемой модели анализ информационных характеристик сводится к подсчету энтропии одномерного распределения. Отсюда совсем не следует, что при кодировании источников независимых сообщений нужно кодировать каждую букву независимо от других.

Пример 1.5.2 *Энтропия на сообщение марковского источника.* Для стационарного источника, описываемого моделью цепи Маркова связности s, можем записать

$$H(X|X^n) = H(X_{n+1}|X_1, ..., X_n) =$$

$$= H(X_{n+1}|X_{n-s+1}, ..., X_n) = H(X|X^s).$$

Правая часть не зависит от n, значит

$$H(X|X^\infty) = H(X|X^s).$$

Рассмотрим другой подход. Используя стационарность и свойства условной энтропии, запишем

$$\begin{aligned} H(X^n) &= H(X_1 \dots X_s X_{s+1} \dots X_n) = \\ &= H(X_1 \dots X_s) + H(X_{s+1} \dots X_n | X_1, \dots, X_s). \end{aligned} \quad (1.14)$$

Из

$$\begin{aligned} H(X_{s+1} \dots X_n | X_1, \dots, X_s) &= H(X_{s+1} | X_1, \dots, X_s) + \\ &+ H(X_{s+2} | X_1, \dots, X_{s+1}) + \dots \\ &+ H(X_n | X_1, \dots, X_{n-1}), \end{aligned}$$

учитывая марковость и стационарность, получаем

$$H(X_{s+1} ... X_n | X_1, ..., X_s) = (n-s) H(X|X^s).$$

Теперь (1.14) принимает вид

$$H(X^n) = s H_s(X) + (n-s) H(X|X^s). \quad (1.15)$$

Теперь можно поделить обе части тождества на n и устремить n к бесконечности. Результатом будет ожидаемое равенство

$$H_\infty(X) = H(X|X^s).$$

Поучительно записать (1.15) в виде

$$\begin{aligned} H_n(X) &= H(X|X^s) + \frac{s}{n}(H_s(X) - H(X|X^s)) = \\ &= H(X|X^n) + \frac{s}{n}(H_s(X) - H(X|X^s)). \end{aligned}$$

Отсюда, в частности, хорошо видно, какова разница между $H_n(X)$ и $H(X|X^n)$. При разбиении последовательности букв на блоки длины n мы получаем среднюю энтропию на сообщение немного большую, чем минимальная достижимая величина $H(X|X^s)$. "Потери" или, точнее сказать, дополнительные затраты связаны с "забвением" тех s букв, которые предшествовали рассматриваемому блоку.

1.6 Множество типичных последовательностей для дискретного постоянного источника. Источники с памятью

Рассмотрим дискретный постоянный источник, порождающий последовательность независимых сообщений из ансамбля $X = \{x, p(x)\}$. Множество типичных последовательностей длины n обозначим как $T_n(\delta)$. Оно

представляет собой множество последовательностей, средняя собственная информация которых близка к энтропии $H(X)$, то есть

$$T_n(\delta) = \left\{ \boldsymbol{x} : \left| \frac{1}{n} I(\boldsymbol{x}) - H(X) \right| \leq \delta \right\}, \tag{1.16}$$

где δ – некоторая положительная константа. Приведем в виде теоремы некоторые свойства множества $T_n(\delta)$.

Теорема 1.5 *Для любого $\delta > 0$ имеют место следующие утверждения:*

1. $$\lim_{n\to\infty} P\left(T_n(\delta)\right) = 1.$$

2. *Для любого натурального n справедливо неравенство*
$$|T_n(\delta)| \leq 2^{n(H(X)+\delta)}.$$

3. *Для любого $\varepsilon > 0$ существует n_0 такое, что при всех $n \geq n_0$*
$$|T_n(\delta)| \geq (1-\varepsilon) 2^{n(H(X)-\delta)}.$$

4. *Для $\boldsymbol{x} \in T_n(\delta)$*
$$2^{-n(H(X)+\delta)} \leq p(\boldsymbol{x}) \leq 2^{-n(H(X)-\delta)}. \tag{1.17}$$

Доказательство. Первое утверждение является прямым следствием неравенства Чебышева: для n независимых случайных величин $\xi_i, i = 1, ..., n$ с математическим ожиданием m и дисперсией σ^2 имеет место неравенство

$$P\left(\left| \frac{1}{n} \sum_{i=1}^{n} \xi_i - m \right| \geq \varepsilon \right) \leq \frac{\sigma^2}{n\varepsilon^2}. \tag{1.18}$$

Применив это неравенство к собственным информациям букв и устремив n к бесконечности, получаем нужный результат.

В силу (1.16) имеет место утверждение 4. Левая и правая части (1.17) могут рассматриваться как оценки снизу и сверху для вероятностей последовательностей из T. Для оценки числа элементов в этом множестве заметим, что

$$1 \geq P(T_n(\delta)) = \sum_{\boldsymbol{x}\in T_n(\delta)} p(\boldsymbol{x}) \geq |T_n(\delta)| \min_{\boldsymbol{x}\in T_n(\delta)} p(\boldsymbol{x}) \geq |T_n(\delta)| 2^{-n(H+\delta_0)}.$$

Отсюда вытекает второе утверждение теоремы. Осталось убедиться в справедливости третьего утверждения. Из первого утверждения следует, что для любого $\varepsilon > 0$ найдется n_0 такое, что при $n > n_0$ имеет место неравенство

$$P\left(T_n(\delta)\right) \geq 1 - \varepsilon. \tag{1.19}$$

Оценивая вероятность $T_n(\delta)$ как произведение числа элементов в множестве на величину максимального элемента и применяя утверждение 4, получаем

$$P\left(T_n(\delta)\right) \leq |T_n(\delta)| \max_{\boldsymbol{x} \in T_n(\delta)} p(\boldsymbol{x}) \leq |T_n(\delta)|\, 2^{-n(H(X)-\delta)}. \tag{1.20}$$

Из (1.19) и (1.20) следует утверждение 3. □

Из теоремы становится понятна связь энтропии с затратами на передачу информации. Из утверждения 1 следует, что достаточно кодировать последовательности из $T_n(\delta)$. Поскольку таких последовательностей примерно $2^{nH(X)}$ (утверждения 2 и 3), для передачи номера последовательности достаточно затратить $nH(X)$ бит на указание номера последовательности, т.е. $H(X)$ бит на одно сообщение.

Также просто объясняется справедливость того факта, что при затратах меньше $H(X)$ бит на одно сообщение надежная передача невозможна. Если взаимно однозначно кодируется M последовательностей, то суммарная вероятность однозначно кодируемых последовательностей из $T_n(\delta)$ имеет порядок $M2^{-nH(X)}$. Вероятность ошибки примерно равна $1 - M2^{-nH(X)}$. Пока число M меньше $2^{nH(X)}$ (соответственно, скорость кода меньше $H(X)$), произведение $M2^{-nH(X)}$ будет маленьким, а вероятность ошибки будет заведомо большой.

Посмотрим теперь внимательнее, какие именно последовательности являются типичными. Обозначим через $\tau_x(\boldsymbol{x})$ число появлений буквы x в последовательности $\boldsymbol{x}$. Набор чисел $\boldsymbol{\tau}(\boldsymbol{x}) = \{\tau_x(\boldsymbol{x}), x \in X\}$ называют *композицией последовательности* $\boldsymbol{x}$.

Для дискретного постоянного источника вероятность последовательности $\boldsymbol{x} = (x_1, ..., x_n)$ может быть записана в виде

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i) = \prod_{x \in X} p(x)^{\tau_x(\boldsymbol{x})}.$$

Это представление получено изменением порядка сомножителей в произведении. Собственная информация последовательности в расчете на

букву равна

$$\frac{1}{n}I(\boldsymbol{x}) = -\sum_{x\in X}\frac{\tau_x(\boldsymbol{x})}{n}\log p(x).$$

Эта величина близка к энтропии $H(X)$, если

$$\frac{\tau_x(\boldsymbol{x})}{n} \approx\ p(x)$$

Мы пришли к естественному результату: множество типичных последовательностей состоит из тех последовательностей, в которых доля элементов x приблизительно равна вероятности символа x.

Глава 2

Неравномерное кодирование дискретных источников

2.1 Постановка задачи неравномерного побуквенного кодирования

Предположим, что для некоторого дискретного источника X с известным распределением вероятностей $\{p(x), x \in X\}$ требуется построить эффективный неравномерный двоичный код над алфавитом $A = \{a\}$. Как и в предыдущем разделе, мы сосредоточим внимание на двоичных кодах, т.е. мы предполагаем, что $A = \{0, 1\}$.

Нам необходим такой двоичный код, который допускает однозначное разделение последовательности кодовых слов на отдельные кодовые слова без использования каких-либо дополнительных символов. Это требование называют свойством *однозначной декодируемости*.

Неравномерный побуквенный код $C = \{\boldsymbol{c}\}$ объема $|C| = M$ над алфавитом A определяется как произвольное множество последовательностей одинаковой или различной длины из букв алфавита A.

Код является *однозначно декодируемым*, если любая последовательность символов из A единственным способом разбивается на отдельные кодовые слова.

Пример 2.1.1 Для источника $X = \{0, 1, 2, 3\}$ среди четырех кодов

1. $C_1 = \{00, 01, 10, 11\}$;
2. $C_2 = \{1, 01, 001, 000\}$;

3. $C_3 = \{1, 10, 100, 000\}$;

4. $C_4 = \{0, 1, 10, 01\}$;

первые три кода однозначно декодируемы, последний код – нет.

Первый код этого примера – равномерный код. Понятно, что любой равномерный код может быть однозначно декодирован.

Для декодирования второго кода можно применить следующую стратегию. Декодер считывает символ за символом, и каждый раз проверяет, не совпадает ли полученная последовательность с одним из кодовых слов. В случае успеха соответствующее сообщение выдается получателю, и декодер приступает к декодированию следующего сообщения. В случае кода C_2 неоднозначности не может быть, поскольку ни одно слово не является продолжением другого.

Если ни одно кодовое слово не является началом другого, код называется *префиксным*. Префиксные коды являются однозначно декодируемыми.

Код C_3 заведомо не префиксный. Тем не менее, мы утверждаем, что он – однозначно декодируемый. Каждое слово кода C_3 получено переписыванием в обратном порядке соответствующего слова кода C_2. Для декодирования последовательности кодовых слов кода C_3 можно переписать принятую последовательность в обратном порядке и для декодирования использовать декодер кода C_2. Таким образом, мы приходим к следующему выводу.

Префиксность – достаточное, но не необходимое условие однозначной декодируемости.

Графически удобно представлять префиксные коды в виде кодовых деревьев. В частности, кодовое дерево кода C_2 примера 2.1.1 представлено на рис.2.1

Узлы дерева размещаются на ярусах. На начальном (нулевом) ярусе расположен один узел, называемый *корнем дерева.* Узлы следующих ярусов связаны с узлами предыдущих ярусов ребрами. В случае двоичного кода из каждого узла исходит не более двух ребер. Ребрам приписаны кодовые символы. В данном примере принято соглашение о том, что ребру, ведущему вверх, приписывается символ 0, а ребру, ведущему вниз – символ 1. Таким образом, каждой вершине дерева соответствует последовательность, считываемая вдоль пути, связывающего корень дерева с данным узлом.

Узел называется *концевым*, если из него не исходит ни одного ребра. Код называют *древовидным* , если в качестве кодовых слов он содержит

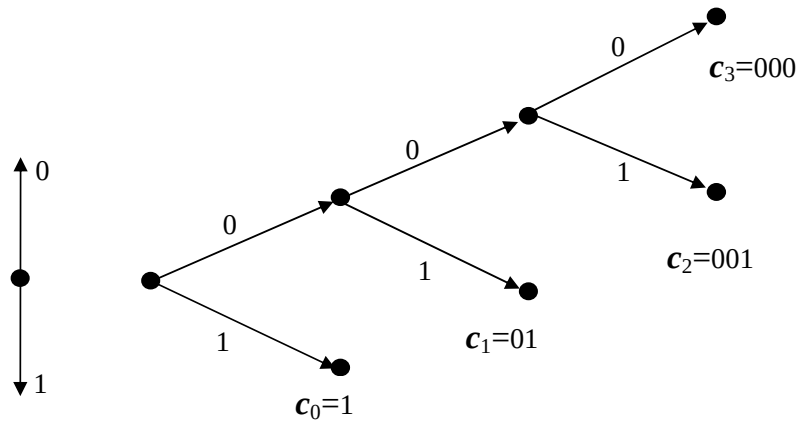


Рис. 2.1: Пример двоичного кодового дерева

только кодовые слова, соответствующие концевым вершинам кодового дерева.

Древовидность кода и префиксность – синонимы в том смысле, что всякий древовидный код является префиксным, и всякий префиксный код может быть представлен с помощью кодового дерева. Мы будем рассматривать только префиксные коды. В связи с этим возникает вопрос: не потеряли ли мы оптимальное решение задачи, сузив класс однозначно декодируемых кодов, среди которых мы ищем наилучшие? Ниже мы убедимся в том, что ответ на этот вопрос отрицательный.

Обсудим теперь, какие именно префиксные коды считать хорошими. Поскольку сама цель рассмотрения неравномерного кодирования состояла в уменьшении затрат на передачу сообщений, логично выбрать в качестве критерия качества кода среднюю длину кодовых слов. Рассмотрим источник $X = \{1, ..., M\}$, который порождает буквы с вероятностями $\{p_1, ..., p_M\}$. Предположим, что для кодирования букв источника выбран код $C = \{\boldsymbol{c}_1, ..., \boldsymbol{c}_M\}$ с длинами кодовых слов $l_1, \ldots, l_M$ соответственно.

Средней длиной кодовых слов называется величина

$$\bar{l} = \mathbf{M}[l_i] = \sum_{i=1}^{M} p_i l_i.$$

Еще один немаловажный аспект, который должен быть учтен при сравнении способов неравномерного кодирования, – это сложность реализации кодирования и декодирования. Подводя итог, мы формулируем задачу побуквенного неравномерного кодирования как задачу построения однозначно декодируемого кода с наименьшей средней длиной кодо-

вых слов при заданных ограничениях на сложность.

2.2 Неравенство Крафта

Требование префиксности накладывает жесткие ограничения на множество длин кодовых слов и не дает возможности выбирать кодовые слова слишком короткими. Формально эти ограничения записываются в виде изящного неравенства, называемого неравенством Крафта.

Теорема 2.1 *Необходимым и достаточным условием существования префиксного кода объема M с длинами кодовых слов $l_1, ..., l_M$ является выполнение неравенства Крафта*

$$\sum_{i=1}^{M} 2^{-l_i} \leq 1. \tag{2.1}$$

Доказательство. Начнем *с необходимости*. Мы должны убедиться в том, что неравенство (2.1) верно для любого префиксного кода.

Рассмотрим двоичное кодовое дерево произвольного префиксного кода объема M с длинами кодовых слов $l_1, ..., l_M$. Выберем целое число L, такое, что $L \geq \max_i l_i$. Продолжим все пути в дереве до яруса с номером L. На последнем ярусе мы получим 2^L вершин. Заметим, что концевая вершина исходного дерева, расположенная на глубине l_i, имеет 2 потомка на глубине l_i+1, 4 потомка на глубине l_i+2, и т.д. На глубине L будет 2^{L-l_i} потомков этой вершины. Множества потомков различных концевых вершин не пересекаются, поэтому суммарное число потомков не превышает общего числа вершин на ярусе L. Получаем неравенство

$$\sum_{i=1}^{M} 2^{L-l_i} \leq 2^L.$$

Поделив обе части на 2^L, получим требуемый результат. Рис. 2.2 иллюстрирует доказательство необходимости выполнения неравенства Крафта на примере кода из 4 слов. В этом примере $l_1 = 2$, $l_2 = l_3 = 3$, $l_4 = 1$, $L = 3$. Кодовое дерево показано сплошными линиями а ребра, появившиеся при продлении дерева до яруса с номером L показаны пунктиром.

Чтобы убедиться в *достаточности*, нужно показать, что из справедливости (2.1) вытекает существование кода с заданным набором длин кодовых слов. Построим такой код. Без потери общности можем считать числа l_i упорядоченными по возрастанию.

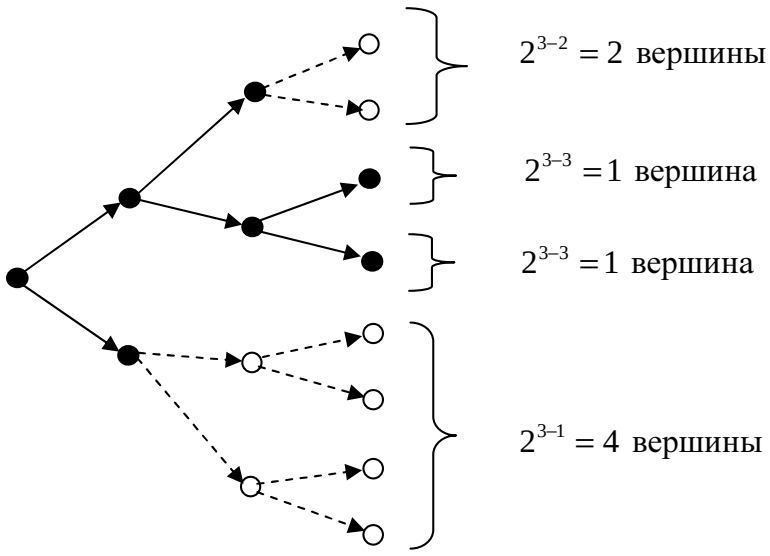


Рис. 2.2: Пояснение к доказательству необходимости неравенства Крафта для существования префиксного кода.

Из общего числа 2^{l_1} вершин на ярусе l_1 выберем одну любую, сделаем ее концевой и закрепим ее за первым кодовым словом. Продолжим оставшиеся вершины до яруса l_2. Из общего числа возможных вершин нужно исключить $2^{l_2-l_1}$ вершин, которые принадлежат поддереву, начинающемуся в узле, соответствующем первому слову. На ярусе l_2 останется

$$2^{l_2} - 2^{l_2-l_1} \geq 1$$

вершин. Последнее неравенство следует из (2.1), в чем нетрудно убедиться, поделив его правую и левую часть на 2^{l_2}. Сделаем одну из них концевой и закрепим ее за вторым словом. Аналогично, для третьего слова мы получим множество из

$$2^{l_3} - 2^{l_3-l_2} - 2^{l_3-l_1} \geq 1$$

вершин. В силу (2.1) всегда найдется одна для третьего слова. Продолжая построение, на последнем ярусе с номером l_M мы получим

$$2^{l_M} - 2^{l_M-l_{M-1}} - 2^{l_M-l_{M-2}} - \ldots - 2^{l_M-l_1}$$

вершин. Простые выкладки показывают, что это число не меньше 1, если неравенство (2.1) верно. Выбрав эту вершину для последнего слова, мы завершим построение префиксного кода. Рис. 2.3 иллюстрирует процесс построения кодового дерева для набора длин кодовых слов $l_1 = 1$, $l_2 = 2$, $l_3 = l_4 = 3$. □

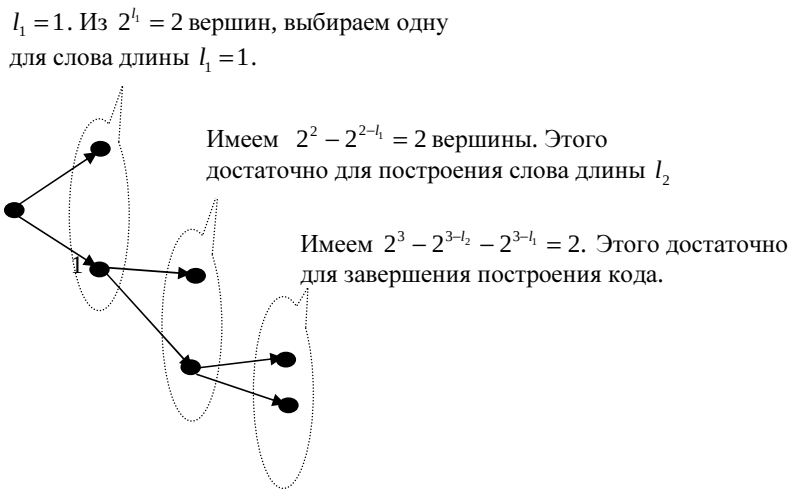


Рис. 2.3: Построение двоичного кодового дерева, удовлетворяющего неравенству Крафта.

В примере, показанном на рис. 2.3, неравенство Крафта выполняется с равенством. Внимательно просмотрев доказательство необходимости в Теореме 2.1, убеждаемся, что для достижения равенства в (2.1) кодовое дерево должно быть *полным*, т.е. каждая промежуточная вершина дерева должна иметь ровно 2 потомка и всем концевым вершинам должны быть сопоставлены кодовые слова.

Неравенство Крафта как бы ограничивает снизу длины кодовых слов префиксного кода заданного объема M. На этом будет впоследствии построено доказательство обратной теоремы кодирования. В связи с этим важно быть уверенным, что оно имеет место не только для древовидных (префиксных) кодов, но и для любых других однозначно декодируемых кодов. Это утверждение действительно имеет место и его доказательство можно найти, в частности, в [12].

2.3 Теоремы побуквенного неравномерного кодирования

2.3.1 Прямая теорема

Начнем непосредственно с формулировки теоремы кодирования.

Теорема 2.2 *Для ансамбля $X = \{x, p(x)\}$ с энтропией H существует побуквенный неравномерный префиксный код со средней длиной кодовых слов $\bar{l} < H + 1$.*

Доказательство. Рассмотрим источник над алфавитом $X = \{1, ..., M\}$ с вероятностями букв соответственно $p_1, ..., p_M$. Сопоставим букве x_m кодовое слово длины $l_m = \lceil -\log p_m \rceil$ при $m = 1, ..., M$. (Запись $\lfloor a \rfloor$ означают округление числа a вниз до ближайшего целого, а запись $\lceil a \rceil$ – округление числа a вверх до ближайшего целого). Префиксный код с таким набором длин кодовых слов в соответствии с Теоремой 2.1 существует, поскольку длины кодовых слов удовлетворяют неравенству Крафта

$$\sum_{m=1}^{M} 2^{-l_m} = \sum_{m=1}^{M} 2^{-\lceil -\log p_m \rceil} \leq \sum_{m=1}^{M} 2^{\log p_m} = \sum_{m=1}^{M} p_m = 1.$$

Средняя длина кодовых слов кода

$$\begin{aligned} \bar{l} &= \sum_{m=1}^{M} p_m l_m = \\ &= \sum_{m=1}^{M} p_m \lceil -\log p_m \rceil < \\ &< \sum_{m=1}^{M} p_m (-\log p_m + 1) = \\ &= H + \sum_{m=1}^{M} p_m = \\ &= H + 1, \end{aligned}$$

что и требовалось доказать. □

На примере двоичного источника в том, что теорема достаточно точна и ее результат не может быть улучшен, если не использовать никакой дополнительной информации об источнике. Действительно, предположим, что дан двоичный источник $X = \{0, 1\}$ с вероятностями букв $\{\varepsilon, 1-\varepsilon\}$. Минимально достижимая длина кодовых слов наилучшего кода, очевидно, равна 1. Теорема говорит, что средняя длина кодовых слов не больше $1 + \eta(\varepsilon)$, т.е. стремится к 1 при $\varepsilon \to 0$. Таким образом, для данного примера двоичного источника теорема точна.

2.3.2 Обратная теорема

Обратная теорема неравномерного кодирования устанавливает нижнюю границу средней длины кодовых слов любого однозначно декодируемого кода.

Теорема 2.3 *Для любого однозначно декодируемого кода дискретного источника $X = \{x, p(x)\}$ с энтропией H средняя длина кодовых слов $\bar{l}$ удовлетворяет неравенству*

$$\bar{l} \geq H. \tag{2.2}$$

Другими словами, не существует кода со средней длиной кодовых слов меньше H и обладающего свойством однозначной декодируемости.

Доказательство. Пусть $l(x)$ обозначает длину кодового слова для сообщения x. Имеем

$$H - \bar{l} = -\sum_{x \in X} p(x) \log p(x) - \sum_{x \in X} p(x) l(x) = \sum_{x \in X} p(x) \log \frac{2^{-l(x)}}{p(x)}.$$

Применяя уже знакомое нам неравенство для логарифма

$$\log x \leq (x - 1) \log e,$$

получаем

$$\begin{aligned} H - \bar{l} &\leq \log e \sum_{x \in X} p(x) \left(\frac{2^{-l(x)}}{p(x)} - 1 \right) = \\ &= \log e \left(\sum_{x \in X} 2^{-l(x)} - \sum_{x \in X} p(x) \right) \leq \\ &\leq \log e \left(1 - \sum_{x \in X} p(x) \right) = 0. \end{aligned} \tag{2.3}$$

Здесь мы использовали неравенство Крафта и условие нормировки вероятностей. Из (2.3) следует утверждение теоремы. □

Обсудим вопрос о том, при каких условиях возможно равенство в обратной теореме. Для этого равенство должно иметь место в первом и втором неравенствах в (2.3). Для этого при каждом x должно выполняться соотношение $p(x) = 2^{-l(x)}$. В то же время, для такого распределения вероятностей существует префиксный код с длинами кодовых слов $l(x) = \lceil -\log p(x) \rceil = -\log p(x)$. Для этого кода неравенство Крафта выполняется с равенством, и средняя длина кодового слова равна $\bar{l} = H$.

Таким образом, мы установили справедливость следующего утверждения.

Следствие 2.4 *Для существования кода со средней длиной кодовых слов $\bar{l} = H$ необходимо и достаточно чтобы все вероятности сообщений $x \in X$ имели вид $p(x) = 2^{-l(x)}$, где $\{l(x)\}$ – целые положительные числа.*

2.4 Оптимальный побуквенный код – код Хаффмена

Рассмотрим ансамбль сообщений $X = \{1, ..., M\}$ с вероятностями сообщений $\{p_1, ..., p_M\}$. Без потери общности мы считаем сообщения упорядоченными по убыванию вероятностей, т.е. $p_1 \leq p_2 \leq ... \leq p_M$. Наша задача состоит в построении оптимального кода, т.е. кода с наименьшей возможной средней длиной кодовых слов. Понятно, что при заданных вероятностях такой код может не быть единственным, возможно существование семейства оптимальных кодов. Мы установим некоторые свойства всех кодов этого семейства. Эти свойства подскажут нам простой путь к нахождению одного из оптимальных кодов.

Пусть двоичный код $C = \{\boldsymbol{c}_1, ..., \boldsymbol{c}_M\}$ с длинами кодовых слов $\{l_1, ..., l_M\}$ оптимален для рассматриваемого ансамбля сообщений.

Свойство 2.4.1 *Если $p_i < p_j$, то $l_i \geq l_j$.*

Доказательство. Свойство легко доказывается методом "от противного". Предположим, что $l_i < l_j$. Рассмотрим другой код C', в котором сообщению x_i соответствует слово $\boldsymbol{c}_j$, а сообщению x_j – слово $\boldsymbol{c}_i$. Нетрудно убедиться в том, что средняя длина кодовых слов для кода C' меньше, чем для кода C, что противоречит предположению об оптимальности кода C. □

Свойство 2.4.2 *Не менее двух кодовых слов имеют одинаковую длину $l_M = \max_m l_m$.*

Доказательство. Если предположить, что имеется только одно слово максимальной длины, то соответствующее кодовое дерево будет неполным. Очевидно, слово максимальной длины можно будет сделать короче по меньшей мере на 1 символ. При этом уменьшится средняя длина кодовых слов, что противоречит предположению об оптимальности кода. □

Свойство 2.4.3 *Среди кодовых слов длины $l_M = \max_m l_m$ найдутся два слова, отличающиеся только в одном последнем символе.*

Доказательство. Согласно предыдущему свойству, два слова длины l_M существуют в любом оптимальном коде. Рассмотрим концевой

узел, соответствующий одному из слов максимальной длины. Чтобы дерево было полным, должен существовать узел, имеющий общий предшествующий узел с данным узлом. Соответствующие двум концевым вершинам кодовые слова имеют одинаковую длину l_M и отличаются в одном последнем символе. □

Прежде чем сформулировать следующее свойство, введем дополнительные обозначения. Для рассматриваемого ансамбля $X = \{1, ..., M\}$ и некоторого кода C, удовлетворяющего свойствам 2.4.1 – 2.4.3, введем вспомогательный ансамбль $X' = \{1, ..., M-1\}$, сообщениям которого сопоставим вероятности $\{p'_1, ..., p'_{M-1}\}$ следующим образом

$$p'_1 = p_1, \quad \ldots, \quad p'_{M-2} = p_{M-2}, \quad p'_{M-1} = p_{M-1} + p_M.$$

Из кода C построим код C' для ансамбля X', приписав сообщениям $x'_1, ...x'_{M-2}$ те же кодовые слова, что и в коде C, т.е. $\boldsymbol{c}'_i = \boldsymbol{c}_i$, $i = 1, ..., M-2$, а сообщению x'_{M-1} – слово $\boldsymbol{c}'_{M-1}$, представляющее собой общую часть слов $\boldsymbol{c}_{M-1}$ и $\boldsymbol{c}_M$ (согласно свойству 2.4.3 эти два кодовых слова отличаются только в одном последнем символе).

Свойство 2.4.4 *Если код C' для X' оптимален, то код C оптимален для X.*

Доказательство. Длины кодовых слов кодов C и C' по построению связаны соотношениями

$$l_m = \begin{cases} l'_m \text{ при } m \le M-2, \\ l'_{M-1} + 1 \text{ при } m = M-1, M. \end{cases}$$

Отсюда

$$\begin{aligned} \bar{l} &= \sum_{m=1}^{M} p_m l_m = \\ &= \sum_{m=1}^{M-2} p_m l_m + p_{M-1} l_{M-1} + p_M l_M = \\ &= \sum_{m=1}^{M-2} p_m l_m + (p_{M-1} + p_M)(l'_{M-1} + 1) = \\ &= \sum_{m=1}^{M-2} p'_m l'_m + p'_{M-1} l'_{M-1} + p_{M-1} + p_M = \\ &= \sum_{m=1}^{M-1} p'_m l'_m + p_{M-1} + p_M = \bar{l}' + p_{M-1} + p_M. \end{aligned}$$

Input: Объем алфавита M, вероятности букв
Output: Двоичное дерево кода Хаффмена

`Инициализация:`
количество необработанных узлов $M_0 = M$
while $M_0 > 1$ **do**
В списке необработанных узлов найти два узла с наименьшими вероятностями.
Исключить эти узлы из списка необработанных.
Ввести новый узел, приписать ему суммарную вероятность двух исключенных узлов.
Новый узел связать ребрами с исключенными узлами.
$M_0 \leftarrow M_0 - 1$.
end

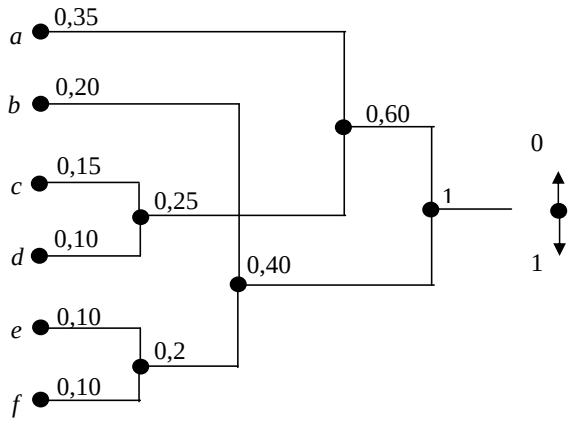
Рис. 2.4: Алгоритм построения кодового дерева кода Хаффмена

где $\bar{l}' = \sum_{m=1}^{M-1} p'_m l'_m$ – средняя длина кодовых слов кода C'. Последние два слагаемых в правой части не зависят от кода, поэтому код, минимизирующий $\bar{l}'$, одновременно обеспечивает минимум для $\bar{l}$. □

Итак, сформулированные свойства оптимальных префиксных кодов сводят задачу построения кода объема M к задаче построения кодов объема $M' = M - 1$. Это означает, что мы получили рекуррентное правило построения кодового дерева оптимального неравномерного кода. Это правило в виде алгоритма представлено на рис.2.4.

После выполнения алгоритма будет получено кодовое дерево кода, который, как доказано выше, имеет наименьшую возможную среднюю длину кодовых слов.

Пример 2.4.1 Рассмотрим ансамбль буквенных сообщений $X = \{a, b, c, d, e, f\}$ с вероятностями букв *{0,35, 0,2, 0,15, 0,1, 0,1, 0,1}* соответственно. Кодовое дерево и код Хаффмена показаны на рис. 2.5. Энтропия источника $H = 2,4016$. Средняя длина кодовых слов равна $\bar{l} = 2(0,35 + 0,2) + 3(0,15 + 3 \times 0,1) = 2,45$. Согласно свойствам 2.4.1 – 2.4.4 не существует кода для X со средней длиной кодовых слов меньшей, чем 2,45.



Буква	Кодовое слово
a	00
b	10
c	010
d	011
e	110
f	111

Рис. 2.5: Пример построения кода Хаффмена.

2.5 Код Шеннона

Рассмотрим источник, выбирающий буквы из множества $X = \{1, ..., M\}$ с вероятностями $\{p_1, ..., p_M\}$. Считаем, что буквы упорядочены по убыванию вероятностей, т.е. $p_1 \geq p_2 \geq ... \geq p_M$. Сопоставим, кроме того, каждой букве так называемую *кумулятивную вероятность* по правилу

$$q_1 = 0, \quad q_2 = p_1, \quad \ldots, \quad q_M = \sum_{i=1}^{M-1} p_i.$$

Кодовым словом *кода Шеннона* для сообщения с номером m является двоичная последовательность, представляющая собой первые $l_m = \lceil -\log p_m \rceil$ разрядов после запятой в двоичной записи числа q_m.

Пример 2.5.1 Рассмотрим тот же ансамбль, что и в примере 2.4.1. В таблице 2.1 приведены промежуточные вычисления и результат построения кода Шеннона. Средняя длина кодовых слов $\bar{l} = 2,95$. В данном случае избыточность кода Шеннона оказалась на 0,5 бита больше, чем избыточность кода Хаффмена. Кодовое дерево кода показано на рис. 2.6. Из этого рисунка понятно, почему код неэффективен. Кодовые слова для букв b, d, e, f могут быть укорочены на 1 бит без потери свойства однозначной декодируемости.

Докажем однозначную декодируемость кода Шеннона. Для этого выберем сообщения с номерами i и j, $i < j$. Кодовое слово $\boldsymbol{c}_i$ для i заведомо короче, чем слово $\boldsymbol{c}_j$ для j, поэтому достаточно доказать, что эти слова

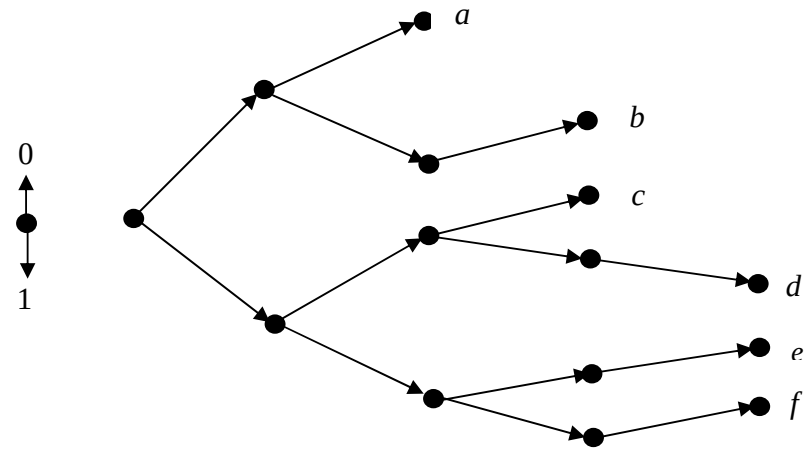


Рис. 2.6: Кодовое дерево кода Шеннона для ансамбля из примера 2.5.1.

Таблица 2.1: Построение кода Шеннона для примера 2.5.1

x	p_m	q_m	l_m	Двоичная запись q_m	Кодовое слово $\boldsymbol{c}_m$
a	0,35	0,00	2	0,00...	00
b	0,20	0,35	3	0,0101...	010
c	0,15	0,55	3	0,10001...	100
d	0,10	0,70	4	0,10110...	1011
e	0,10	0,80	4	0,11001...	1100
f	0,10	0,90	4	0,11100...	1110

отличаются в одном из первых l_i символов. Рассмотрим разность

$$q_j - q_i = \sum_{k=1}^{j-1} p_k - \sum_{k=1}^{i-1} p_k = \sum_{k=i}^{j-1} p_k \geq p_i.$$

Вспомним, что длина слова и его вероятность связаны соотношением

$$l_i = \lceil -\log p_i \rceil \geq -\log p_i.$$

Поэтому

$$p_i \geq 2^{-l_i}.$$

С учетом этого неравенства

$$q_j - q_i \geq 2^{-l_i}.$$

В двоичной записи числа в правой части мы имеем после запятой $l_i - 1$ нулей и единицу в позиции с номером l_i. Это означает, что по меньшей

мере в одном из l_i разрядов слова $\boldsymbol{c}_i$ и $\boldsymbol{c}_j$ отличаются и, следовательно, $\boldsymbol{c}_i$ не является префиксом для $\boldsymbol{c}_j$. Поскольку это верно для любой пары слов, то код является префиксным.

Заметим, что длины кодовых слов в коде Шеннона точно такие же, какие были выбраны при доказательстве прямой теоремы кодирования. Повторяя выкладки, получим уже известную оценку для средней длины кодовых слов

$$\bar{l} < H + 1.$$

Примечательно, что при построении кода Шеннона мы выбрали длины кодовых слов приблизительно равными (чуть большими) собственной информации соответствующих сообщений. В результате средняя длина кодовых слов оказалось приблизительно равной (чуть большей) энтропии ансамбля.

Хотя конструкция кода и анализ его характеристик уже достаточно понятны, мы обсудим еще одну, графическую, интерпретацию процесса кодирования. Она будет полезна в дальнейшем при обсуждении арифметического кодирования.

Рассмотрим числовой отрезок [0,1), на котором расположим один за другим отрезки длины $p_1, ..., p_M$. На рис. 2.7 приведен пример разметки отрезка для случая $M = 3$, $p_1 = 0,6$, $p_2 = 0,3$, $p_3 = 0,1$. Как видно на рис. 2.7а, кумулятивные вероятности $q_1 = 0$, $q_2 = 0,6$, и $q_3 = 0,9$ соответствуют началам отрезков. Эти точки идентифицируют сообщения источника.

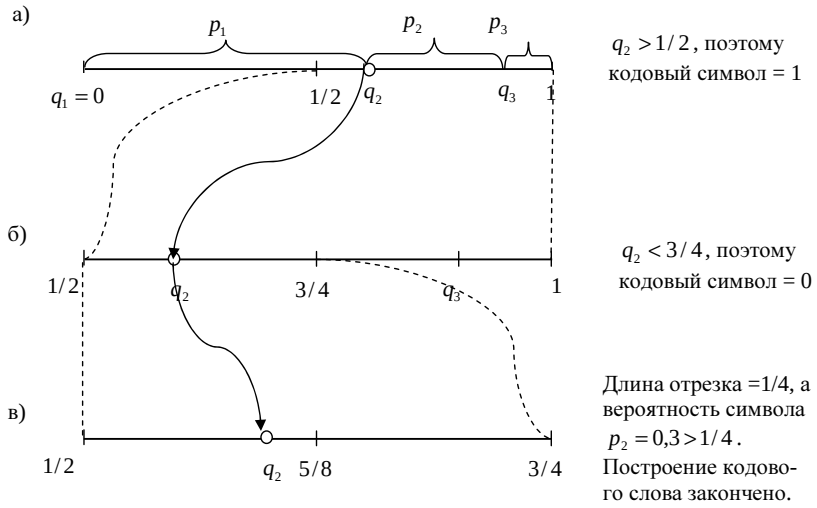


Рис. 2.7: Графическая интерпретация кода Шеннона

Предположим, что требуется закодировать сообщение с номером $m = 2$. Соответствующая ему точка q_m помечена на рис. 2.7а-в. кружком. На первом шаге передачи передачей кодового символа 0 либо 1 кодер указывает, в какой половине отрезка [0,1) (левой или правой) находится начало соответствующего сообщению отрезка. В данном случае передается 1 и тем самым область возможных положений передаваемой точки уменьшается вдвое, что и показано на рис. 2.7б. На следующем шаге передается символ 0, т.к. точка находится в левой половине интервала и длина интервала неопределенности уменьшается до 1/4. Заметим, что после второго шага в интервале неопределенности осталась только одна точка и поэтому передача может быть завершена. Это произошло не случайно. Дело в том, что длина интервала равна 1/4, что меньше длины кратчайшего прилегающего отрезка 0,3. Именно поэтому гарантируется единственность точки и, значит, однозначность декодирования.

В общем случае после передачи l_m двоичных символов длина интервала неопределенности равна 2^{-l_m}. Декодирование будет однозначным, если $2^{-l_m} \leq p_m$ или $l_m \geq -\log p_m$. Это условие в точности совпадает с правилом выбора длин кодовых слов в коде Шеннона.

Заметим, что при выборе длины кодовых слов мы ориентировались только на отрезок, лежащий справа от точки q_m. Упорядоченность букв по убыванию вероятностей гарантирует, что левый отрезок всегда будет длиннее правого.

2.6 Код Гилберта-Мура

При построении кода Шеннона мы требовали упорядоченности сообщений по убыванию вероятностей. В алгоритме построения кода Шеннона сортировка букв входного алфавита, пожалуй, наиболее трудоемкая часть. Мы заметно упростим построение кода, если модифицируем кодирование таким образом, чтобы упорядоченность не требовалась. Графическая интерпретация кода Шеннона, показанная на рис. 2.7, подсказывает почти очевидный путь к решению этой задачи.

Предположим, что вероятности не упорядочены и тогда при кодировании сообщения с номером m нужно учитывать не только вероятность (длину отрезка) p_m, но и длину предшествующего отрезка p_{m-1}, которая может быть очень маленькой, почти нулевой, и тогда длина слова будет большой даже если вероятность p_m велика. Как же избавиться от влияния p_{m-1}?

Очень просто. Нужно соответствующую сообщению точку переме-

стить из начала отрезка (точка q_m) в его середину (точка $q_m + p_m/2$), а длину кодового слова l_m выбрать так, чтобы к концу передачи кодового слова длина интервала неопределенности была не больше $p_m/2$. Эти l_m бит и будут кодовым словом кода Гилберта-Мура!

Определим код Гилберта-Мура формально.

Рассмотрим источник, выбирающий буквы из алфавита $X = \{1, ..., M\}$ с вероятностями $\{p_1, ..., p_M\}$. Сопоставим каждой букве $m = 1, ..., M$ кумулятивную вероятность $q_m = \sum_{i=1}^{m-1} p_i$ и вычислим для каждой буквы величину σ_m по формуле

$$\sigma_m = q_m + \frac{p_m}{2}.$$

Кодовым словом кода Гилберта-Мура для x_m является двоичная последовательность, представляющая собой первые $l_m = \lceil -\log(p_m/2) \rceil$ разрядов после запятой в двоичной записи числа σ_m.

Пример 2.6.1 Рассмотрим источник с распределением вероятностей $p_1 = 0,1$, $p_2 = 0,6$, $p_3 = 0,3$. Вычисления, связанные с построением кода Гилберта-Мура для этого источника, приведены в таблице 2.2. Запись $[a]$ обозначает представление числа a в двоичной форме. В последнем столбце таблице показано, как выглядели бы кодовые слова $\tilde{\boldsymbol{c}}_m$ соответствующего кода Шеннона, если бы мы забыли упорядочить буквы по вероятностям. Видно, что код получился бы непрефиксным в отличие от кода Гилберта-Мура.

Таблица 2.2: Пример кода Гилберта-Мура

x_m	p_m	q_m	σ_m	l_m	$\boldsymbol{c}_m$ [a]	$\tilde{\boldsymbol{c}}_m$ [b]
1	0,1	0,0=[0,00000...]	0,05=[0,00001...]	5	00001	0000
2	0,6	0,1=[0,00011...]	0,40=[0,01100...]	2	01	0
3	0,3	0,7=[0,10110...]	0,85=[0,11011...]	3	110	10

[a] Кодовые слова кода Гилберта-Мура

[b] Кодовые слова кода Шеннона без упорядочения вероятностей букв

Докажем однозначную декодируемость кода Гилберта-Мура в общем случае. Для этого выберем сообщения с номерами i и j, $i < j$. Понятно, что $\sigma_j > \sigma_i$. Нужно доказать, что соответствующие слова $\boldsymbol{c}_i$ и $\boldsymbol{c}_j$ отличаются хотя бы в одном из первых $\min\{l_i, l_j\}$ кодовых символов.

Рассмотрим разность

$$\begin{aligned}\sigma_j - \sigma_i &= \sum_{h=1}^{j-1} p_h + \frac{p_j}{2} - \sum_{h=1}^{i-1} p_h - \frac{p_i}{2} = \\ &= \sum_{h=i}^{j-1} p_h + \frac{p_j - p_i}{2} \geq \\ &\geq p_i + \frac{p_j - p_i}{2} = \\ &= \frac{p_j + p_i}{2} \geq \frac{\max\{p_i, p_j\}}{2}.\end{aligned}$$

Вспомним, что длина слова и его вероятность связаны соотношением

$$l_m = \left\lceil -\log \frac{p_m}{2} \right\rceil \geq -\log \frac{p_m}{2}.$$

Отсюда следует

$$\sigma_j - \sigma_i \geq \frac{\max\{p_i, p_j\}}{2} \geq 2^{-\min\{l_i, l_j\}},$$

а это означает, что слова $\boldsymbol{c}_i$ и $\boldsymbol{c}_j$ отличаются в одном из первых $\min\{l_i, l_j\}$ двоичных символов и поэтому ни одно из двух слов не может быть началом другого.

Поскольку все слова кода Гилберта-Мура ровно на единицу длиннее слов кода Шеннона, получаем следующую оценку средней длины кодовых слов

$$\bar{l} < H + 2.$$

Мы завершим рассмотрение кода Гилберта-Мура графической интерпретацией, представленной на рис. 2.8 для источника из примера 2.6.1. Мы снова предполагаем, что передается буква с номером 2. Ей соответствует точка σ_2. По построению, соседние точки σ удалены от нее на расстояние по меньшей мере $p_2/2$. Кодер передает бит за битом и при этом каждый раз интервал неопределенности сужается вдвое. Передачу можно завершить, когда длина интервала неопределенности будет не больше $p_2/2$. В данном примере достаточно передать 2 бита.

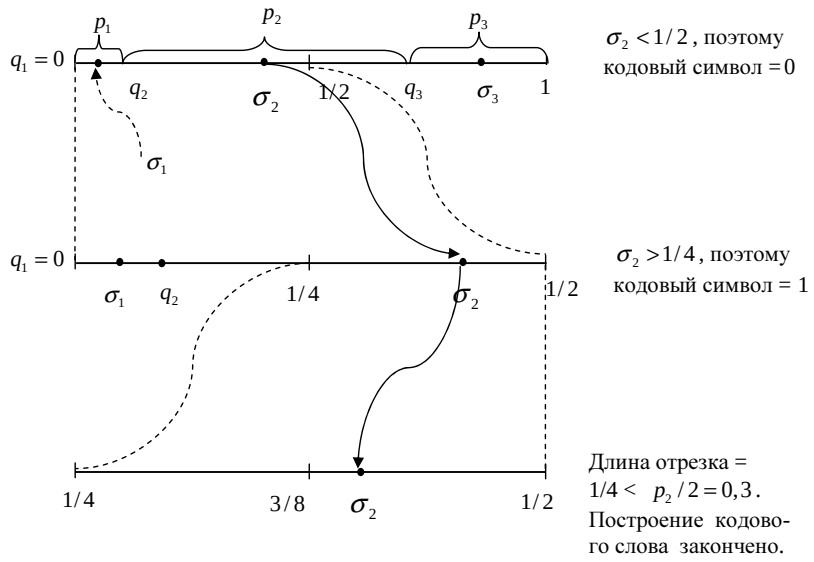


Рис. 2.8: Графическая интерпретация кода Гилберта-Мура

2.7 Неравномерное кодирование для стационарного источника.

2.7.1 Постановка задачи. Теоремы кодирования

В этом параграфе мы перейдем к решению задачи кодирования последовательностей сообщений. Разумеется, если мы рассматриваем стационарный источник и его распределение вероятностей на буквах не меняется от буквы к букве, то любой из описанных выше способов кодирования может быть использован для кодирования отдельных сообщений источника. Во многих случаях именно такой подход используется на практике как самый простой и достаточно эффективный.

В то же время, можно выделить класс ситуаций, когда побуквенное кодирование заведомо неоптимально. Во-первых, из теоремы об энтропии на сообщение стационарного источника следует, что учет памяти источника потенциально может значительно повысить эффективность кодирования. Во-вторых, побуквенные методы затрачивают как минимум 1 бит на сообщение, тогда как энтропия на сообщения может быть значительно меньше единицы.

Итак, рассмотрим последовательность $x_1, x_2, ..., x_i \in X = \{x\}$, наблюдаемую на выходе дискретного стационарного источника, для которого известно вероятностное описание, т.е. можно вычислить все многомерные распределения вероятностей и по ним – энтропию на сообщение $H = H_\infty(X)$.

Пусть указан некоторый способ кодирования, который для любых n для каждой последовательности $\boldsymbol{x} \in X^n$ на выходе источника строит кодовое слово $\boldsymbol{c}(\boldsymbol{x})$ длины $l(\boldsymbol{x})$. Тогда *средняя скорость кодирования для блоков длины* n определяется как

$$\bar{R}_n = \frac{1}{n}\mathbf{M}\left[l(\boldsymbol{x})\right]$$

бит/сообщение источника. Подбирая длину блоков, при которой средняя скорость будет наименьшей, получаем следующее определение для *средней скорости кодирования*

$$\bar{R} = \inf_n \bar{R}_n.$$

Мы пишем нижнюю грань inf вместо минимума, поскольку наименьшее значение скорости может достигаться в пределе при $n \to \infty$.

Рассматриваемое кодирование называется FV-кодированием (fixed-to-variable), поскольку блоки из фиксированного числа сообщений n кодируются кодовыми словами переменной длины. Наша задача – связать достижимые значения средней скорости FV-кодирования с характеристиками источника, в частности, с его энтропией на сообщение H. Начнем с обратной теоремы кодирования.

Теорема 2.5 *Для дискретного стационарного источника с энтропией на сообщение H для любого FV-кодирования имеет место неравенство*

$$\bar{R} \geq H.$$

Доказательство. Рассмотрим множество X^n. К элементам этого множества применим теорему побуквенного кодирования. Получим

$$\mathbf{M}\left[l(\boldsymbol{x})\right] \geq H(X^n) = nH_n(X) \geq nH_\infty(X) = nH.$$

Здесь второе неравенство имеет место в силу того, что $H_n(X)$ не возрастает с увеличением n. Отсюда следует, что

$$\bar{R}_n \geq H$$

при любых n. Таким образом,

$$\bar{R} = \inf_n R_n \geq H,$$

что и требовалось доказать. □

Почти также просто доказывается прямая теорема кодирования.

Теорема 2.6 *Для дискретного стационарного источника с энтропией на сообщение H и для любого $\delta > 0$ существует способ неравномерного FV-кодирования такой, для которого*

$$\bar{R} \leq H + \delta.$$

Доказательство. Воспользовавшись прямой теоремой побуквенного кодирования для ансамбля X^n, мы можем утверждать, что для некоторого побуквенного кода имеет место неравенство

$$\mathbf{M}\left[l(\boldsymbol{x})\right] \leq H(X^n) + 1 = nH_n(X) + 1. \tag{2.4}$$

По определению предела числовой последовательности существует достаточно большое число n_1 такое, что при всех $n \geq n_1$ имеет место неравенство

$$|H_n(X) - H| \leq \frac{\delta}{2},$$

из которого следует, что

$$H_n(X) \leq H + \frac{\delta}{2}, \quad n > n_1. \tag{2.5}$$

Найдем n_2 такое, что при $n \geq n_2$ имеет место неравенство

$$\frac{1}{n} \leq \frac{\delta}{2}. \tag{2.6}$$

С учетом (2.5) и (2.6) из (2.4) получаем при $n \geq \max\{n_1, n_2\}$

$$\begin{aligned}
\bar{R} &= \inf_m \bar{R}_m \leq \\
&\leq \bar{R}_n = \\
&= \frac{\mathbf{M}\left[l(\boldsymbol{x})\right]}{n} \leq \\
&\leq H_n(X) + \frac{1}{n} \leq \\
&\leq H + \frac{\delta}{2} + \frac{\delta}{2} = \\
&= H + \delta,
\end{aligned}$$

что и требовалось доказать. □

Итак, выбрав достаточно большую длину блоков n и применив к блокам побуквенное кодирование, мы получим кодирование со средней скоростью

$$H \leq \bar{R} \leq H + o(n),$$

где $o(n) \to 0$ при $n \to \infty$.

К сожалению, этот внешне оптимистический результат оказывается почти бесполезным при решении практических задач. Основное препятствие на пути его применения – это экспоненциальный рост сложности при увеличении длины блоков n. Поясним эту проблему следующим простым примером.

Предположим, что кодированию подлежат файлы, хранящиеся в памяти компьютера. Символы источника – байты и, значит, объем алфавита $|X| = 2^8 = 256$. При кодировании последовательностей длины $n = 2$ объем алфавита вырастает до $|X^2| = 2^{16} = 65536$. Далее при $n = 3$, 4,. . . объемы алфавитов будут $2^{24} = 16777216$, $2^{32} = 4294967296, \ldots$ Понятно, что работать с кодами таких размеров невозможно.

Описываемый в следующем параграфе метод арифметического кодирования позволяет эффективно кодировать блоки длины n с избыточностью порядка $2/n$ и со сложностью растущей только пропорционально квадрату длины блока n. За счет пренебрежимо малого проигрыша в скорости кода сложность может быть сделана даже линейной по длине кода. Неудивительно, что арифметическое кодирование все шире применяется в разнообразных системах обработки информации.

2.7.2 Арифметическое кодирование

Рассмотрим для простоты дискретный постоянный источник, выбирающий сообщения из множества $X = \{1, ..., M\}$, с вероятностями $\{p_1, ..., p_M\}$. Обозначим через $\{q_1, ..., q_M\}$ кумулятивные вероятности сообщений. Наша задача состоит в кодировании последовательностей множества $X^n = \{\boldsymbol{x}\}$. При описании алгоритма кодирования мы будем использовать обозначение $\boldsymbol{x}_i^j$ для краткой записи подпоследовательности $(x_i, ..., x_j)$ в последовательности $\boldsymbol{x} = (x_1, ..., x_n)$.

Мы хотели бы применить к ансамблю $X^n = \{\boldsymbol{x}\}$ достаточно простой и эффективной побуквенный код. Упрощение состоит в том, ни кодер ни декодер не хранят и не строят всего множества из $|X^n|$ кодовых слов. Вместо этого при передаче конкретной последовательности $\boldsymbol{x}$ кодером вычисляется кодовое слово $\boldsymbol{c}(\boldsymbol{x})$ только для данной последовательности $\boldsymbol{x}$. Правило кодирования, конечно, известно декодеру и он восстанавливает $\boldsymbol{x}$ по $\boldsymbol{c}(\boldsymbol{x})$, не имея полного списка кодовых слов.

Возможные кандидаты на использование в такой схеме – код Шеннона и код Гилберта-Мура. Однако использование кода Шеннона предполагает упорядоченность сообщений по убыванию вероятностей. При больших n сложность упорядочения окажется недопустимо большой, по-

этому единственным претендентом остается код Гилберта-Мура.

В соответствии с правилом построения кода Гилберта-Мура кодовое слова формируется по вероятности $p(\boldsymbol{x})$ и кумулятивной вероятности $q(\boldsymbol{x})$ как первые $l(\boldsymbol{x}) = \lceil -\log p(\boldsymbol{x}) + 1 \rceil$ разрядов после точки в двоичной записи числа $\sigma(\boldsymbol{x}) = q(\boldsymbol{x}) + p(\boldsymbol{x})/2$.

Для того, чтобы вычислить $q(\boldsymbol{x})$, надо условиться о некоторой нумерации последовательностей из X^n. Наиболее естественный способ нумерации последовательностей – использование лексикографической упорядоченности. Лексикографический порядок на последовательностях будет обозначаться знаком "$\prec$". Запись $\boldsymbol{y} \prec \boldsymbol{x}$ будет означать, что $\boldsymbol{y}$ лексикографически предшествует $\boldsymbol{x}$.

Понятие *лексикографического порядка* определяется следующим образом.

Для последовательностей длины 1 (для отдельных сообщений из X) мы считаем, что сообщение с меньшим номером предшествуют сообщению с большим номером. Если, например, элементы X – числа, то $x \prec x'$, если $x < x'$, $x, x' \in X$.

Для двух последовательностей $\boldsymbol{x} = (x_1, ..., x_n)$, $\boldsymbol{y} = (y_1, ..., y_n)$ обозначим через i наименьший индекс такой, что $x_i \neq y_i$. Тогда $\boldsymbol{y} \prec \boldsymbol{x}$, если $y_i \prec x_i$.

Нетрудно видеть, что лексикографический порядок – это порядок, который обычно используется при составлении словарей.

Итак, основная задача состоит в вычислении кумулятивной вероятности

$$q(\boldsymbol{x}) = \sum_{\boldsymbol{y} \prec \boldsymbol{x}} p(\boldsymbol{y}), \tag{2.7}$$

поскольку для источника без памяти вероятности последовательностей $p(\boldsymbol{x})$ вычисляются достаточно просто по формуле

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i).$$

Выведем рекуррентную формулу для вычисления $q(\boldsymbol{x})$. Для этого выразим вероятность $q(\boldsymbol{x}_1^n)$ через $q(\boldsymbol{x}_1^{n-1})$. Следующая цепочка элемен-

тарных преобразований показывает, как это можно сделать:

$$\begin{aligned} q(\boldsymbol{x}_1^n) &= \sum_{\boldsymbol{y}_1^n \prec \mathbf{x}_1^n} p(\boldsymbol{y}_1^n) = \\ &= \sum_{\boldsymbol{y}_1^{n-1} \prec \boldsymbol{x}_1^{n-1}} \sum_{y_n} p(\boldsymbol{y}_1^{n-1} y_n) + \sum_{\boldsymbol{y}_1^{n-1} = \mathbf{x}_1^{n-1}} \sum_{y_n \prec x_n} p(\boldsymbol{y}_1^{n-1} y_n) = \\ &= \sum_{\boldsymbol{y}_1^{n-1} \prec \boldsymbol{x}_1^{n-1}} p(\boldsymbol{y}_1^{n-1}) + \sum_{\boldsymbol{y}_1^{n-1} = \boldsymbol{x}_1^{n-1}} p(\boldsymbol{y}_1^{n-1}) \sum_{y_n \prec x_n} p(y_n) = \\ &= q(\boldsymbol{x}_1^{n-1}) + p(\boldsymbol{x}_1^{n-1}) q(x_n), \end{aligned}$$

где $q(x_n)$ обозначает кумулятивную вероятность символа x_n.

Подытожим наши выкладки в виде рекуррентных формул

$$q(\boldsymbol{x}_1^n) = q(\boldsymbol{x}_1^{n-1}) + p(\boldsymbol{x}_1^{n-1}) q(x_n); \tag{2.8}$$

$$p(\boldsymbol{x}_1^n) = p(\boldsymbol{x}_1^{n-1}) p(x_n). \tag{2.9}$$

В этой рекурсии каждая пара значений $(q(\boldsymbol{x}_1^i),\ p(\boldsymbol{x}_1^i))$ используется ровно на одном шаге при вычислении следующей пары $(q(\boldsymbol{x}_1^{i+1}),p(\boldsymbol{x}_1^{i+1}))$. Поэтому при реализации арифметического кодирования вновь вычисленные значения записываются в те же ячейки памяти, в которых находились предыдущие значения. В приведенном на рис. 2.9 алгоритме кумулятивные вероятности $q(\boldsymbol{x}_1^i)$, $i = 1, 2, ...$ хранятся в виде переменной F, вероятности последовательностей $p(\boldsymbol{x}_1^i)$, $i = 1, 2, ...$ – в виде переменной G.

Пример 2.7.1 Рассмотрим источник из примера 2.6.1: $X = \{a, b, c\}$, распределение вероятностей $p_a = 0,1$, $p_b = 0,6$, $p_c = 0,3$. Вычисления, выполняемые арифметическим кодером при кодировании последовательности $\boldsymbol{x} = (bcbab)$ длины $n = 5$, приведены в таблице 2.3 В этой таблице через $\hat{F}$ обозначено число $(F + G/2)$ округленное вниз с точностью до $\lceil -\log G + 1 \rceil = 9$ двоичных разрядов.

Графическая интерпретация процесса кодирования аналогичная интерпретации кодов Шеннона и Гилберта-Мура показана на рис. 2.10

Как показано на рисунке, на каждом шаге кодирования пересчитывается начальная точка F и длина G отрезка, в котором будет расположено число, соответствующее кодовой последовательности для заданной

Input: Объем алфавита M
вероятности букв $p_i, i = 1, ..., M$
длина последовательности n
последовательность на выходе источника $(x_1, ..., x_n)$,
Output: Кодовое слово арифметического кода

`Кумулятивные вероятности:`
$q_1 = 0$;
for $i = 2$ **to** M **do**
$q_i = q_{i-1} + p_{i-1}$;
end

`Кодирование:`
for $i = 1$ **to** n **do**
$F \leftarrow F + q(x_i)G$;
$G \leftarrow p(x_i)G$;.
end

`Формирование кодового слова:`
$\boldsymbol{c} \leftarrow$ первые $\lceil -\log G \rceil + 1$ разрядов после запятой в двоичной записи числа $F + G/2$.

Рис. 2.9: Алгоритм арифметического кодирования

Таблица 2.3: Кодирование последовательности арифметическим кодом

Шаг i	x_i	$p(x_i)$	$q(x_i)$	F	G
0	-	-	-	0,0000	1,0000
1	b	0,6	0,1	0,1000	0,6000
2	c	0,3	0,7	0,5200	0,1800
3	b	0,6	0,1	0,5380	0,1080
4	a	0,1	0,0	0,5380	0,0108
5	b	0,6	0,1	0,5391	0,0065
6	Длина кодового слова $\lceil -\log G + 1 \rceil = 9$ Кодовое слово $F + G/2 = 0,5423... \rightarrow$ $\rightarrow \hat{F} = 0,541 \rightarrow 100010101$				

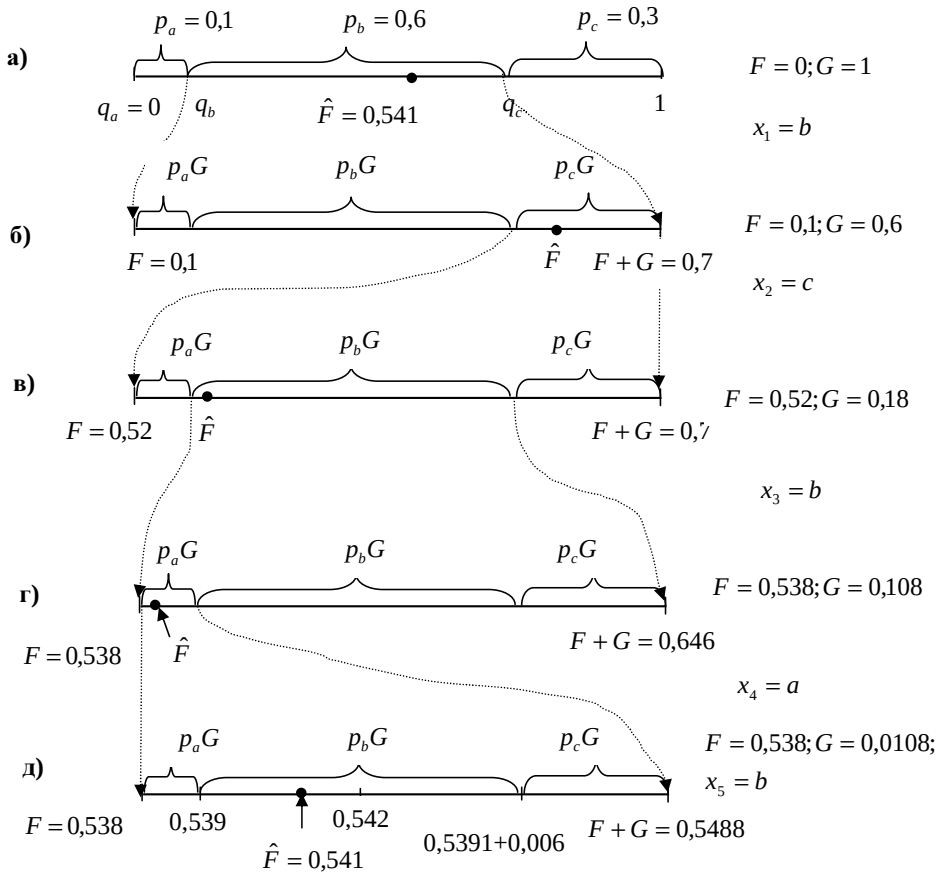


Рис. 2.10: Графическая интерпретация арифметического кодирования

последовательности сообщений. Так, после первого шага ($x_1 = b$) мы знаем, что точка будет расположена в отрезке [0,1, 0,7). Более детально этот отрезок показан на рис. 2.10б. Поскольку вторая буква $x_2 = c$, начальная точка перемещается в значение 0,52, а длина интервала уменьшается до 0,18 и т.д. После 5-го шага $F = 0,5391$. Добавив смещение $G/2$ и округлив до 9 двоичных знаков, получаем величину $\hat{F} = 0,541$. Тем самым вся последовательность сообщений отображается в одну точку интервала [0,1). Эта точка помечена кружком на рисунках 2.10а – д. Как было показано выше, 9 разрядов достаточно для того, чтобы последовательность была восстановлена однозначно, т.е. ближайшая возможная точка, соответствующая другой последовательности сообщений, удалена от $\hat{F}$ на расстояние не менее $1/2^9 = 1/512$.

Обсудим кратко вопрос о сложности кодирования.

Из описания алгоритма следует, что на каждом шаге кодирования выполняется одно сложение и 2 умножения. Отсюда легко сделать неправильный вывод о том, что сложность кодирования последовательности из n сообщений пропорциональна n. Это неверно, поскольку на каждом шаге линейно растет сложность выполнения самих операций сложения и умножения, т.к. нарастает число двоичных разрядов, необходимых для записи операндов.

Предположим, что для представления вероятностей $p_1, ..., p_M$ использованы числа разрядности d. После первого шага кодирования точное представление F и G потребует $2d$ разрядов, ..., после n шагов кодирования кодер и декодер будут работать (в худшем случае) с числами разрядности nd, и, следовательно, суммарная сложность имеет порядок

$$d + 2d + ... + nd = \frac{n(n+1)}{2}d.$$

Таким образом, можно говорить о том, что сложность арифметического кодирования пропорциональна n^2. На самом деле, все же возможна практическая реализация арифметического кодирования со сложностью пропорциональной n, но за счет некоторой (очень небольшой) потери в точности вычислений и, следовательно, в эффективности кодирования.

Рассмотрим декодирование арифметического кода. Декодеру арифметического кода известны алфавит $X = \{1, ..., M\}$, вероятности $\{p_1, ..., p_M\}$, кумулятивные вероятности $\{q_1, ..., q_M\}$, длина последовательности сообщений n и полученное из канала значение $\hat{F}$. Задача состоит в вычислении последовательности сообщений $\boldsymbol{x}$. Эту задачу решает показанный на рис. 2.11 алгоритм.

Input: Объем алфавита M
вероятности букв $\{p_1, ..., p_M\}$
кумулятивные вероятности букв $q_i, i = 1, ..., M$
длина декодируемой последовательности n
кодовое слово в виде числа $\hat{F}$.
Output: Декодированная последовательность букв $(x_1, ..., x_n)$

`Инициализация:` $q_{M+1} = 1$; $S = 0$; $G = 1$.

`Декодирование:`
for $i = 1$ **to** n **do**
    $j = 1$;
    while $S + q_{j+1}G < \hat{F}$ **do**
        $j \leftarrow j + 1$.
    end
    $S \leftarrow S + q_jG$;
    $G \leftarrow p_jG$;
    $x_i = j$.
end

`Результат:последовательность` $(x_1, ..., x_n)$;

Рис. 2.11: Алгоритм декодирования арифметического кода

В этом алгоритме после выполнения i шагов переменная G равна вероятности $p(\boldsymbol{x}_1^i)$ последовательности из первых i символов, а переменная S равна кумулятивной вероятности $q(\boldsymbol{x}_1^i)$.

Пример 2.7.2 Рассмотрим источник $X = \{a, b, c\}$ с распределением вероятностей $p_a = 0,1$, $p_b = 0,6$, $p_c = 0,3$. Предположим, что на вход декодера поступила двоичная последовательность 0100010101. По этой последовательности нужно восстановить последовательность закодированных сообщений. Простое, но не совсем честное решение – заглянуть на несколько страниц назад, ведь в примере 2.7.1 мы получили именно эту последовательность на выходе кодера. Если же добросовестно следовать алгоритму, приведенному на рис. 2.11, то на промежуточных шагах будут получаться числа, приведенные в таблице 2.4. Как видно из таблицы, последовательность сообщений восстановлена правильно.

Детали практической реализации арифметического кодирования обсуждаются в учебнике [12].

Таблица 2.4: Декодирование последовательности из примера 2.7.2

Шаг	S	G	Гипотеза x	$q(x)$	$S + qG$	Решение x_i	$p(x)$
0	100010101$\rightarrow \hat{F}$ =0,541						
1	0,0000	1,0000	a	0,0	0,0000$< \hat{F}$	b	0,6
			b	0,1	**0,1000**$< \hat{F}$		
			c	0,7	0,7000$> \hat{F}$		
2	0,1000	0,6000	a	0,0	0,1000$< \hat{F}$	c	0,3
			b	0,1	0,1600$< \hat{F}$		
			c	0,7	**0,5200**$< \hat{F}$		
3	0,5200	0,1800	a	0,0	0,5200$< \hat{F}$	b	0,6
			b	0,1	**0,5380**$< \hat{F}$		
			c	0,7	0,6460$> \hat{F}$		
4	0,5380	0,1080	a	0,0	**0,5380**$< \hat{F}$	a	0,1
			b	0,1	0,5488$> \hat{F}$		
5	0,5380	0,0108	a	0,0	0,5380$< \hat{F}$	b	0,6
			b	0,1	**0,5391**$< \hat{F}$		
			c	0,7	0,5456$> \hat{F}$		

2.8 Постановка задачи универсального кодирования источников

Рассмотренные в предыдущем разделе алгоритмы кодирования эффективны и практичны. Однако их общим недостатком является то, что они эффективны только в том случае, когда вероятностная модель источника совпадает или очень близка к модели, для которой строился код. В данном разделе мы рассмотрим задачу кодирования источника в условиях отсутствия информации или неполной информированности кодера и декодера о характеристиках источника.

Универсальное кодирование предполагает, что входом кодера является последовательность сообщений $\boldsymbol{x} = (x_1, ..., x_n)$ некоторого источника X. Алфавит кодера, длина последовательности n, алгоритм работы кодера известны декодеру. Статистические свойства источника заранее неизвестны. Задача, как и прежде, состоит в том, чтобы обеспечить эффективное кодирование, то есть найти представление последовательности источника двоичной кодовой последовательностью наименьшей длины. Однако, решать задачу в такой постановке не имеет смысла. Дело в том, что не может существовать алгоритм кодирования, который бы "сжимал" любые последовательности источника. Мы уже говорили, что при неравномерном префиксном кодировании уменьшение длины одной кодовой последовательности на 1 символ достигается увеличением длины двух других последовательностей на 1 символ. Если считать все последовательности источника одинаково вероятными, то любой алгоритм кодирования "в среднем" будет проигрывать равномерному кодированию, затрачивающему на каждое сообщение $\log |X|$ двоичных символов.

Мы существенно упростим ситуацию, если примем предположение о том, что источник сообщений является стационарным. В этом случае мы можем подсчитать для него энтропию на сообщение и тем самым определить минимально достижимую скорость кодирования (понятно, что обратная теорема кодирования остается справедливой при отсутствии у кодера и декодера информации о характеристиках источника). Для стационарного источника естественно считать "хорошим" тот способ кодирования, который обеспечивает скорость близкую к той скорости кодирования, которая могла бы быть достигнута при известной статистике.

Заметим, что при решении многих практических задач есть возможность дополнительно сузить класс возможных моделей источника, то есть помимо предположения о стационарности сделать и другие предположения. Разумеется, любая априорная информация может быть использована для повышения эффективности кодирования.

Опираясь на эти практические соображения, мы приходим к следующей постановке задачи. Пусть $\Omega = \{\omega\}$ представляет собой некоторый класс (множество) моделей источников (не обязательно дискретное). Например, в качестве Ω может рассматриваться множество дискретных постоянных источников. В этом случае конкретный элемент множества определяется одномерным распределением вероятностей на X. Обозначим через H_ω энтропию на сообщение источника для заданной модели $\omega \in \Omega$. Пусть заданный алгоритм кодирования обеспечивает для этой модели при кодировании последовательностей длины n среднюю скорость $\bar{R}_n(\omega)$. Тем самым определена избыточность $r_n(\omega) = \bar{R}_n(\omega) - H_\omega$. *Избыточностью кодирования* для данного алгоритма для класса моделей Ω называется величина

$$r_n(\Omega) = \sup_{\omega\in\Omega} \left[\bar{R}_n(\omega) - H_\omega\right]. \tag{2.10}$$

Задача состоит в построении алгоритма, минимизирующего избыточность $r_n(\Omega)$ для заданного класса моделей Ω.

Вполне естественно, что, как и в случае известной модели источника, с увеличением длины n кодируемой последовательности избыточность кодирования уменьшается. Если для заданного алгоритма имеет место соотношение

$$\lim_{n\to\infty} r_n(\Omega) = 0,$$

кодирование называется *универсальным* для множества Ω. Как мы увидим позже, обеспечить построить универсальные алгоритмы кодирования совсем несложно, и мы будем искать такие универсальные алгоритмы, для которых скорость убывания избыточности максимальна.

Помимо ограничений на множество моделей источников возможны ограничения на множество допустимых алгоритмов. Мы отметим два наиболее важных ограничения.

Во-первых, во многих практических задачах накладывается ограничение на допустимую задержку кодирования. Примерами могут служить кодер, предназначенный для сжатия файлов в реальном времени в процессе записи информации на жесткий диск или кодер, входящий в состав модема и сжимающий информацию перед передачей по линии связи. С другой стороны, можно указать широкий круг задач, в которых задержка кодирования не важна. Например, для производителя программного обеспечения при записи программ и данных на компактный диск задержка кодирования не имеет решающего значения.

По этому критерию различают две крайние ситуации. Один класс алгоритмов рассчитан на применение в тех случаях, когда задержку недо-

пустима. На английском языке такие алгоритмы называют on-line алгоритмами. Мы будем называть их *кодированием без задержки или мгновенным кодированием.* Общим свойством таких алгоритмов является то, что при кодировании каждой вновь поступившей буквы источника разрешается использовать информацию о предшествующих буквах, но не разрешается использовать информацию о будущих буквах.

Второй класс алгоритмов – алгоритмы для тех задач, в которых задержка не имеет значения. Эти алгоритмы на английском языке называются off-line алгоритмами. Мы будем использовать термин *"двухпроходное кодирование".* Этот термин правильно отражает суть кодирования при неограниченной задержке. Кодер сначала исследует статистические свойства последовательности и принимает решение о стратегии кодирования. Затем, на втором проходе, производится собственно кодирование.

Поскольку любой алгоритм мгновенного кодирования можно использовать в системах с неограниченной задержкой, второй класс алгоритмов, конечно, полностью включает первый класс алгоритмов. Вопрос, на который нам предстоит ответить – насколько велик выигрыш, который можно получить, "заглядывая вперед" при кодировании.

Еще одно ограничение, учитываемое при выборе алгоритма – сложность кодирования и декодирования.

Итак, цель исследования алгоритмов универсального кодирования – построение кодеров, обеспечивающих минимальную избыточность для заданного класса моделей при ограничениях на задержку и сложность.

2.9 Двухпроходное побуквенное кодирование

Без потери общности будем считать, что источник выбирает сообщения из множества $X = \{0, ..., M-1\}$. Пусть $\boldsymbol{x} = (x_1, ..., x_n)$ – последовательность на выходе источника. Множество сообщений X, и длину последовательности считаем заранее известными кодеру и декодеру.

Для простоты рассмотрим универсальное кодирование для класса источников без памяти с неизвестным распределением вероятностей на буквах источника. Рассмотрим следующий очевидный способ решения задачи. Кодер сначала просматривает последовательность и определяет число появлений $\tau_n(a)$ каждой буквы $a \in X$ в последовательности $\boldsymbol{x}$ длины n. Затем, используя полученную информацию, вычисляет оценки вероятностей сообщений и строит по ним некоторый код для множества X. На втором проходе этот код используется для кодирования последо-

вательности сообщений источника. Кодовое слово состоит из двух частей $\boldsymbol{c}(\boldsymbol{x}) = (\boldsymbol{c}_1(\boldsymbol{x}), \boldsymbol{c}_2(\boldsymbol{x}))$. Первая часть $\boldsymbol{c}_1(\boldsymbol{x})$ содержит информацию об использованном коде, вторая $\boldsymbol{c}_2(\boldsymbol{x})$ – собственно закодированную последовательность букв.

Декодер сначала по $\boldsymbol{c}_1(\boldsymbol{x})$ строит код, затем по $\boldsymbol{c}_2(\boldsymbol{x})$ восстанавливает одно за другим закодированные сообщения.

В это описание вкладывается целое семейство алгоритмов, отличающихся друг от друга выбором кода и способом передачи служебной информации. Мы рассмотрим алгоритм, основанный на побуквенном кодировании кодом Хаффмена.

Поясним работу двухпроходного побуквенного кодера примером. Предположим, что источником информации являются данные, хранящиеся в памяти ЭВМ в виде байтов, то есть объем алфавита $|X| = 256$. В качестве тестовой последовательности, на которой мы будем испытывать все рассматриваемые в этом разделе методы, мы выбрали поучительную пословицу

```
IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN
```
(2.11)

Мы заменили пробелы подчеркиваниями, чтобы сделать их видимыми и чтобы не забыть, что пробел, как и любой другой символ, должен быть закодирован и передан. Результаты статистического анализа кодируемого текста и соответствующий код Хаффмена для букв алфавита X приведены в таблице 2.5. Обозначим через l_1 и l_2 длину первой и второй части кодового слова. Длина l_2 равна сумме длин кодовых слов информационной последовательности. Подсчитать ее легко:

$$l_2 = 6 + 6 + 12 \times 2 + 5 \times 3 + \ldots + 6 = 178.$$

Чтобы подсчитать l_1, нужно условиться о способе передачи информации о коде. Первое, что приходит на ум – перечислить буквы, длины кодовых слов, сами кодовые слова. Этот подход заведомо неэффективен. Гораздо экономичнее передать структуру кодового дерева. Зная ее, декодер сам единственным образом восстановит все кодовые последовательности.

Кодовое дерево для рассматриваемого кода показано на рис. 2.12. Все вершины дерева размечены. Нулем помечены промежуточные вершины, единицей – концевые. Будем считывать символы, приписанные вершинам, ярус за ярусом, начиная с корня дерева, сверху вниз. В данном случае мы получим двоичную последовательность вида

$$00010000010100110111101101111.$$

Эта последовательность из 29 двоичных символов не только полностью описывает дерево, но и несет информацию о количестве различных букв в последовательности. Действительно, декодер по этой последовательности ярус за ярусом может восстановить дерево. Прочитав первый нулевой символ, он узнает, что последовательность содержала, по меньшей мере, 2 различные буквы. По следующим двум символам, он определяет, что на первом ярусе концевых узлов нет и, следовательно, на следующем (втором) ярусе имеется 4 вершины. Прочитав 4 символа, получаем концевую вершину на втором ярусе и 6 вершин на 3-м ярусе и т.д. На 6-м ярусе все вершины оказались концевыми, значит построение дерева завершено. Чтобы завершить описание кода, нужно указать, какая конкретно буква соответствует каждому кодовому слову. Для этого достаточно 8 бит на каждую букву. В результате затраты на информацию о коде составят

$$l_1 = 29 + 8 \times 15 = 149 \text{ бит.}$$

Передача всего файла потребует

$$l = l_1 + l_2 = 149 + 178 = 327 \text{ бит.} \tag{2.12}$$

Отметим, что без кодирования мы затратили бы 50×8=400 бит. Обращает на себя внимание тот факт, что "вспомогательная" информация (информация о коде) составляет почти половину длины кодовой последовательности. Это наводит на мысль о том, что с увеличением длины кодируемой последовательности доля служебной информации уменьшится, и эффективность кодирования существенно возрастет.

Затраты на служебную информацию можно уменьшить, если заметить, что для одного и того же распределения вероятностей можно построить много кодов Хаффмена. Все эти коды одинаково эффективны. Нет необходимости передавать точную структуру дерева. Нужно передать минимальные сведения, достаточные для построения одного из кодов Хаффмена.

Назовем *регулярным* неравномерный код в котором короткие кодовые слова лексикографически предшествуют более длинным. Регулярный код для рассматриваемого примера приведен в таблице 2.9, а его дерево – на рис. 2.13. Для описания регулярного кода достаточно указать число концевых вершин на каждом из ярусов с номерами $0,\ldots,l_{\max}$, где $l_{\max}$ – максимальная длина кодового слова. Подсчет числа бит на передачу дерева для регулярного кода иллюстрируется таблицей 2.7.

Таблица 2.5: Код Хаффмена для текста (2.11)

Буква	Число появлений	Длина кодового слова	Кодовое слово
I	1	6	010000
F	1	6	010001
_	12	2	00
W	5	3	100
E	4	4	0101
C	2	5	01001
A	4	4	1010
N	3	4	1011
O	5	3	110
T	1	6	011110
D	4	4	0110
S	3	4	1110
U	2	4	1111
L	2	5	01110
H	1	6	011111

Таблица 2.6: Регулярный код Хаффмена

Буква	Номер яруса (длина кодового слова)	Кодовое слово
_	2	00
O	3	010
W	3	011
A	4	1000
D	4	1001
E	4	1010
N	4	1011
S	4	1100
U	4	1101
C	5	11100
L	5	11101
F	6	111100
H	6	111101
I	6	111110
T	6	111111

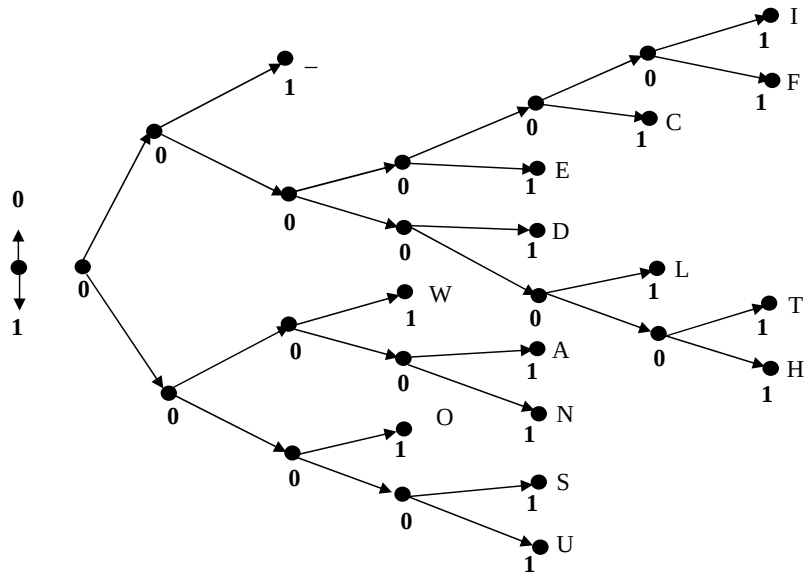


Рис. 2.12: Кодовое дерево кода Хаффмена для текста (2.11)

Таблица 2.7: Подсчет числа бит на передачу дерева регулярного кода

Ярус	Общее число вершин	Число концевых вершин n_i	Диапазон значений n_i	Затраты в битах
0	1	0	0...1	1
1	2	0	0...2	2
2	4	1	0...4	3
3	6	2	0...6	3
4	8	6	0...8	4
5	4	2	0...4	3
6	4	4	0...4	3
Всего				19

Лемма 2.7 *Полное кодовое дерево, имеющее M концевых вершин, имеет $M-1$ промежуточных вершин. Для полного описания дерева достаточно $2M-1$ бит.*

Доказательство. Первое из двух утверждений леммы легко доказать с помощью метода математической индукции. Мы предоставляем это сделать читателю в качестве упражнения. Для доказательства второго утверждения достаточно указать алгоритм построения описания дерева с помощью двоичной последовательности длины $2M-1$. Такой алгоритм был описан выше на примере кодирования текста (2.11). □

Оценку асимптотической эффективности двухпроходного кодирования сформулируем в виде следующей теоремы.

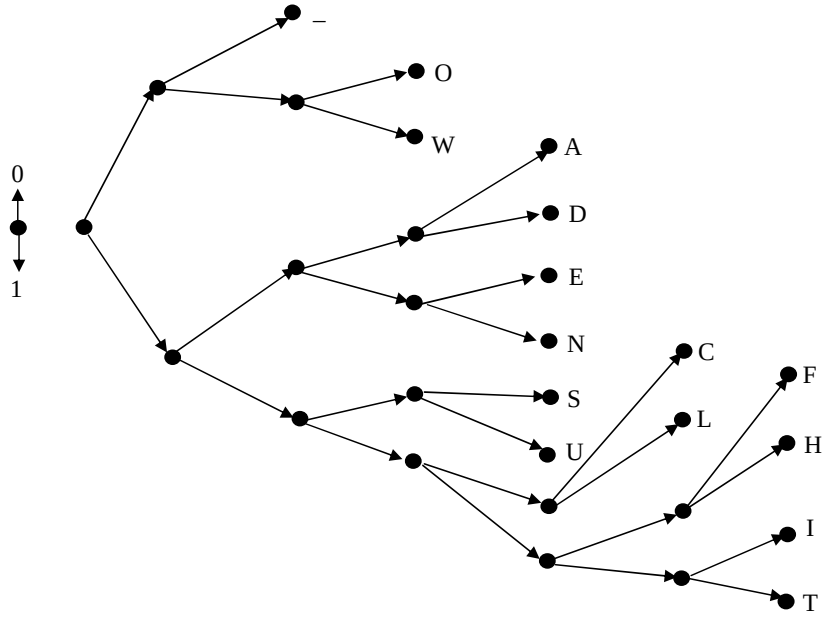


Рис. 2.13: Кодовое дерево для регулярного кода Хаффмена

Теорема 2.8 *При двухпроходном кодировании с использованием кода Хаффмена дискретного постоянного источника с объемом алфавита M и энтропией H средняя скорость кодирования удовлетворяет неравенству*

$$\bar{R} \leq H + 1 + \frac{1}{n}\left(M \log M + 3M - 1\right). \tag{2.13}$$

Подробное доказательство имеется в [12].

2.10 Нумерационное кодирование

Сейчас мы рассмотрим один из исторически первых методов универсального кодирования. Его автором является Фитингоф Б. М. [19].

Считаем, что алфавитом источника служит множество чисел $X = \{0, ..., M-1\}$. Пусть $\boldsymbol{x} = (x_1, ..., x_n)$ – последовательность на выходе источника. Рассмотрим следующий способ двухпроходного кодирования.

Кодовое слово нумерационного кода состоит из двух частей. Первая часть содержит закодированный равномерным кодом номер композиции $\boldsymbol{\tau}_n(\boldsymbol{x}) = (\tau_0(\boldsymbol{x}), ..., \tau_{M-1}(\boldsymbol{x}))$ в списке всех возможных композиций последовательностей длины n над алфавитом X. Вторая часть содержит закодированный равномерным кодом номер данной последовательности $\boldsymbol{x}$ в лексикографически упорядоченном списке последовательностей с одинаковой композицией $\boldsymbol{\tau}_n(\boldsymbol{x})$.

Оценим избыточность нумерационного кодирования Число различных композиций можно подсчитать по формуле ([12])

$$N_\tau(n, M) = \binom{n + M - 1}{M - 1} \tag{2.14}$$

и число последовательностей с фиксированной композицией $N(\boldsymbol{\tau})$ по формуле ([12])

$$N(\boldsymbol{\tau}) = \frac{n!}{\tau_0!...\tau_{M-1}!}. \tag{2.15}$$

Для заданной последовательности $\boldsymbol{x}$ через $l_1(\boldsymbol{x})$ и $l_2(\boldsymbol{x})$ обозначим длину первой и второй частей кодового слова. Из формул (2.14) и (2.15) получаем формулу для длины $l(\boldsymbol{x})$ кодового слова:

$$\begin{aligned} l(\boldsymbol{x}) &= l_1(\boldsymbol{x}) + l_2(\boldsymbol{x}) = \\ &= \lceil \log N_\tau(M) \rceil + \lceil \log N(\boldsymbol{\tau}) \rceil = \\ &= \left\lceil \log \binom{n + M - 1}{M - 1} \right\rceil + \left\lceil \log \frac{n!}{\prod_{i=0}^{M-1} \tau_i(\boldsymbol{x})!} \right\rceil . \end{aligned} \tag{2.16}$$

Обратимся к нашему примеру текста

```
IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN
```

Непосредственный подсчет по формулам (2.14) – (2.16) приводит к результатам

$$\begin{aligned} l_1 &= \left\lceil \log \binom{50 + 255}{255} \right\rceil = 190 \text{ бит}, \\ l_2 &= \left\lceil \log \left(\frac{50!}{12!5!^2 4!^3 3!^2 2!^3} \right) \right\rceil = 150 \text{ бит}. \end{aligned}$$

Сравнивая с предыдущим примером, видим, что l_1 неожиданно велико, а l_2 рекордно мало. Не может быть, чтобы не существовало более эффективного правила передачи композиции.

С другой стороны, формула (2.14) точна и, следовательно, уменьшение битовых затрат может быть получено только в том случае, если мы откажемся от равномерного кодирования номеров композиций. Для каких-то композиций мы должны тратить еще больше бит, чем раньше, и тогда, возможно, для некоторых композиций, кодирование станет значительно более эффективным. По-видимому, в "привилегированное" положение должны быть поставлены те композиции, для которых "сжатие" в

принципе осуществимо, т.е. композиции, содержащие сильно разнящиеся компоненты.

Мы будем отдельно передавать композицию, а затем указывать, какие буквы соответствуют различным компонентам композиции. Композицию перед передачей отсортируем. В данном примере сортированная композиция имеет вид $\tilde{\boldsymbol{\tau}} = (\tilde{\tau}_0, ..., \tilde{\tau}_{M-1})$=(12, 5, 5, 4, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1, 1, 0,. . . ., 0). Для элементов композиции имеют место неравенства

$$1 \le \tilde{\tau}_0 \le n, \quad \tilde{\tau}_{j+1} \le \tilde{\tau}_j, \quad j \ge 0.$$

Приписывая всем таким комбинациям одинаковые вероятности, с помощью арифметического кодирования можно передать номер композиции, затратив не более

$$\left\lceil \log \left(n \prod_{j:\tau_j>0} \tau_j \right) \right\rceil . \tag{2.17}$$

бит.

Осталось оценить затраты на передачу соответствия между буквами алфавита и компонентами композиции. Для одинаковых значений композиции порядок следования букв не существенен. Введем композицию сортированной композиции и обозначим ее $\boldsymbol{\tau}' = (\tau'_0, \tau'_1, ...)$. В данном примере $\boldsymbol{\tau}'$=(1,2,3,2,3,4,241). Достаточно передать, в какую из этих 7 категорий попала каждая буква алфавита. Для подсчета числа различных вариантов годится формула (2.15). Поэтому затраты на передачу композиции можно подсчитать как

$$l_1 = \left\lceil \log \left(n \prod_{j:\tau_j>0} \tau_j \right) \right\rceil + \left\lceil \log \left(\frac{M!}{\prod_j \tau'_j!} \right) \right\rceil = 25 + 108 = 133 \text{ бит}. \tag{2.18}$$

Окончательный результат для нумерационного кодирования составляет

$$l = l_1 + l_2 = 283 \text{ бит}$$

Оценим теперь эффективность нумерационного кодирования для произвольного дискретного постоянного источника с неизвестным распределением вероятностей. Нас интересует случай когда длина последовательности n достаточно велика и выполняется неравенство $n >> M$. Для оценки числа композиций вместо формулы (2.14) мы воспользуемся оценкой

$$N_\tau(n, M) \le (n+1)^{M-1} . \tag{2.19}$$

Она вытекает из того, что каждая из компонент принимает одно из $(n+1)$ значений от 0 до n и последняя компонента композиции однозначно вычисляется по первым $M-1$ компонентам. Для оценки числа последовательностей с заданной композицией воспользуемся (2.15). Получим

$$\begin{aligned} l(\boldsymbol{x}) &= l_1(\boldsymbol{x}) + l_2(\boldsymbol{x}) = \\ &= \lceil \log N_\tau(M) \rceil + \lceil \log N(\boldsymbol{\tau}) \rceil \leq \\ &\leq nH(\hat{\mathbf{p}}) - \frac{M-1}{2}\log(2\pi n) - \\ &\quad -\frac{1}{2}\sum_i \log(\hat{p}_i) + (M-1)\log(n+1) + 1. \end{aligned}$$

Теперь поделим обе части на n и усредним по всем последовательностям $\boldsymbol{x}$. Результат сформулируем в виде теоремы.

Теорема 2.9 *При нумерационном кодировании дискретного постоянного источника с объемом алфавита M и энтропией H средняя скорость кодирования удовлетворяет неравенству*

$$\bar{R} \leq H + \frac{M-1}{2}\frac{\log(n+1)+K}{n}, \tag{2.20}$$

где величина K не зависит от длины последовательности n.

Сравнивая результат с оценкой (2.13) средней скорости двухпроходного арифметического кодирования, убеждаемся, что при больших n избыточность нумерационного кодирования примерно в 2 раза меньше.

2.11 Адаптивное кодирование

Мы переходим к изучению универсального кодирования без задержки. Кодеру при поступлении на его вход очередного сообщения теперь недоступна информация о сообщениях, которые появятся в будущем. Поэтому способ кодирования текущего сообщения будет зависеть только от того, какими были предыдущие сообщения.

Естественным решением задачи является следующий подход. Кодер (и декодер) по последовательности уже переданных сообщений оценивают вероятности возможных значений следующего сообщения и строят

для него код в соответствии с этими оценками вероятностей. Если источник стационарен, то с увеличением длины уже закодированной последовательности оценки вероятностей будут все более точными и кодирование – все более эффективным. У декодера не возникнет проблем с восстановлением данных, поскольку он располагает всей информацией, использованной кодером для построения кода.

В рамках этой общей схемы есть несколько степеней свободы выбора конкретной модификации алгоритма. В частности, можно менять следующие характеристики алгоритма:

- длину последовательности сообщений, по которой вычисляются оценки вероятностей;
- формулы для вычисления оценок вероятностей;
- способ кодирования при известных оценках вероятностей.

Обсудим эти возможности. Начнем с вопроса о выборе длины "окна", т.е. длины последовательности, по которой оцениваются вероятности сообщений. Интуитивно ясно, что увеличение длины окна должно приводить к повышению точности оценок, и, как следствие, к повышению эффективности кодирования. Это соображение верно, если верна гипотеза о стационарности источника. В противном случае выбор короткого окна может оказаться более удачным, поскольку позволит кодеру и декодеру быстрее адаптироваться к изменениям статистических свойств источника. Более детальный анализ показывает, что и для стационарного источника достаточно выбирать окно с длиной в несколько раз превышающей объем алфавита источника. Дальнейшее увеличение длины окна не приведет заметному уменьшению избыточности. Исходя из этого, кажется целесообразным всегда выбирать окна конечной длины.

Однако, на пути реализации такого подхода имеется "подводный камень", это относительно высокая сложность кодирования. Дело в том, что при "бесконечном окне" (когда окном служит вся предшествующая последовательность сообщений) вся необходимая информация о предшествующих сообщениях хранится в виде счетчиков числа появления различных букв (число счетчиков равно объему алфавита источника). При окне конечной длины при поступлении от источника новой буквы нужно не только нарастить на 1 значение соответствующего счетчика, но и уменьшить на 1 значение счетчика для той буквы, которая в данный момент времени оказалась за пределами окна. Для этого придется помнить всю последовательность букв, хранящихся в окне. С учетом этого обстоятельства обычно вероятности оцениваются по всей предшествующей

последовательности. (В действительности, в практических схемах сжатия применяют некоторые ухищрения, позволяющие обеспечить быструю адаптацию оценок вероятностей без хранения "окна". Описание этих схем выходит за рамки данного пособия).

Выбор формул для оценивания вероятностей и выбор способа кодирования взаимосвязаны. На первый взгляд, оценка вероятности того, что $x_{n+1} = a$, должна вычисляться по последовательности $x_1, ..., x_n$ по формуле

$$\hat{p}_n(a) = \frac{\tau_n(a)}{n}, \tag{2.21}$$

где через $\tau_n(a)$ обозначено число появлений буквы a в последовательности длины n. Однако, при использовании этой формулы для некоторых букв оценка вероятности окажется равной нулю. Это не вызовет серьезных проблем, если для кодирования используется, например, код Хаффмена. Однако при использовании кодов Шеннона, Гилберта-Мура или арифметического кодирования нулевые значения вероятностей недопустимы.

В литературе по кодированию источников можно найти много способов записи оценок вероятностей букв (см., например, [11]). Например, можно воспользоваться формулой

$$\hat{p}_n(a) = \frac{\tau_n(a) + 1}{n + M}. \tag{2.22}$$

Смысл этого приближения состоит в том, что в числителе формулы добавлена 1 к счетчику числа появлений букв и тем самым мы исключили нулевые вероятности. К знаменателю пришлось добавить объем алфавита, иначе нарушилось бы условие нормировки вероятностей. При увеличении длины n различие между формулами (2.21) и (2.22) становится несущественным, их можно считать асимптотически эквивалентными. Однако, при длине n сопоставимой с объемом алфавита M оценки, подсчитанные по формуле (2.22) будут значительно меньше несмещенных оценок (2.21), и поэтому затраты на кодирование букв (минус логарифмы вероятностей букв) будут заметно больше. Отсюда следует, что, по крайней мере, для коротких последовательностей, формула (2.22) неэффективна.

Пример 2.11.1 Кодируем последовательность

`IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN`

Чтобы определить длину кодового слова, нужно вычислить число G, представляющее собой оценку вероятности последовательности. До начала кодирования $G = 1$. После кодирования буквы "I" в соответствии с (2.22) $G = 1/256$ и т. д., в результате шаг за шагом получим

$$G = 1 \cdot \frac{1}{256} \cdot \frac{1}{257} \cdot \frac{1}{258} \cdot \frac{1}{259} \cdot \frac{1}{260} \cdot \frac{2}{261} \cdot \ldots =$$

$$= \frac{12!(5!)^2(4!)^3(3!)^2(2!)^3}{256 \cdot \ldots \cdot 305}.$$

Длина кодового слова равна

$$l = \lceil -\log G \rceil + 1 = 342 \text{ бит}. \tag{2.23}$$

Вывод нижней границы избыточности подсказывает, что, возможно, смещение оценки в (2.22) выбрано не лучшим способом. Попробуем подставить смещение $1/2$. Получим

$$\hat{p}_n(a) = \frac{\tau_n(a) + 1/2}{n + M/2} = \frac{2\tau_n(a) + 1}{2n + M}. \tag{2.24}$$

Пример 2.11.2 Для той же последовательности входных данных, что и в предыдущем примере, с использованием оценки (2.24) получим

$$G = 1 \cdot \frac{1}{256} \cdot \frac{1}{258} \cdot \frac{1}{260} \cdot \frac{1}{262} \cdot \frac{1}{264} \cdot \frac{3}{266} \cdot \ldots =$$

$$= \frac{23!!(9!!)^2(7!!)^3(5!!)^2(3!!)^3}{256 \cdot \ldots \cdot 305}.$$

Здесь и дальше используются обозначения

$$(2n-1)!! = 1 \times 3 \times \ldots \times (2n-1),$$

$$(2n)!! = 2 \times 4 \times \ldots \times (2n).$$

Новая длина кодового слова равна

$$l = \lceil -\log G \rceil + 1 = 323 \text{ бит}. \tag{2.25}$$

Простое усовершенствование формулы дало заметный выигрыш в эффективности кодирования. Еще более практичен подход, основанный на введении в алфавит дополнительной буквы, которую мы назовем *esc*-символом. Название буквы, как мы увидим позже, вполне соответствует способу ее применения.

Тем буквам алфавита, которые уже встречались в последовательности из n предшествующих букв, приписываются вероятности

$$\hat{p}_n(a) = \frac{\tau_n(a)}{n+1}, \quad \tau_n(a) > 0.$$

Если же в момент времени $n+1$ появилась буква, которой ранее ни разу не было, то передается *esc*-символ, которому приписывается вероятность

$$\hat{p}_n(esc) = \frac{1}{n+1},$$

а затем передается сама буква, при этом всем буквам, которые не встречались, приписываются одинаковые вероятности.

Эту стратегию кодирования (назовем ее *A-алгоритмом*) можно подытожить в виде следующей формулы

$$\hat{p}_n(a) = \begin{cases} \frac{\tau_n(a)}{n+1}, & \text{если } \tau_n(a) > 0; \\ \frac{1}{n+1}\frac{1}{M-M_n}, & \text{если } \tau_n(a) = 0, \end{cases} \tag{2.26}$$

где M_n обозначает число различных букв, содержащихся в последовательности длины n. Подход, основанный на использовании *esc*-символа, мы исследуем далее на примере, а затем оценим его эффективность применительно к произвольному дискретному постоянному источнику.

Пример 2.11.3 *(Алгоритм A).* Снова кодируем последовательность

`IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN`

После выполнения первых шагов кодирования в соответствии с (2.26) имеем

$$G = 1 \cdot \frac{1}{2} \cdot \frac{1}{256} \cdot \frac{1}{3} \cdot \frac{1}{255} \ldots$$

Заметим, что следующему появлению пробела будет соответствовать сомножитель $1/6$, третьему пробелу – сомножитель $2/13, \ldots$, последнему – $11/47$. Получим

$$G = \frac{11!(4!)^2(3!)^3(2!)^2}{50!} \cdot \frac{1}{256 \cdot 255 \cdot \ldots \cdot 242}.$$

Следовательно, длина кодового слова равна

$$l = \lceil -\log G \rceil + 1 = 291 \text{ } бит. \tag{2.27}$$

Результат получился почти такой же, как при нумерационном кодировании, хотя данный алгоритм является однопроходным. Таким образом, возможность "заглянуть в будущее" в данном примере не дает большого выигрыша в кодировании.

Вычисления, выполненные в последнем примере, легко обобщаются на случай произвольной последовательности $\boldsymbol{x} = (x_1, ..., x_n)$ с композицией $(\tau_1, ..., \tau_M)$. Получим

$$\begin{aligned} G &= \frac{\prod_{i=1}^{M_n} (\tau_i - 1)!}{n!} \cdot \frac{(M - M_n)!}{M!} = \\ &= \frac{\prod_{i=1}^{M_n} \tau_i!}{n!} \cdot \frac{(M - M_n)!}{M! \prod_{i=1}^{M_n} \tau_i} \geq \\ &\geq \frac{\prod_{i=1}^{M_n} \tau_i!}{n!} M^{-M} (n+1)^{-M}. \end{aligned}$$

Применим приведенные выше асимптотические оценки полиномиальных коэффициентов. Повторив выкладки, использованные при выводе (2.20), получим следующий результат.

Теорема 2.10 *При адаптивном арифметическом кодировании дискретного постоянного источника с объемом алфавита M и энтропией H средняя скорость кодирования удовлетворяет неравенству*

$$\bar{R} \leq H + \frac{M}{2} \frac{\log(n+1) + K}{n}, \tag{2.28}$$

где величина K не зависит от длины последовательности n.

Заметим, что асимптотические оценки эффективности адаптивного кодирования получаются примерно такими же, что и при нумерационном кодировании. Это вполне согласуется с цифрами, полученными в примере 2.11.3 для короткой последовательности на выходе источника.

Отметим, что алгоритм A (формула (2.26)) – простой для понимания и анализа алгоритм, но не самый лучший. Действительно, например, после кодирования первой буквы источника на втором шаге мы имеем одинаковые вероятности *esc*-символа и единственного известного кодеру и декодеру первого символа. Это не совсем справедливо, поскольку

с *esc*-символом ассоциируются все остальные буквы алфавита. Вероятно, кодирование станет более эффективным, если на первых шагах *esc*-символ будет иметь больший вес, но с течением времени приписываемая ему вероятность будет уменьшаться.

Рассмотрим альтернативную оценку

$$\hat{p}_n(a) = \begin{cases} \frac{\tau_n(a)-1/2}{n}, & \text{если } \tau_n(a) > 0; \\ \frac{M_n}{2n}\frac{1}{M-M_n}, & \text{если } \tau_n(a) = 0. \end{cases} \tag{2.29}$$

Кодирование с использованием этой оценки назовем *D-алгоритмом*. Легко убедиться, что условие нормировки выполняется и что по сравнению с A-алгоритмом заметно увеличена вероятность, приписываемая новым символам, по крайней мере, на первых шагах кодирования. Проверим эффективность D-алгоритма на примере.

Пример 2.11.4 *(Алгоритм D).* Кодируем ту же последовательность. После нескольких первых шагов имеем

$$G = 1 \cdot \frac{1}{2} \cdot \frac{1}{256} \cdot \frac{2}{4} \cdot \frac{1}{255} \cdot \frac{3}{6} \cdot \frac{1}{254}\frac{4}{8} \cdot \frac{1}{253}\frac{1}{10} \cdot \frac{5}{12} \cdot \frac{1}{252}\ldots$$

Заметим, что в знаменателе будем иметь произведение двух арифметических прогрессий:

$$(2 \times 4 \times \ldots \times 100) \times (256 \times 255 \times \ldots \times 242).$$

В числителе *esc*-символы породят произведение $1{\times}2{\times}3{\times}\ldots{\times}14$, а каждая буква, появившаяся τ раз, вызовет появление произведения $1{\times}3{\times}5{\times}\ldots{\times}(2\tau-3)$. В целом для всей последовательности получим

$$G = \frac{(2\times 12-3)!!((2\times 5-3)!!)^2((2\times 4-3)!!)^3((2\times 3-3)!!)^2}{100!!}\times$$

$$\times\frac{14!}{256\cdot 255\cdot\ldots\cdot 242}.$$

Следовательно, длина кодового слова равна

$$l = \lceil -\log G\rceil + 1 = 283 \quad \textit{бит}.$$

Результат получился лучше, чем при кодировании по алгоритму A и в точности такой же, как и при нумерационном кодировании. Таким образом, снова убеждаемся, что возможность “заглянуть в будущее” не всегда дает выигрыш при кодировании.

В завершение параграфа отметим, что формулы алгоритмов A и D совпадают с формулами для оценок условных вероятностей символов при заданном контексте, применяемыми в алгоритмах универсального кодирования PPMA [36] и PPMD [50] соответственно. Алгоритмы PPM будут рассмотрены ниже.

2.12 Сравнение алгоритмов

Для удобства сравнения алгоритмов их характеристики сведены в таблицу 2.8. В этой таблице в формулах для асимптотической избыточности через n обозначена длина последовательности, M– объем алфавита, а величины K_i, $i = 1, ..., 5$ представляют собой константы, независящие от n.

Полученные результаты показывают, что, если не принимать во внимание сложность алгоритмов, предпочтения заслуживают нумерационное кодирование и адаптивное арифметическое кодирование. Неожиданными кажется тот факт, что возможность выполнения двух проходов при кодировании почти не сказывается на достижимом сжатии. Причина в том, что адаптивное кодирование обеспечивает практически оптимальное кодирование (его избыточность блика к нижней границе), поэтому выигрыш в принципе большим быть не может.

Таблица 2.8: Сравнение алгоритмов универсального кодирования

Алгоритм	Число проходов	Асимптотическая избыточность	Длина слова для текста (2.11)
Двухпроходное кодирование, код Хаффмена	2	$1 + K_1/n$	302
Нумерационное кодирование	2	$\frac{M \log n + K_3}{2n}$	283
Адаптивное кодирование (алгоритм A)	1	$\frac{M \log n + K_4}{2n}$	291
Адаптивное кодирование (алгоритм D)	1	$\frac{M \log n + K_5}{2n}$	283

2.13 Алгоритмы кодирования источников, применяемые в архиваторах

В разделе 2 мы познакомились с методами кодирования при известной статистике источника. Было показано, что, применяя арифметическое кодирование, можно получить скорость кодирования сколь угодно близкую к нижнему пределу – к скорости создания информации источником. Далее в разделе 3 мы получили такой же оптимистический результат и для случая, когда статистические характеристики источника неизвестны. Понятно, что методы раздела 3 можно применять к последовательностям букв (вместо отдельных букв) и тем самым добиться скорости кодирования сколь угодно близкой к энтропии на сообщение. Другой очевидный способ развития этих методов – аппроксимация модели источника цепью Маркова достаточно высокого порядка. Оба пути оказываются непрактичными. Во-первых, их сложность быстро растет с увеличением длины блока или порядка аппроксимирующей цепи Маркова. Во-вторых, при кодировании последовательностей конечной длины, в частности, при кодировании типичных файлов пользователей персональных компьютеров, асимптотически оптимальные методы не всегда дают наилучший результат. Точнее говоря, различные способы кодирования, имея асимптотически эквивалентные характеристики, могут существенно различаться по эффективности при использовании в реальных условиях.

В данном разделе мы опишем наиболее эффективные способы универсального кодирования источников. Наиболее известные и широко применяемые – методы Зива-Лемпела. Их описанию предшествует описание так называемых “монотонных” кодов, метода интервального кодирования и метода “стопка книг”. Эти способы кодирования входят как составная часть в более сложные алгоритмы кодирования. Помимо этого они имеют самостоятельное значение, в частности, они часто используются при кодировании видеоинформации.

Далее мы приводим описание алгоритмов-рекордсменов по сжатию – алгоритма предсказания по частичному совпадению (PPM) и алгоритма Барроуза-Уилера. В завершение раздела мы обсудим критерии сравнения эффективности архиваторов и приведем характеристики лучших из них.

Более подробное описание алгоритмов и примеры их применения даны в [12].

Точный математический анализ практических алгоритмов кодирования сложен и выходит за рамки данного курса. Однако, как мы увидим из описания алгоритмов, они являются естественным развитием асимп-

тотически эффективных теоретико-информационных методов, описанных в предыдущих разделах.

2.14 Монотонные коды

Все рассмотренные в разделе ?? универсальные кодеры, однопроходные и двухпроходные, используют при кодировании полученные тем или иным способом оценки вероятностей букв источника. Если объем алфавита очень велик (например, если в качестве алфавита рассматриваются не отдельные буквы, а пары, тройки букв), то хранение и модификация информации о каждой букве потребует значительных затрат памяти и времени. В этих случаях достаточно эффективными оказываются очень простые методы универсального кодирования, которым посвящен данный параграф.

Последовательность букв источника будет преобразовываться в последовательность натуральных чисел, поэтому нам понадобятся префиксные коды бесконечного объема. Префиксный код множества натуральных чисел N=$\{1,2,\dots\}$ мы будем называть *монотонным кодом,* если для любых $i, j \in \mathrm{N}$, $i < j$, длины соответствующих кодовых слов l_i и l_j удовлетворяют неравенству $l_i \le l_j$.

Приведем несколько примеров монотонных кодов. *Унарный код.* Напомним, что запись вида 0^m или 1^m означает соответственно серию из m нулей или единиц. Унарный код сопоставляет числу i двоичную комбинацию вида $1^{i-1}0$. Например, унарными кодами чисел 1, 2 и 3 являются последовательности unar(1)=0, unar(2)=10 и unar(3)=110 соответственно. Длина кодового слова для числа i равна $l_i = i$.

Нетрудно проверить, что унарный код оптимален (минимизирует среднюю длину кодовых слов), если числа i распределены по геометрическому закону

$$p_i = (1-\alpha)\alpha^{i-1}, \quad i = 1, 2, ...$$

с параметром $\alpha = 1/2$, т.е. при $p_i = 2^{-i}$, $i = 1, 2, ...$ Для значений $\alpha > 1/2$ более эффективен код Голомба.

Код Голомба [48]. Введем параметр $T = 2^m$. Код Голомба для числа i состоит из двух частей. Первая часть – унарный код числа $\lfloor i/T \rfloor + 1$, вторая часть – двоичная запись в виде последовательности длины m остатка от деления i на T. Очевидно, длина кода Голомба для числа i равна $l_i = \lfloor i/T \rfloor + 1 + m$. Например, при $m = 3$ и $i = 21$ имеем

$\lfloor i/T \rfloor + 1 = 3$, остаток равен 5. Поэтому кодом Голомба числа 21 будет последовательность (110)(101)=110101.

В технической литературе код Голомба иногда называют кодом Райса.

Код Галлагера-ВанВухриса [46] – естественное обобщение кода Голомба на случай, когда параметр T не является степенью двойки. Как и в коде Голомба, префикс – закодированное унарным кодом число $\lfloor i/T \rfloor + 1$. Далее следует двоичный код остатка. Если $\log T$ – не целое число, то для представления некоторых (меньших) значений остатка от деления i на T используется $m = \lfloor \log T \rfloor$ бит, для остальных (больших) значений используется последовательность из $m + 1$ бит. Точнее говоря, остаток кодируется кодом Хаффмена в предположении, что все значения остатка равновероятны. Например, при $T = 5$ кодовыми словами для значений остатков 0, 1, 2, 3 и 4 будут последовательности 00, 01, 10, 110 и 111 соответственно. В частности, при $T = 5$ числу 21 соответствует кодовое слово (1110)(01=111001), а числу 24 – слово (1110)(111)=1110111.

Оптимальное значение параметра T для кода Галлагера-ВанВухриса связано с параметром геометрического распределения α соотношением

$$\alpha^T + \alpha^{T+1} \leq 1 < \alpha^{T-1} + \alpha^T. \tag{2.30}$$

Благодаря чрезвычайно простой схеме кодирования и достаточно высокой эффективности коды Голомба и Галлагера-ВанВухриса часто применяют при кодировании аудио и видеосигналов. Для применения в схемах универсального кодирования они не очень хороши. Во-первых, нужно знать (или оценивать) значение параметра T, во-вторых, длина кодового слова l_i, как и для унарного кода, линейно растет с увеличением числа i и для некоторых распределений вероятностей букв источника кодирование может быть неэффективным.

Код Левенштейна. [14] Этот код проще всего объяснить на конкретном примере. Предположим, что нужно передать число $i = 21$. Двоичное представление этого числа имеет вид 10101. Непосредственно использовать при кодировании двоичные представления натуральных чисел нельзя, такой код не будет префиксным. Самый простой выход состоит в том, чтобы приписать в начале слова префикс, указывающий длину двоичной записи числа (в данном случае это число 5). Если это число закодировать префиксным, например, унарным, кодом и приписать слева к двоичной записи числа, то код получится однозначно декодируемым. В данном примере для числа 21 получим кодовое слово (11110)(10101)=1111010101. В общем случае длина двоичного представления будет равна $2 \lceil \log i \rceil$.

Будем шаг за шагом улучшать этот способ кодирования. Заметим,

что первая значащая цифра двоичной записи числа – всегда 1. Ее можно не передавать, декодер сам допишет недостающую единицу, если будет знать длину двоичной записи числа. Обозначим через $\mathrm{bin}'(i)$ двоичную запись натурального числа i без первой единицы, прямыми скобками обозначается длина двоичной последовательности. Итак, простой монотонный код числа i имеет следующую структуру

$$\mathrm{mon}(i) = \left(\mathrm{unar}\left(\left|\mathrm{bin}'(i)\right| + 1\right), \mathrm{bin}'(i)\right). \tag{2.31}$$

Например, mon(21)=(unar(5),0101)=111100101.

Длины кодовых слов этого монотонного кода равны

$$l_i = 2\lfloor \log i \rfloor + 1. \tag{2.32}$$

Чтобы сделать запись еще короче, с длиной двоичной записи можно поступить так же, как и с самим числом, т.е. передать его значащие разряды (кроме первой единицы), затем длину двоичной записи числа значащих разрядов и т.д. Итерации продолжаются, пока не останется один значащий разряд. Чтобы декодирование было однозначным, достаточно приписать префикс, содержащий закодированное префиксным кодом число итераций. Заметим, что минимальное число итераций равно 0 (при кодировании числа 1). Поэтому в качестве префиксного кода можно выбрать унарный код увеличенного на 1 числа итераций. Полученное кодовое слово будет кодовым словом *кода Левенштейна.*

Например, для числа 21 вычисляется $\mathrm{bin}'(21)$=0101, затем $\mathrm{bin}'(4)$=00, $\mathrm{bin}'(2)$=0. Число итераций равно 3, поэтому кодовое слово кода Левенштейна имеет вид

$$\mathrm{lev}(21) = (1110)(0)(00)(0101) = 11100000101.$$

Декодер кода Левенштейна, декодируя унарный код, узнает, что итераций было 3. Прочитав один значащий разряд (0) и дописав к нему в начало 1, получает последовательность 10. Это означает, что на предпоследней итерации длина числа была 2. Прочитав 2 разряда и дописав слева 1, получает 100. Теперь декодер считывает 4 разряда и дописывает слева 1. Получается последовательность 10101, ей соответствует число 21. Поскольку это уже последняя 3-я итерация, число 21 является результатом декодирования.

В таблице 2.9 приведены кодовые слова, полученные по правилу (2.31) и кодовые слова кода Левенштейна для некоторых чисел натурального ряда и длины кодовых слов.

Упрощенный код Левенштейна (Код Элайеса).

Мы опишем более простой код, приведенный в работе Элайеса [40]. Числу $i = 1$ припишем кодовое слово elias(1)=0. Для чисел $i > 1$ кодовые слова вычисляются по следующем правилу:

$$\text{elias}(i)=(\text{unar}(\ |\text{bin'}(|\text{bin'}(i)|)|+2\ )\ ,\ \text{bin'}(|\text{bin'}(i)|),\ \text{bin'}(i)\ ) \quad (4.1.4)$$

То есть, кодовое слово состоит из 3 частей. Справа (в третьей части) записываем двоичное представление числа без первой единицы. Ей предшествует вторая часть, в которой пишется двоичное представление длины третьей части, тоже без первой единицы. В первой части записан унарный код увеличенной на 2 длины второй части кодового слова.

Например, для числа 21 кодовое слово имеет вид

$$\text{elias}(21) = (1110)(00)(0101) = 1110000101.$$

Длина кодового слова кода Элайеса для произвольного числа i равна

$$l_i = \begin{cases} 1, & i = 1, \\ \lfloor \log i \rfloor + 2 \lfloor \log \lfloor \log i \rfloor \rfloor + 2, & i > 1, \end{cases} \quad (2.33)$$

и, следовательно, удовлетворяет неравенству

$$l_i \leq \log i + 2 \log(1 + \log i) + 2, \quad i = 1, 2, \ldots \quad . \quad (2.34)$$

Во втором слагаемом добавлена 1 под знаком логарифма для того, чтобы придать смысл этому выражению при $i = 1$.

Для того чтобы численно сравнить три монотонных кода, обратимся к Таблице 2.9. Как ни странно, простейший код (2.31) в большинстве случаев дает наилучший результат, но для этого кода длина кодовых слов растет с увеличением i быстрее, чем для двух других кодов. Это видно, в частности, из сравнения формул (2.32) и (2.33). Коды Левенштейна и Элайеса практически эквивалентны, выигрыш кода Левенштейна проявляется только при астрономически больших значениях i.

Следующая теорема проясняет роль монотонных кодов в универсальном кодировании источников.

Теорема 2.11 *Пусть случайная величина i принимает значения из множества чисел натурального ряда и распределение вероятностей случайной величины удовлетворяет условию: $p_i \leq p_j$ если $i > j$. Тогда при использовании кода Элайеса средняя длина кодовых слов $\bar{l}$ удовлетворяет неравенству*

$$\bar{l} \leq H(1 + o(H)),$$

где через H обозначена энтропия случайной величины i, и $o(H) \to 0$ при $H \to \infty$.

Таблица 2.9: Таблица монотонных кодов некоторых чисел

i	Монотонный код (2.31)		Код Левенштейна		Код Элайеса	
	$\mathrm{mon}(i)$	l_i	$\mathrm{lev}(i)$	l_i	$\mathrm{elias}(i)$	l_i
1	0	1	0	1	0	1
2	100	3	100	3	100	3
3	101	3	101	3	101	3
4	11000	5	110000	6	110000	6
...	...	5	...	6	...	6
7	11011	5	110011	6	110011	6
8	1110000	7	1101000	7	1101000	7
...	...	7	...	7	...	7
15	1110111	7	1101111	7	1101111	7
16	111100000	9	11100000000	11	1110000000	10
...	...	9	...	11	...	10
31	111101111	9	11100001111	11	1110001111	10
32	$1^5 00^5$	11	111000100000	12	11100100000	11
...	...	...	...	12	...	11
63	$1^5 01^5$	11	111000111111	12	11100111111	11
2^{15}	$1^{15} 00^{15}$	31	$1^4 01111110^{15}$	26	$1^5 011110^{15}$	25
2^{1023}	$1^{1023} 00^{1023}$	2047	$1^4 010011^9 0^{1023}$	1041	$1^{10} 01^9 10^{1023}$	1043

Доказательство. Прежде всего, из упорядоченности вероятностей по убыванию и условия нормировки вероятностей вытекают неравенства

$$p_i \le \frac{1}{i} \quad , \quad i \le \frac{1}{p_i}, \quad i = 1, 2, \ldots \quad . \tag{2.35}$$

Из (2.34) имеем

$$\begin{aligned} \bar{l} &= \sum_{i=1}^{\infty} l_i p_i \le \\ &\le \sum_{i=1}^{\infty} p_i (\log i + 2 \log\log i + 2) \le \\ &\le \sum_{i=1}^{\infty} p_i \log \frac{1}{p_i} + 2 \sum_{i=1}^{\infty} p_i \log \left(1 + \log \frac{1}{p_i}\right) + 2 \le \\ &\le H + 2 \log(1 + H) + 2. \end{aligned} \tag{2.36}$$

Здесь первое неравенство основано на (2.34), второе использует (2.35). Последнее неравенство основано на выпуклости $\cap$ функции $\log x$. Из (2.36) следует утверждение теоремы. $\square$

Теорема 2.11 утверждает, что монотонный код с длиной i-го кодового слова порядка $\log i + 2 \log\log i$ обеспечивает достаточно эффективное кодирование для любого источника с монотонно убывающими вероятностями букв. Поскольку буквы всегда можно перенумеровать, это означает, что при кодировании можно не знать вероятности букв, достаточно знать, как упорядочены вероятности букв.

Отметим, что за отсутствие точной информации о вероятностях букв мы платим избыточностью кодирования. При точно известных вероятностях побуквенное кодирование имеет избыточность не больше 1. Избыточность универсального монотонного кода имеет порядок $2 \log H + 2$, что может быть приемлемым только при очень больших значениях энтропии источника H.

Отметим также простоту реализации кодирования. В отличие, например, от кода Хаффмена, нет необходимости в хранении таблиц кодовых слов, кодирование и декодирование сводятся к вычислениям по приведенным выше достаточно простым формулам.

2.15 Интервальное кодирование и метод "стопка книг"

В этом параграфе мы рассмотрим два способа взаимно однозначного преобразования последовательности букв источника в последовательность чисел натурального ряда. При определенных условиях (например, при сильной зависимости букв) побуквенное кодирование преобразованной последовательности оказывается заметно более эффективным, чем кодирование исходной последовательности источника.

Начнем с *интервального кодирования.*

Пусть $X = \{0, 1, ..., M-1\}$ – алфавит источника и на выходе источника наблюдается последовательность $x_1, x_2, ...$ Алгоритм описывается рекуррентно. Предположим, что начальная часть последовательности $\boldsymbol{x}_1^{n-1} = (x_1, ..., x_{n-1})$ уже закодирована и передана и, следовательно, известна декодеру. Вместо буквы x_n передается длина интервала (количество букв) между предыдущим и текущим появлением данной буквы. Иными словами, кодер вычисляет и передает минимальное число $r_n = r$ такое, что $x_{n-r} = x_n$. Тем самым исходная последовательность $x_1, x_2, ...$ преобразуется в последовательность чисел $r_1, r_2, ...$

Для завершения описания алгоритма нужно определить правило вычисления интервалов для тех букв, которые не встречались в последовательности $\boldsymbol{x}_1^{n-1}$. Проще всего условиться о том, что первой передаваемой букве предшествовали все буквы алфавита в заранее согласованном (для определенности, в алфавитном) порядке.

Опишем *метод "стопка книг".*

Этот метод имеет несколько названий и переоткрывался разными авторами. По-видимому, первым его описал и проанализировал его эффективность Б. Я. Рябко [16]. Заметно позже в точности тот же алгоритм описан в [27] под названием "move-to-front coding". Именно под таким названием он чаще всего упоминается в технической литературе. Еще раз он открыт в [40] и назван "recency rank coding". Это последнее название, которое можно перевести как "кодирование степени новизны", наиболее точно отражает суть дела.

Как и при кодировании длин интервалов, предположим, что начальная часть последовательности $\boldsymbol{x}_1^{n-1} = (x_1, ..., x_{n-1})$ уже закодирована и передана и, следовательно, известна декодеру. Но теперь вместо буквы x_n передается количество *различных* букв между предыдущим и текущим появлением данной буквы. Иными словами, кодер вычисляет минимальное число $r_n = r$ такое, что $x_{n-r} = x_n$. Затем вычисляется число d_n различных букв в последовательности $\boldsymbol{x}_{n-r+1}^{n-1}$. Тем самым исходная

последовательность $x_1, x_2, ...$ преобразуется в последовательность чисел $d_1, d_2, ...$

Для передачи букв, которые не встречались в $\boldsymbol{x}_1^{n-1}$, можно использовать стратегию, описанную выше для интервального кодирования.

Пример 2.15.1 Пусть источник с алфавитом $X = \{a, b, c\}$ породил последовательность *cabbbabbac*. Рассмотрим работу двух методов кодирования в предположении, что по умолчанию кодеры (и декодеры) считают, что этой последовательности предшествовала последовательность *abc*. Тогда при интервальном кодировании будет получена последовательность 0,3,3,0,0,3,1,0,2,9. При кодировании с помощью стопки книг получим последовательность 0,2,2,0,0,1,1,0,1,2.

Поясним происхождение названия "стопка книг". Предположим, что студент, готовясь к экзамену, сложил книги стопкой на столе. Всякий раз, выбирая из стопки нужную книгу, после использования, он не возвращает книгу на свое место, а кладет ее сверху. Верхняя книга теперь имеет наименьший номер, а номера книг, которые раньше находились выше нее, увеличились на 1. Сколько книг находятся выше данной? Ровно столько, сколько различных книг использовалось между текущим и предыдущим использованием данной книги. Таким образом, последовательность $d_1, d_2, ...$ это как раз и есть последовательность номеров книг, извлекаемых из стопки.

Пример 2.15.2 Применим "стопку книг" к пословице

```
IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN
```

Для этого вместо букв нужно подставить их номера в таблице ASCII-кода. В частности, пробелу соответствует число 32, а буквам A ... Z – числа 65 ... 90. Результатом будет последовательность чисел
74, 72, 35, 88, 73, 3, 72, 71, 80, 1, 81, 86, 6, 76, 4, 3, 6, 86, 3, 10, 10, 3, 3, 6, 87, 83, 9, 6, 6, 7, 3, 8, 81, 9, 9, 9, 9, 7, 2, 5, 3, 10, 8, 3, 10, 10, 3, 13, 6, 13
Применим универсальный монотонный код, описываемый формулой (2.31). Затраты на передачу последовательности составят 332 бита.

С одной стороны, мы получили заметное сжатие по сравнению с исходными затратами, составляющими 50×8=400 бит. С другой стороны, сравнивая этот результат с другими, приведенными в таблице 2.8, видим, что данный способ кодирования несколько проигрывает тем, которые рассматривались выше. Этот проигрыш – адекватная плата за простоту реализации.

Нетрудно понять, почему описанные методы преобразования потоков данных могут быть полезны при универсальном кодировании. Чем больше вероятность буквы, тем короче интервалы между ее появлениями. После преобразования последовательности источника в последовательность интервалов или последовательность номеров в стопке книг наиболее вероятными будут малые числа и менее вероятными – большие. Тем самым создаются предпосылки для успешного применения монотонных кодов, описанных в предыдущем параграфе. Из описания двух алгоритмов следует, что с точки зрения применения монотонных кодов метод стопки книг предпочтительнее, поскольку порождает числа, меньшие или равные числам, порождаемым интервальным кодером.

Ниже мы приводим верхнюю оценку средней скорости интервального кодирования, поскольку его анализ существенно проще. Понятно, что эта же граница верна и для сжатия стопкой книг.

Теорема 2.12 *Пусть $H(X)$ – энтропия одномерного распределения дискретного стационарного источника. Средняя скорость интервального кодирования в сочетании с использованием кода Элайеса удовлетворяет неравенству*

$$\bar{R} \le H(X)(1 + o(H(X)),$$

где $o(H) \to 0$ при $H \to \infty$.

Доказательство. Обозначим через $r_i(a)$ длину интервала между i-м и $(i-1)$-м появлениями буквы $x = a$ и через $l_i(a)$ соответствующие затраты на передачу буквы. В соответствии с оценкой (2.34)

$$l_i(a) \le \log r_i(a) + 2\log(1 + \log r_i(a)) + 2. \tag{2.37}$$

Обозначим через $\bar{l}(a)$ и $\bar{r}(a)$ соответственно средние (по множеству последовательностей источника) затраты на передачу буквы a и среднюю длину интервала между появлениями буквы a. В силу выпуклости логарифма из (2.37) после усреднения получаем

$$\bar{\bar{l}}(a) \le \log \bar{r}(a) + 2\log(1 + \log \bar{r}(a)) + 2. \tag{2.38}$$

В соответствии с леммой Каца (Кас) (ее доказательство приведено в [12]) имеет место неравенство

$$\bar{r}(a) \le \frac{1}{p(a)}. \tag{2.39}$$

Подставим эту оценку в (2.37) и усредним по всем буквам алфавита. Далее снова примем во внимание выпуклость логарифма. В результате получим

$$\bar{R} = \sum_a p(a)\bar{l}(a) \leq H(X) + 2\log(1 + H(X)) + 2.$$

Отсюда следует утверждение теоремы. □

Итак, преобразование потока данных с помощью интервального кодирования или кодирования по методу стопки книг с последующим применением кода Левенштейна или кода Элайеса гарантирует достаточно эффективное универсальное кодирование при очень простой реализации кодера и декодера. Эти методы не требуют накопления статистических данных об источнике и построения соответствующего кода источника. Подобные идеи лежат в основе алгоритмов Зива-Лемпела, изучаемых в следующих параграфах.

2.16 Метод скользящего словаря (LZ-77)

Этот алгоритм после его опубликования Зивом и Лемпелом в статье [60] был многократно модифицирован, некоторые модификации получили самостоятельные названия с аббревиатурами вида LZX, где X – первая буква имени автора модификации. В конце параграфа мы коротко обсудим практические аспекты применения алгоритма, в частности, простые усовершенствования, повышающие коэффициент сжатия. Начнем с формального описания алгоритма.

Параметром алгоритма является длина "окна наблюдения" W. Эту величину можно также интерпретировать как "объем скользящего словаря".

Пусть $X = \{0, 1, ..., M-1\}$ – алфавит источника и на выходе источника наблюдается последовательность $x_1, x_2, ..., x_n$. Алгоритм работы кодера показан на Рис.2.14. Кодер LZ-77 хранит в памяти скользящий словарь объема W. Словами словаря служат всевозможные подпоследовательности следующих друг за другом букв, содержащиеся в последних W буквах источника. При поступлении на вход кодера очередных букв источника кодер находит как можно более длинную последовательность, уже имеющуюся в словаре. В канал передается флаг (1 или 0), указывающий на то, найдено или нет подходящее словарное слово. В случае успеха (флаг равен 1) словарное слово передается указанием удаления начала

Заданы:
алфавит источника $X = \{0, 1, ..., M-1\}$;
длина "окна" W
Input: Длина последовательности n;
Последовательность источника $\boldsymbol{x} = \boldsymbol{x}_1^n$;
Output: Кодовое слово $\boldsymbol{c}$ для $\boldsymbol{x}$;

`Инициализация:`
$N = 0$, $\boldsymbol{c}$ - `"пустое" слово;`

while $N < n$ **do**

Находим максимальное l такое, что $\boldsymbol{x}_{N+1}^{N+l} = \boldsymbol{x}_{N-d+1}^{N-d+l}$ для некоторого $d \in \{1, ..., W\}$;

if $l > 0$ **then** `(последовательность найдена)`

- к кодовому слову $\boldsymbol{c}$ дописываются:

  - флаг 1;

  - расстояние до образца d в виде двоичной последовательности длины $\lceil \log W \rceil$;

  - длина совпадения l в виде слова неравномерного префиксного кода;

- $N \leftarrow N + l$;

else `(последовательность не найдена)`

- к кодовому слову $\boldsymbol{c}$ дописываются:

  - флаг 0;

  - новая буква x_{N+1} в виде двоичной последовательности длины $\lceil \log M \rceil$;

- $N \leftarrow N + 1$;

end

Рис. 2.14: Алгоритм LZ-77.

слова от текущей позиции и длины словарного слова. Расстояние до слова d передается равномерным кодом, длина слова (длина совпадения) l – некоторым неравномерным кодом. Например, можно воспользоваться любым из монотонных кодов раздела 4.2. Если же словарного слова не нашлось, передается значение флага равное 0 и за ним следует очередная буква источника, передаваемая “без кодирования”.

2.17 Алгоритм LZW (LZ-78)

Так сложилось, что из двух алгоритмов Зива-Лемпела второй алгоритм, описанный в работе [61], поначалу казался более практичным и первым вошел в практику сжатия. В частности, он использован в стандарте V40bis для передачи данных по стандартным телефонным каналам общего пользования. С учетом модификации, предложенной Велчем в работе [57], ZL-78 действительно выглядит элегантнее первого алгоритма. Ниже мы приводим описание этой модификации, называемой алгоритмом LZW.

Алгоритм LZW показан на рис. 2.15.

Идея алгоритма состоит в том, что вместо последовательностей букв передаются номера слов в некотором словаре. Кодер и декодер в процессе работы синхронно формируют этот словарь. На каждом шаге словарь пополняется одним новым словом, которое до этого в словаре отсутствовало, но является продолжением на одну букву одного из слов словаря.

Пусть $X = \{0, 1, ..., M-1\}$ – алфавит источника и на выходе источника наблюдается последовательность $x_1, x_2, ...$ Для простоты описания алгоритма будем считать, что в начале работы кодера каждая из букв алфавита является словом длины 1 и входит в состав словаря. На каждом следующем шаге находим самое длинное слово, совпадающее с началом подлежащей кодированию последовательности. Пусть l – длина совпадения. Эти l букв передаются в виде ссылки на соответствующее слово словаря. Если объем словаря равен c, то для передачи этой ссылки достаточно $\lceil \log(c-1) \rceil$ бит. Словарь пополняется новым словом, которое получается дописыванием к использованному на данном шаге слову следующей за ним в потоке кодируемых данных буквы.

Поскольку декодер не знает еще этой новой буквы, он сможет выполнить эту операцию только с задержкой на один шаг. Чтобы избежать неоднозначности кодирования, мы запрещаем кодеру пользоваться последним построенным словом словаря, что отражено в алгоритме на

Задан алфавит источника $X = \{0, 1, ..., M-1\}$;
Input: Длина последовательности n;
Последовательность источника $\boldsymbol{x} = \boldsymbol{x}_1^n$;
Output: Кодовое слово $\boldsymbol{c}$ для $\boldsymbol{x}$;

`Инициализация:`
$N = 0$;
$\boldsymbol{c}$ – "пустое" слово;
Словарь состоит из M слов длины 1, т.е. из букв алфавита X
Число слов в словаре $c = M$.

while $N < n$ **do**

- Находим максимальное l такое, что $\boldsymbol{x}_{N+1}^{N+l}$ совпадает с j-м словом словаря при для некотором $j < c$ (на первом шаге допускается $j = c$).
- К кодовому слову дописывается номер словарного слова j в виде двоичной последовательности длины $\lceil \log(c-1) \rceil$ (на первом шаге дописывается последовательность длины $\lceil \log c \rceil = \lceil \log M \rceil$).
- В словарь дописывается новое слово длины $l+1$ вида $\boldsymbol{x}_{N+1}^{N+l+1} = (\boldsymbol{x}_{N+1}^{N+l}, x_{N+l+1})$.
- $N \leftarrow N + l$;
- $c \leftarrow c + 1$;

end

Рис. 2.15: Алгоритм LZW.

рис. 2.15. Исключение составляет первый шаг, когда словарь полностью известен кодеру и декодеру.

Описанный выше способ инициализации алгоритма – не единственный возможный. В частности, при кодировании коротких последовательностей нецелесообразно инициализировать алгоритм, заполняя словарь всеми буквами алфавита источника. Разумнее использовать принцип "*esc*-кода". В начале работы алгоритма словарь пуст и в нем, в ячейке с нулевым номером хранится *esc*-код. В дальнейшем, при получении от источника буквы, отсутствующей в словаре, мы передаем *esc*-код, которому соответствует нулевая последовательность длины $\lceil \log(c-1) \rceil$, а вслед за *esc*-кодом – стандартный код новой буквы.

2.18 Предсказание по частичному совпадению

Сейчас мы рассмотрим наиболее "логичный" алгоритм кодирования при неизвестной статистике источника. На первый взгляд он является почти тривиальным развитием адаптивного арифметического кодирования, которое обсуждалось выше. Разница состоит в том, что в процессе кодирования вместо безусловных вероятностей букв оцениваются их условные вероятности, при известном "контексте", т.е. при известных предшествующих буквах. Клеари и Виттену [36] принадлежит идея использовать контексты различной длины, зависящей от предшествующей закодированной последовательности данных. Этот метод кодирования получил название PPM (prediction by partial matching). Практическая реализация этой простой идеи порождает много вопросов, которые мы кратко обсудим в данном параграфе.

С момента опубликования исходной версии алгоритма было предложено очень много его модификаций. Все же любой из PPM-алгоритмов вкладывается в следующую схему.

Предположим, что последовательность $\boldsymbol{x}_1^t = (x_1, ..., x_t)$ первых t букв источника уже передана и предстоит передать символ x_{t+1}. При кодировании очередной буквы выполняются следующие шаги.

1. Находим *контекст* $\boldsymbol{x}_{t-d+1}^t$ наибольшей длины d, не превышающей заданной величины D. Под *контекстом* понимается последовательность $\boldsymbol{s} = \boldsymbol{x}_{t-d+1}^t$, непосредственно предшествующая кодируемому символу и такая, что в точности такая же последовательность $\boldsymbol{s}$ уже встречалась в переданной последовательности $\boldsymbol{x}_1^{t-1}$.

2. Для всех возможных значений символа x_{t+1} вычисляются оценки условных вероятностей символа при известном контексте $\boldsymbol{s}$.

3. Значение символа x_{t+1} кодируется арифметическим кодом в соответствии с вычисленной на шаге 2 оценкой условной вероятности.

Варианты алгоритма PPM отличаются друг от друга выбором максимальной длины контекста D, и , главное, способом выполнения шага 2, т.е. вычисления оценки условной вероятности символа.

Естественной оценкой условной вероятности буквы $x_{t+1} = a$ после контекста $\boldsymbol{s} = \boldsymbol{x}_{t-d+1}^{t}$ кажется оценка вида

$$\hat{p}_t(a|\boldsymbol{s}) = \frac{\tau_t(\boldsymbol{s}, a)}{\tau_t(\boldsymbol{s})}.$$

В числителе этой формулы записано число появлений буквы a после контекста $\boldsymbol{s}$ в предшествующей последовательности символов $\boldsymbol{x}_1^t$, в знаменателе – общее число появлений $\boldsymbol{t}$ в $\boldsymbol{x}_1^{t-1}$. Однако, арифметический кодер не сможет воспользоваться этой формулой для тех букв a, которые не встречались с заданным контекстом, т.е. при $N(\boldsymbol{s}, a) = 0$. Чтобы исправить ситуацию, следует воспользоваться тем же подходом, что и при побуквенном адаптивном кодировании – ввести esc-символы, указывающие на то, что при заданном контексте данная буква не встречалась.

Стратегия PPM-кодера (точнее его упрощенная версия, не учитывающая так называемых “исключений”, о которых речь пойдет ниже) иллюстрируется алгоритмом, представленным на Рис. 2.16.

Итак, кодер старается передать букву арифметическим кодером в соответствии с вероятностями, определяемыми контекстом наибольшей возможной длины. Если это не удается сделать (буква еще не встречалась вслед за таким контекстом), передается *esc*-символ, контекст укорачивается на одну букву. В конечном счете, буква будет передана с учетом контекста меньшей (возможно, нулевой) длины, либо как “новая буква”, если она не встречалась в уже закодированной последовательности букв

Чтобы окончательно конкретизировать алгоритм, нужно выписать формулы для вероятностей букв и *esc*-символов. Вопрос об адаптивном оценивании вероятностей букв мы уже рассматривали в связи с адаптивным арифметическим кодированием. Перепишем оценки (2.26) и (2.29) так, чтобы использовать их в рамках алгоритма PPM. Оценки условных

Заданы алфавит источника $X = \{0, 1, ..., M-1\}$, максимальная длина контекста D
Input: Длина последовательности n;
Последовательность источника $\boldsymbol{x} = \boldsymbol{x}_1^n$;
Output: Кодовое слово $\boldsymbol{c}$ для $\boldsymbol{x}$;

`Инициализация:`
$t = 0$;
$\boldsymbol{c}$ – "пустое" слово;

while $t < n$ **do**

Находим наибольшее d такое, что $\hat{p}_t(\boldsymbol{x}_{t-d+1}^t) > 0$.
Выбираем контекст $\boldsymbol{s} = \boldsymbol{x}_{t-d+1}^t$.
if $\hat{p}_t(x_{t+1}|\boldsymbol{s}) > 0$ **then**
кодируем x_{t+1} в соответствии с $\hat{p}_t(x|\boldsymbol{s})$.
else
while $\hat{p}_t(x_{t+1}|\boldsymbol{s}) = 0$ **do**
Кодируем esc в соответствии с $\hat{p}_t(esc|\boldsymbol{s})$.
`Уменьшаем длину контекста:`
$d \leftarrow d - 1$, $\boldsymbol{s} = \boldsymbol{x}_{t-d+1}^t$.
if $d > 0$ **then**
Кодируем x_{t+1} в соответствии с $\hat{p}_t(x|\boldsymbol{s})$.
else
`Кодирование без контекста:`
if $\hat{p}_t(x) > 0$ **then**
Кодируем x_{t+1} в соответствии с $\hat{p}_t(x)$.
else
Вычисляем $\hat{p}(esc)$ и передаем esc.
Кодируем x_{t+1} в соответствии с равномерным распределением на не встречавшихся в $\boldsymbol{x}_1^t$ буквах
end
$t \leftarrow t + 1$;
end

Рис. 2.16: Идея алгоритма PPM

вероятностей букв и *esc*-символов для алгоритма A имеют вид

$$\hat{p}_t(a|\boldsymbol{s}) = \frac{\tau_t(\boldsymbol{s},a)}{\tau_t(\boldsymbol{s})+1} \quad , \quad \tau_t(\boldsymbol{s},a) > 0, \tag{2.40}$$

$$\hat{p}_t(esc|\boldsymbol{s}) = \frac{1}{\tau_t(\boldsymbol{s})+1}, \tag{2.41}$$

где $\tau_t(\boldsymbol{s})$ и $\tau_t(\boldsymbol{s},a)$ обозначают число последовательностей соответственно $\boldsymbol{s}$ и $(\boldsymbol{s},a)$, содержащихся в последовательности длины t. Аналогично изменяются оценки для алгоритма D:

$$\hat{p}_t(a|\boldsymbol{s}) = \frac{\tau_t(\boldsymbol{s},a)-1/2}{\tau_t(\boldsymbol{s})} \quad , \quad \tau_t(\boldsymbol{s},a) > 0, \tag{2.42}$$

$$\hat{p}_t(esc|\boldsymbol{s}) = \frac{M_t(\boldsymbol{s})}{2\tau_t(\boldsymbol{s})}, \tag{2.43}$$

где $M_t(\boldsymbol{s})$ – число различных букв, появившихся в последовательности длины вслед за контекстом $\boldsymbol{s}$.

Формулы (2.40)-(2.43) почти завершают описание двух версий алгоритма PPM: PPMA, предложенного в работе [36], и его модификации PPMD, описанной в [50]. Осталось еще одно небольшое, но важное усовершенствование, введенное еще в самой первой работе [36]. Это усовершенствование называется техникой “исключений” (exclusions). Поясним идею этого метода примером.

Предположим, что при кодировании текста на русском языке на некотором шаге контекст имеет вид “кор”, а следующая за ним буква – буква “т”, причем сочетание букв ”корт” ранее не встречалось. Предположим также, что предыдущие появления последовательности “кор” имели продолжениями “а” и “с”. В соответствии с алгоритмом, предается *esc*-символ и контекст укорачивается на единицу, т.е. теперь контекстом будет “ор”. Метод исключений рекомендует принять во внимание то, что не все сочетания “ор” следует использовать для подсчета условной вероятности буквы “т”. Действительно, ведь и кодер и декодер (благодаря передаче *esc*-символа) уже знают, что ни буква “а”, ни буква “с” не могут следовать за “ор”. Формулы (2.40) – (2.43) в данном случае можно уточнить, подставив в них вместо τ_t(“ор”) величину

$$\tau'_t(\text{ор}) = \tau_t(\text{ор}) - \tau_t(\text{ора}) - \tau_t(\text{орс}).$$

После такой корректировки знаменатели в оценках вероятностей букв уменьшатся, а числители останутся прежними. Оценки вероятностей

букв увеличатся, а значит затраты на передачу уменьшатся. (Заметим, что число различных букв $M_n(\boldsymbol{s})$ в числителе (2.43) тоже должно быть пересчитано с учетом исключений).

Следующий пример поможет глубже понять работу алгоритма PPM.

Пример 2.18.1 Применим алгоритмы PPM с параметром D=5 к пословице

IF_WE_CANNOT_DO_AS_WE_ WOULD_WE_SHOULD_DO_AS_WE_CAN

Результаты работы алгоритма по шагам приведены в таблице 2.10. Знаки # в графе "контекст" соответствуют контекстам нулевой длины. Штрихом помечены те вероятности, которые подсчитаны с учетом исключений. В графе $\tau_t(\boldsymbol{s})$ представлено число появлений данного контекста и, если он укорачивался в процессе кодирования, то, через запятую, – число появлений укороченных контекстов. В графе $M_t(\boldsymbol{s})$ также через запятую приведены значения числа различных букв, следовавших за контекстом и его укорочениями.

Рассмотрим несколько характерных шагов кодирования на примере алгоритма PPMA.

Шаг 6. Пробел встретился второй раз. Поскольку предшествующая ему буква E встречалась только один раз, контекст имеет нулевую длину. Для такого контекста пробел имеет вероятность 1/(5+1)=1/6.

Шаг 7. Контекстом длины 1 по отношению к C является пробел. После пробела встречалась только буква W. Нужно передать *esc* (его вероятность равна 1/2) и перейти к контексту нулевой длины. Но и при контексте нулевой длины буква C не встречалась. Снова надо передать *esc*, вероятность которого равна 1/7. Однако, известно, что среди возможных букв не может быть буквы W. Это повышает вероятность *esc*-символа до 1/6.

Шаг 22. Это наиболее типичный шаг, алгоритм работает на этом шаге весьма эффективно. Контекстом по отношению к пробелу выступает _WE . Эта комбинация уже встречалась, и за ней следовал пробел. Таким образом, следующими буквами могут быть пробел и *esc* с равными вероятностями. Поэтому для передачи пробела потребуется всего 1 бит.

Шаг 23. Букве W предшествует контекст _WE_. При предыдущем появлении этого контекста за ним следовала другая буква (C). Поэтому передается *esc* (вероятность 1/2). Контекст укорачивается и преобразовывается в WE_. За этим контекстом ранее следовала только C и надо

снова передавать *esc*. Его вероятность без учета исключений была бы равна 1/2. Но при передаче предыдущего *esc* мы уже информировали, что буква C исключена, поэтому вероятность *esc*-символа равна 1. Контекст сократился до E_. Снова вероятность *esc* равна 1, и контекст сокращается до пробела. После пробела встречались W(2 раза), C, D и A. Буква C невозможна, значит вероятность W равна 2/5.

Шаг 32. Контекст к S – последовательность "_WE_". Она встречалась дважды, но не разу после нее не было буквы S. Передаем *esc*, имеющий вероятность 1/3. Последовательно укорачивая контекст, приходим к контексту, состоящему из одного пробела. Пробел уже встречался 7 раз, но ни разу перед S. Исключаем буквы C и W, тогда возможны только D и A. Поэтому вероятность *esc* равна 1/3. При подсчете вероятности буквы S при пустом контексте из 31 возможности нужно исключить все буквы, которые следовали после пробела (все буквы W, C, D, A, всего 9 букв). Окончательный результат 1/23.

Аналогичные, но еще более запутанные, подсчеты выполняет кодер для алгоритма PPMD. Как и должно было получиться, PPMD экономит биты на контекстах и больше бит тратит на передачу букв. Поскольку в нашем примере мы кодируем короткую последовательность букв, алгоритм PPMD оказался заметно эффективнее алгоритма PPMA и позволяет передать тестовую последовательность кодовым словом длины 232 бит.

2.19 Сжатие с использованием преобразования Барроуза-Уилера

Представленный в данном параграфе способ кодирования значительно отличается от всех рассмотренных ранее. Авторы этого способа, Барроуз и Уилер, описали его впервые в техническом отчете [34] в 1994 г. Предложенный алгоритм показывал неожиданно хорошие результаты, но далеко не рекордные в смысле соотношения между сложностью реализации и достижимым сжатием. С тех пор значительные усилия специалистов в области сжатия данных были направлены на то, чтобы повысить эффективность кодирования и найти теоретическое обоснование алгоритма. В результате на основе преобразования Барроуза-Уилера были построены архиваторы, которые наравне с PPM-кодерами являются рекордсменами по сжатию при сравнении на стандартных наборах файлов (см. раздел 2.20).

Опишем сначала *прямое преобразование Барроуза-Уиллера*. Будем

Таблица 2.10: Пример использования алгоритма PPMA

Шаг	Буква	Контекст $\boldsymbol{s}$	$\tau_t(\boldsymbol{s})$	PPMA	
				$\hat{p}_t(esc\|\boldsymbol{s})$	$\hat{p}_t(a\|\boldsymbol{s})$
1	I	#	0	1	1/256
2	F	#	1	1/2	1/255
3	_	#	2	1/3	1/254
4	W	#	3	1/4	1/253
5	E	#	4	1/5	1/252
6	_	#	5		1/6
7	C	_	1,6	1/2×1/6’	1/251
8	A	#	7	1/8	1/250
9	N	#	8	1/9	1/249
10	N	#	9		1/10
11	O	N	1,10	1/2×1/9’	1/248
12	T	#	11	1/12	1/247
13	_	#	12		2/13
14	D	_	2,13	1/3×1/12’	1/246
15	O	#	14		1/15
16	_	O	1,15	1/2	3/15’
17	A	_	3,16	1/4	1/14’
18	S	A	1,17	1/2×1/16’	1/245
19	_	#	18		4/19
20	W	_	4		1/5
21	E	_W	1		1/2
22	_	_WE	1		1/2
23	W	_WE_	1,1,1,5	1/2×1’×1’	2/5’
24	O	_W	2,2,23	1/3×1’	2/22’
25	U	O	2,24	1/3×1/18’	1/244
26	L	#	25	1/26	1/243
27	D	#	26		1/27
28	_	D	1,27	1/2	6/25’
29	W	_	6		3/7
30	E	_W	3		2/4
31	_	_WE	2		2/3
32	S	_WE_	2,2,2,7,31	1/3×1’×1’×1/3’	1/23’
33	H	S	1,32	1/2×1/25’	1/242
34	O	#	33		3/34
35	U	O	3		1/4
36	L	OU	1		1/2
37	D	OUL	1		1/2
38	_	OULD	1		1/2
39	D	OULD_	1,1,1,1,8	1/2×1’×1’×1’	1/4’
40	O	_D	1		1/2
41	_	_DO	1		1/2
42	A	_DO_	1		1/2
43	S	_DO_A	1		1/2
44	_	DO_AS	1		1/2
45	W	O_AS_	1		1/2
46	E	_AS_W	1		1/2
47	_	AS_WE	1		1/2
48	C	S_WE_	1,3	1/2	1/3’
49	A	_WE_C	1		1/2
50	N	WE_CA	1		1/2
Итого $\lceil 62{,}44 + 185{,}62 \rceil + 1 = 250$ бит					

иллюстрировать его на примере.

Пример 2.19.1 *Прямое преобразование Барроуза-Уилера.* Рассмотрим уже знакомую последовательность

IF_WE_CANNOT_DO_AS_WE_WOULD_WE_ SHOULD_DO_AS_WE_CAN

Все циклические сдвиги последовательности должны быть лексикографически упорядочены. Результат выполнения этого шага для данного примера представлен в таблице 2.11. Результатом преобразования являются

- последний столбец таблицы, строками которой служат лексикографически упорядоченные циклические сдвиги исходной последовательности;
- номер той строки, в которой записана исходная последовательность.

В нашем примере результатами преобразования являются последовательность
CC_____LLWWWWISNUUAANNHWDD_AAOOO____OOEEDTESFDSE
и число 16.

Отметим примечательное свойство преобразованной последовательности: символы в этой ней "сгруппировались". Последовательность почти целиком состоит из серий одинаковых символов. Это дает основание надеяться на то, что сжимать эту последовательность будет проще, чем исходную. Платой является необходимость передать номер строки (он же – номер позиции, на которой находится первый символ исходной последовательности).

Покажем, что по результатам преобразования можно восстановить исходную последовательность. Эта задача решается с помощью *обратного преобразования Барроуза-Уиллера*. Поясним его на примере, представленном в таблице 2.11.

Пример 2.19.2 *Обратное преобразование Барроуза-Уиллера.*

В результате прямого преобразования известен последний столбец таблицы, представленной в таблице 2.11 и номер исходной строки. Требуется восстановить саму исходную строку. Мы легко можем восстановить первый столбец таблицы циклических сдвигов. Для этого достаточно просто произвести сортировку преобразованной последовательности. Итак,

в таблице 2.11 мы знаем первый, последний столбец и индекс исходной строки (число 16).

Понятно, что в 16 ячейке первого столбца мы найдем первую букву последовательности, букву I. Заметим, что, в силу цикличности, в нашей таблице та строка, которая заканчивается первой буквой, начинается со второй буквы. Буквой I заканчивается 14 строка. Заглянув в ее первый столбец, заключаем, что вторым символом был F. Каким был следующий символ?

Находим F в последнем столбце (строка 46) и считываем первый символ той же строки. Оказывается, третьим символом был пробел. В этом месте мы сталкиваемся с небольшой проблемой. Пробелов в последнем столбце много. Какой из них "правильный"?

Выручает то, что все строки таблицы лексикографически упорядочены. Одинаковые символы последнего столбца упорядочены по значениям соответствующих символов первого столбца. Букве F последнего столбца соответствует 9-й по счету пробел первого столбца. Значит, чтобы найти "наш" пробел, мы должны пропустить 8 пробелов последнего столбца и остановиться на девятом. Так мы попадаем на строку 34. В ее первом столбце находим следующий символ W. Продолжая, мы правильно восстановим всю последовательность.

Теперь нужно выбрать способ кодирования преобразованной последовательности. Для этого целесообразно использовать описанный выше в разделе 2.15 метод стопки книг.

Пример 2.19.3 *Кодирование результата преобразования методом стопки книг.*

Перепишем результат прямого преобразования Барроуза-Уиллера
`CC______LLWWWWISNUUAANNHWDD_AAOOO_____OOEEDTESFDSE`
следующим образом. Будем писать символ *esc* всякий раз при появлении "новой" буквы в тексте и "степень новизны" (число различных букв между предыдущим и текущим появлением буквы), если буква встречалась. Получаем
esc 0 *esc* 0 0 0 0 0 *esc* 0 *esc* 0 0 0 *esc* *esc* *esc* *esc* 0 *esc* 0
2 0 *esc* 6 *esc* 0 9 5 0 *esc* 0 0 2 0 0 0 0 1 0 *esc* 0 4 *esc* 2 10
esc 4 2 3

Воспользуемся нумерационным кодированием для передачи этой последовательности. Композиция последовательности и подсчет количества бит на ее передачу даны в таблице 2.12.

В первой строчке таблицы при определении числа бит мы приняли во внимание то, что число различных букв в последовательности может

Таблица 2.11: Лексикографическая сортировка циклических сдвигов

0	A	NIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_	C
1	A	NNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_	C
2	A	S_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO	_
3	A	S_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO	_
4	C	ANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE	_
5	C	ANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE	_
6	D	O_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD	_
7	D	O_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT	_
8	D	_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOU	L
9	D	_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOU	L
10	E	_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_	W
11	E	_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_	W
12	E	_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_	W
13	E	_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_	W
14	F	_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CAN	I*
15	H	OULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_	S
16	I	F_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CA	N
17	L	D_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHO	U
18	L	D_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WO	U
19	N	IF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_C	A
20	N	NOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_C	A
21	N	OT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CA	N
22	O	T_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CAN	N
23	O	ULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_S	H
24	O	ULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_	W
25	O	_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_	D
26	O	_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_	D
27	S	HOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE	_
28	S	_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_	A
29	S	_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_	A
30	T	_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANN	O
31	U	LD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SH	O
32	U	LD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_W	O
33	W	E_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS	_
34	W	E_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF	_
35	W	E_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD	_
36	W	E_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS	_
37	W	OULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE	_
38	_	AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_D	O
38	_	AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_D	O
40	_	CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_W	E
41	_	CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_W	E
42	_	DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOUL	D
43	_	DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNO	T
44	_	SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_W	E
45	_	WE_CANIF_WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_A	S
46	_	WE_CANNOT_DO_AS_WE_WOULD_WE_SHOULD_DO_AS_WE_CANI	F
47	_	WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_WE_WOUL	D
48	_	WE_WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_A	S
49	_	WOULD_WE_SHOULD_DO_AS_WE_CANIF_WE_CANNOT_DO_AS_W	E

Таблица 2.12: Кодирование композиции преобразованной последовательности

Буква	Количество	Биты
esc	15	log(50)
0	23	log(50–15+1)=log(36)
1	1	log(50–15–23+1)=log(13)
2	4	log(12)
3	1	log(8)
4	2	log(7)
5	1	log(5)
6	1	log(4)
7	0	log(3)
8	0	log(3)
9	1	log(3)
10	1	log(2)
Итого $\lceil 33,98 \rceil + 1$=35 бит		

принимать значения от 1 до 50. Нули в преобразованной последовательности появляются при повторении букв. Число нулей находится в интервале от 0 до 35. Исходя из этого, для передачи числа нулей достаточно log(36) бит и т.д. Приведенный в таблице подсчет подразумевает, что все значения в рассматриваемых диапазонах равновероятны и применяется арифметическое кодирование. В этом случае затраты на передачу композиции составят 35 бит. Можно использовать простое равномерное кодирование, отводя $\lceil \log(i+1) \rceil$ бит на передачу числа, значения которого в диапазоне от 0 до i. Тогда затраты составили бы 38 бит.

Для передачи всех букв, соответствующих *esc*-символам последовательности достаточно 8×15=120 бит.

Для передачи номера последовательности в списке всех последовательностей с заданной композицией потребуется

$$\left\lceil \log \frac{50!}{23!15!4!2!1!...1!} \right\rceil = 94 \quad \text{бит} .$$

В итоге получаем оценку

$$l = 35 + 120 + 94 + 6 = 255 \quad \text{бит} ,$$

где последние 6 бит учитывают затраты на передачу индекса исходной строки в таблице циклических сдвигов. Затраты получились примерно такие же, что и при использовании алгоритма PPMA.

Заметим, что рассмотренный метод передачи преобразованной последовательности не очень эффективен. Он учитывает высокую вероятность нулей в преобразованной последовательности, но заведомо не учитывает их тенденции к группированию.

С момента изобретения алгоритма Барроуза-Уилера в 1994 г. по настоящее время было предложено много алгоритмов более эффективного кодирования результатов преобразования. Один из самых удачных алгоритмов предложен Е.Биндером (Binder E.) в 2000 г. и называется методом *кодирования расстояний.* Подробное описание метода приведено в [12].

2.20 Сравнение способов кодирования. Характеристики архиваторов

Завершим раздел, посвященный кодированию дискретных источников, сравнением изученных алгоритмов.

Сначала обратимся к тексту из 50 букв, на котором мы иллюстрировали принципы работы алгоритмов. Результаты вычислений (см. [12]) сведены в таблицу 2.13

Хотя маленькая длина нашего текста не позволяет делать надежных выводов о сравнительных характеристиках алгоритмов, приведенные в таблице данные довольно красноречивы. Предложенные относительно недавно алгоритмы PPMD и кодирование с использованием преобразования Барроуза-Уиллера, существенно эффективнее других методов кодирования.

Интересно сравнить изученные алгоритмы в реальных условиях применения и выбрать лучший из них. Эта задача не имеет однозначного решения. В частности, при кодировании различных файлов данных предпочтительными будут различные алгоритмы.

Многолетняя конкуренция разработчиков алгоритмов сжатия информации привела к созданию наборов тестовых файлов, на которых принято производить сравнение алгоритмов. Чаще всего используется набор, называемый Calgary Corpus, состоящий из 19 файлов, общий объем которых составляет 3251493 байт. В состав набора входя тексты на английском языке (художественные и технические), библиографический список, новости в виде сообщений электронной почты, программы на языках C, Lisp и Паскаль, факсимильное изображение, геофизические данные, исполняемые коды программ.

В таблице 2.14 приведены сведения об алгоритмах с наилучшим сжа-

Таблица 2.13: Сравнение алгоритмов на примере короткого текста

Алгоритм	Раздел	Затраты (бит)
Передача без кодирования		400
Двухпроходное кодирование, код Хаффмена	2.9	302
Нумерационное кодирование	2.10	283
Адаптивное арифметическое кодирование (алгоритм A)	2.11	293
Адаптивное арифметическое кодирование (алгоритм D)	2.11	283
Алгоритм LZ77	2.16	280
Алгоритм LZFG	2.16	299
Алгоритм LZW	2.17	291
Алгоритм PPMA	2.18	250
Алгоритм PPMD	2.18	**232**
Преобразование BW+ стопка книг	2.19	255
Преобразование BW+ кодирование расстояний	2.19	**235**

тием и, для сравнения, результаты тестирования широко используемого архиватора PKZIP. Представленные в таблице данные заимствованы из Интернет-страницы http://compression.ca/ и отражают достижения в области сжатия данных на конец 2002 г. На этой же странице можно получить более подробные данные об этих и многих других архиваторах.

Таблица 2.14: Сравнение архиваторов на тестовом наборе Calgary Corpus

Архиватор (автор)	Принцип работы	Скорость (бит/байт)
RK 1.04.01 (M. Taylor)	LZ, PPMZ	1.8226
SBC 0.968 (S. J. Mäkinen)	Преобразование Барроуза-Уиллера	1.8846
RAR(Win32)3.00b5 (Е.Рошаль)	LZ77 variant + Huffman	1.9244
ACB 2.00c (Г.Буяновский)	Associative Coding Algorithm	1.9546
PPMD vE (Д. Шкарин)	PPM	2.0153 2.0153
PKZIP 2.6.02 Win95 (PKWARE)	LZH, LZW	2.6062

Завершая раздел, повторим, что сжатие компьютерных файлов – лишь одно из многочисленных приложений теории кодирования дискретных источников. Каждый из методов, перечисленных в таблицах 2.13, 2.14, имеет свою область предпочтительного применения в различных задачах хранения, обработки и передачи информации.

Глава 3

Кодирование для дискретных каналов с шумом

В настоящем разделе рассматривается кодирование с целью защиты передаваемой информации от помех и искажений при передаче по каналам связи, либо при хранении информации. Решение задачи состоит в том, что при кодировании в информацию искусственно вносится избыточность таким образом, чтобы при искажении части передаваемых данных сообщения источника могли быть правильно декодированы. Таким образом, в некотором смысле задача кодирования для каналов противоположна задаче кодирования источника.

3.1 Постановка задачи помехоустойчивого кодирования

Начнем с примера. Рассмотрим систему связи, в которой входом канала являются двоичные сигналы, соответствующие символам алфавита $X = \{0, 1\}$. На выходе канала каждый сигнал анализируется и выносится решение в пользу одного из двоичных символов выходного алфавита $Y = X$.

Будем считать, что подлежащая передаче информация представлена в двоичной форме и поэтому элементарные сообщения представляют собой также двоичные символы 0 либо 1. Таким образом, мы можем непосредственно использовать входные символы канала для передачи сообщений источника. Предположим, однако, что из-за шума в канале

связи при передаче сигналов возможны ошибки. Как можно усовершенствовать систему связи, с тем, чтобы устранить ошибки или, по меньшей мере, уменьшить их долю в принятой последовательности сообщений?

Приведем примеры кодов, исправляющих ошибки.

Пример 3.1.1 Будем вместо сообщения 0 передавать последовательность 000, а вместо 1 – последовательность 111. Множество передаваемых по каналу последовательностей образует код, который в данном случае имеет длину 3 и содержит 2 кодовых слова (мощность кода равна 2). Считая, что правильная передача символов более вероятна, чем неправильная, легко предложить разумную стратегию принятия решений декодером. Если принятая из канала последовательность "больше похожа" на 000, чем на 111, декодер считает, что передавалось сообщение 0, в противном случае принимает решение в пользу сообщения 1. Иначе говоря, если число единиц меньше 2, решение принимается в пользу 0, в противном случае – в пользу 1. □

Число несовпадающих позиций в двух последовательностях $\boldsymbol{x}$ и $\boldsymbol{y}$ называют *расстоянием Хэмминга* между этими последовательностями. Описанный в примере декодер является декодером, принимающим решения по принципу минимума расстояния Хэмминга.

Назовем множество последовательностей $\boldsymbol{y}$ на выходе канала, декодируемых в пользу некоторого кодового слова (точнее, в пользу сообщения, соответствующего этому слову), *решающей областью* слова (сообщения). Решающие области для данного примера приведены в таблице 3.1.

Таблица 3.1: Корректирующий код из примера 3.1.1

Сообщения	Кодовые слова	Решающие области
0	000	{000, 001, 010, 100}
1	111	{011, 101, 110, 111}

Нетрудно видеть, что одиночная ошибка в канале связи при использовании такого кода не опасна. Декодер примет правильное решение. Про такой код говорят, что он исправляет все ошибки кратности 1. Заметим, что на передачу одного бита информации данный код затрачивает 3 двоичных сигнала. Разумной характеристикой кода служит его скорость, которая указывает количество бит информации, переносимых одним сигналом. Скорость кода в данном примере равна

$R = 1/3 = 0,33$(бит/символ канала).

Рассмотрим еще один, немного более сложный, пример кода, исправляющего одиночные ошибки.

Пример 3.1.2 Объединим биты передаваемого сообщения в пары. Множество "расширенных сообщений" состоит теперь из 4 различных двоичных комбинаций. Приведенный в таблице 3.2. пример показывает одну из возможных конструкций кода для этого множества сообщений. □

Таблица 3.2: Код примера 3.1.2

Сообщения	Кодовые слова	Решающие области
00	00000	{00000,00001,00010,00100, 01000,10000,11000,10001}
01	10110	{10110,10111,10100,10010, 11110,00110,01110,00111}
10	01011	{01011,01010,01001,01111, 00011,11011,10011,11010}
11	11101	{11101,11100,11111,11001, 10101,01101,00101,01100}

Непосредственной проверкой можно убедиться в том, что этот код также не боится однократных ошибок, но его скорость несколько больше, поскольку 5 сигналов будет затрачено уже на передачу двух двоичных сообщений. Поэтому

$$R = \frac{2}{5} = 0,4 \quad \text{бит/символ канала.}$$

Понятно, что рассмотренные коды не спасают полностью от ошибок при передаче сообщений. Код примера 3.1.1 не защищает от двукратных ошибок. Код примера 3.1.2 исправляет некоторые комбинации 2-кратных ошибок (например, 11000, 10001), но все остальные комбинации из 2 или большего числа ошибок неминуемо приводят к выдаче получателю неправильно декодированных сообщений.

Приведенные выше примеры убеждают в том, что важными характеристиками кода, защищающего информацию от ошибок, являются его скорость и помехоустойчивость. Последняя характеристика может быть численно измерена вероятностью выдачи получателю ошибочного сообщения.

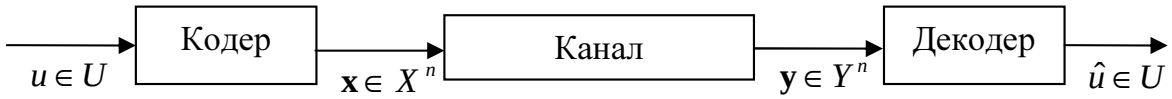


Рис. 3.1: Схема системы связи

Перейдем теперь к формальной постановке задачи. Для этого рассмотрим схему, представленную на рис. 3.1.

В качестве множества сообщений без потери общности можно рассматривать множество чисел $U = \{1, ..., M\}$.

Кодом канала над алфавитом X называется любое множество последовательностей $A = \{\boldsymbol{x}_m\}$, $m = 1, ..., M$, $A \in X^n$. Сами последовательности называются *кодовыми словами*, их длина n называется *длиной кода*, количество последовательностей M называется *мощностью кода*, величина

$$R = \frac{\log M}{n} \tag{3.1}$$

называется *скоростью кода*. Скорость кода измеряется в битах на символ канала.

В частном случае когда $M = 2^k$, каждому из кодовых слов можно сопоставить последовательность из k информационных символов (k бит). Для их передачи будет использовано кодовое слово длины n. Скорость передачи в битах на символ канала составит $R = \log M/n = k/n$ бит/символ.

На каждом такте работы системы связи кодер получает от источника некоторое сообщение u и преобразует в соответствующее кодовое слово $\boldsymbol{x}$. В результате передачи по каналу на выходе канала появляется последовательность $\boldsymbol{y}$. Декодер по последовательности $\boldsymbol{y}$ вычисляет оценку номера переданного сообщения $\hat{u}$. Событие $\hat{u} \neq u$ называется *ошибкой декодирования*, а его вероятность – *вероятностью ошибки*.

Задача состоит в том, чтобы обеспечить наибольшую возможную скорость кода при наименьшей вероятности ошибки. Эти два требования противоречат друг другу. Приведенные выше примеры кодов показывают, что с увеличением числа кодовых слов уменьшаются размеры решающих областей, уменьшается расстояние между кодовыми словами, увеличивается вероятность ошибки. Тем не менее, оказывается, что для каждого канала можно указать скорость кода, при которой вероятность ошибки может быть сделана сколь угодно малой.

Число C называется *пропускной способностью канала*, если при любой скорости кода $R < C$ существуют коды, обеспечивающие сколь угод-

но малую вероятность ошибки и, напротив, при $R > C$ существует константа $\varepsilon > 0$, такая, что вероятность ошибки любого кода ограничена снизу величиной ε (т.е. вероятность ошибки не может быть сделана сколь угодно малой).

Для того, чтобы утверждать, что число C для заданной модели канала является пропускной способностью, нужно доказать две теоремы кодирования: прямую и обратную. Прямая теорема кодирования утверждает, что при любой скорости меньшей C достижима сколь угодно малая вероятность ошибки, обратная теорема – напротив, что при скорости большей C вероятность ошибки будет большой для любого кода.

При решении практических задач помехоустойчивого кодирования помимо скорости и вероятности ошибки необходимо принимать во внимание еще один параметр системы связи – сложность ее практической реализации. Как уже говорилось, практические аспекты построения корректирующих кодов не рассматриваются в данном курсе. В последующих параграфах мы научимся вычислять пропускную способность для некоторых простых моделей каналов и докажем соответствующие теоремы кодирования.

3.2 Модели каналов

Канал преобразует дискретную входную последовательность над алфавитом X в выходную последовательность, элементы которой принадлежат, вообще говоря, другому алфавиту Y. Мы будем говорить, что *модель канала* задана, если для любых n и любых последовательностей $\boldsymbol{x} \in X^n$, $\boldsymbol{y} \in Y^n$ указано правило вычисления условной вероятности $p(\boldsymbol{y}|\boldsymbol{x})$.

Напомним обозначение $\boldsymbol{x}_i^n = (x_i, ..., x_n)$. Канал называется *стационарным*, если для любых j и n и любых $\boldsymbol{x}_{j+1}^{j+n} \in X^n$, $\boldsymbol{y}_{j+1}^{j+n} \in Y^n$ условные вероятности $p(\boldsymbol{y}_{j+1}^{j+n}|\boldsymbol{x}_{j+1}^{j+n})$ определяются однозначно значениями символов последовательностей и не зависят от положения последовательностей во времени (от индекса j).

Канал называется *каналом без памяти*, если для любых j и n и любых $\boldsymbol{x}_{j+1}^{j+n} \in X^n$, $\boldsymbol{y}_{j+1}^{j+n} \in Y^n$

$$p(\boldsymbol{y}_{j+1}^{j+n}|\boldsymbol{x}_{j+1}^{j+n}) = \prod_{i=j+1}^{j+n} p(y_i|x_i).$$

Согласно данному определению, в канале без памяти события, происхо-

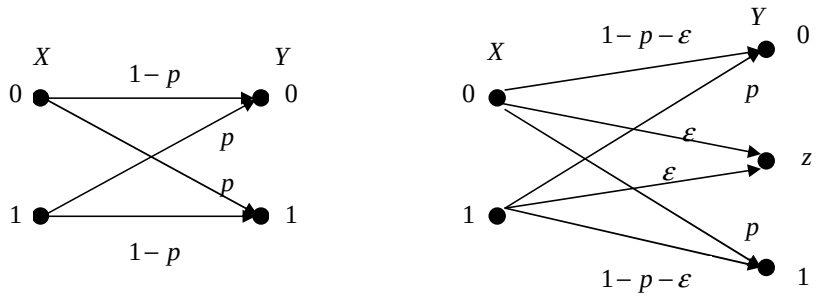
дящие при передаче последовательных символов, независимы.

Стационарный канал без памяти называют *дискретным постоянным каналом* (ДПК).

Для описания ДПК достаточно указать одномерные условные вероятности $\{p(y|x), x \in X, y \in Y\}$. Положим $X = \{0, ..., K-1\}$, $Y = \{0, ..., L-1\}$. Обозначим $p_{ij} = p(y = j|x = i\}$, $i \in X$, $j \in Y$. Переходные вероятности канала p_{ij} удобно записывать в виде матрицы

$$\begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0,L-1} \\ p_{10} & p_{11} & \cdots & p_{1,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{K-1,0} & p_{K-1,1} & \cdots & p_{K-1,L-1} \end{bmatrix}.$$

Эта стохастическая матрица называется *матрицей переходных вероятностей* канала. Она полностью описывает модель ДПК. Приведем два простых, но важных примера ДПК.



а) ДСК б) ДСтК

Рис. 3.2: Примеры дискретных постоянных каналов

Пример 3.2.1 *Двоичный симметричный канал (ДСК).* Для данной модели $X = Y = \{0, 1\}$, а переходные вероятности удовлетворяют условиям $p_{10} = p_{01} = p$, $p_{00} = p_{11} = 1 - p$. Матрица переходных вероятностей имеет вид

$$P = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}.$$

Удобной графической иллюстрацией ДПК являются диаграммы, пример которой для случая ДСК показан на рис. 3.2а. Узлы графа соответствуют входным и выходным символам, ребра показывают возможные трансформации символов в канале, над ребрами записаны соответствующие вероятности переходов.

В ДСК искажения последовательно передаваемых двоичных символов происходят независимо от результата передачи других символов, вероятность искажения p (*переходная вероятность* ДСК) не изменяется во времени и не зависит от значения передаваемого символа.

Пример 3.2.2 *Двоичный симметричный канал со стираниями (ДСтК).* Диаграмма этой модели показана на рис. 3.2б, а матрица переходных вероятностей имеет вид

$$P = \begin{bmatrix} 1-p-\varepsilon & \varepsilon & p \\ p & \varepsilon & 1-p-\varepsilon \end{bmatrix}.$$

Входной алфавит ДСтК содержит два символа, 0 и 1, а выходной алфавит помимо этих двух символов содержит дополнительный символ z, который называют *символом стирания.* В ДСтК каждый из входных символов может с вероятностью p превратиться в противоположный символ и с вероятностью ε может принять неопределенное значение, называемое стиранием. Вероятности ошибки p и стирания ε постоянны, они не зависят от результатов передачи предыдущих и следующих символов и от значения передаваемого в данный момент символа.

3.3 Взаимная информация. Средняя взаимная информация

При изучении каналов связи нас будет интересовать возможность получения информации о передаваемых сообщениях по символам, наблюдаемым на входе канала. Говоря более формально, для заданного произведения $XY = \{(x, y), p(x, y)\}$ дискретных ансамблей X и Y нужно количественно измерить информацию об элементах $x \in X$ входного ансамбля, содержащуюся в выходных символах $y \in Y$.

Подходящей мерой такой информации является *взаимная информация*, определяемая для любых пар $(x, y) \in XY$ соотношением

$$I(x; y) = I(x) - I(x|y). \tag{3.2}$$

Отметим, что мы используем в выражении $I(x; y)$ в качестве разделителя точку с запятой, чтобы не перепутать эту величину с собственной информацией $I(x, y)$ пары сообщений (x, y).

Взаимная информация определена только для пар (x, y), имеющих ненулевую вероятность.

Попробуем пояснить смысл введенного понятия. Уменьшаемое $I(x)$ представляет собой количество собственной информации, содержащейся в сообщении x. Вычитаемое – условная собственная информация при известном y, иными словами, это количество информации, оставшейся в x после получения y. Таким образом, разность $I(x;y)$ представляет собой изменение информации в x благодаря получению y. Вполне разумно принять эту величину в качестве меры информации, содержащейся в y об x.

Отметим свойства взаимной информации.

Свойство 3.3.1 *Симметричность:* $I(x;y) = I(y;x)$.

Свойство 3.3.2 *Если* x *и* y *независимы, то* $I(x,y) = 0$.

Доказательства этих свойств следуют из определения взаимной информации и определений условной вероятности и независимости.

Взаимная информация характеризует пары сообщений и является случайной величиной определенной на произведении ансамблей XY.

Средней взаимной информацией между ансамблями X и Y называется величина

$$I(X;Y) = \mathbf{M}\left[I(x;y)\right].$$

Формула, выражающая среднюю взаимную информацию через совместное распределение вероятностей, имеет вид

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\frac{p(y|x)}{p(y)}. \tag{3.3}$$

Заметим, что средняя взаимная информация определена для произвольных дискретных ансамблей, включая ансамбли, содержащие элементы, имеющие нулевую вероятность. (Вклад пар (x,y) таких, что $p(x,y) = 0$ в величину $I(X;Y)$ будет равен нулю).

Изучим свойства введенной информационной меры.

Свойство 3.3.3 *Симметричность* $I(X;Y) = I(Y;X)$.

Свойство 3.3.4 *Неотрицательность:* $I(X;Y) \geq 0$.

Свойство 3.3.5 *Тождество* $I(X;Y) = 0$ *имеет место тогда и только тогда, когда ансамбли* X *и* Y *независимы.*

Свойство 3.3.6

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(XY).$$

Свойство 3.3.7

$$I(X;Y) \leq \min\{H(X), H(Y)\}.$$

Свойство 3.3.8

$$I(X;Y) \leq \min\{\log|X|, \log|Y|\}.$$

Свойство 3.3.9 *Взаимная информация $I(X;Y)$ – выпуклая $\cap$ функция распределения вероятностей $p(x)$.*

Свойство 3.3.10 *Взаимная информация $I(X;Y)$ – выпуклая $\cup$ функция условных распределений $p(y|x)$.*

Доказательства. Свойство 3.3.3 следует из симметричности взаимной информации. Усредняя правую и левую часть (3.2) по всем парам $(x,y) \in XY$, и учитывая симметричность взаимной информации, получаем первые два тождества свойства 3.3.6. Последнее тождество следует из свойств условной энтропии. Поскольку условная энтропия не превышает безусловной, из свойства 3.3.6 вытекает свойство 3.3.4. Условная энтропия равна безусловной только для независимых ансамблей, поэтому справедливо и свойство 3.3.5. Свойство 3.3.7 также следует из 3.3.6 и свойства неотрицательности условной энтропии. Далее, свойство 3.3.8 является следствием 3.3.7 и свойства энтропии $H(X) \leq \log|X|$. Доказательство свойств 3.3.9 и 3.3.10 можно найти в [12]. □

3.4 Условная средняя взаимная информация. Теорема о переработке информации

Введем еще одно понятие, используемое при анализе информационных характеристик систем. Рассмотрим произведение трех ансамблей $XYZ = \{(x,y,z), p(x,y,z)\}$. Зафиксируем элемент $z \in Z$ и рассмотрим условное распределение

$$p(x,y|z) = \frac{p(x,y,z)}{p(z)}.$$

Это распределение на ансамбле XY, равно как и любое другое, может быть использовано для вычисления средней взаимной информации между X и Y. Будем обозначать эту взаимную информацию как $I(X;Y|z)$. Формула для вычисления этой величины имеет вид

$$I(X;Y|z) = \sum_{x\in X}\sum_{y\in Y} p(x,y|z)\log\frac{p(y|x,z)}{p(y|z)}.$$

Эта величина принимает случайные значения, вероятности которых определяются распределением $p(z)$.

Средней условной взаимной информацией между X и Y при условии Z называется величина

$$I(X;Y|Z) = \mathbf{M}\,[I(X;Y|z] = \sum_{x\in X}\sum_{y\in Y}\sum_{z\in Z} p(x,y,z)\log\frac{p(y|x,z)}{p(y|z)}. \qquad (3.4)$$

Разумеется, средняя условная информация обладает всеми свойствами средней взаимной информации. Помимо этого, непосредственно из (3.4) получаем

$$I(X;Y|Z) = H(Y|Z) - H(Y|XZ). \qquad (3.5)$$

С помощью этой формулы легко доказать тождества

$$I(X;YZ) = I(X;Y) + I(X;Z|Y), \qquad (3.6)$$
$$I(X;YZ) = I(X;Z) + I(X;Y|Z). \qquad (3.7)$$

Первое из них доказывается следующей цепочкой преобразований:

$$\begin{aligned} I(X;YZ) &= H(X) - H(X|YZ) = \\ &= [H(X) - H(X|Y)] + [H(X|Y) - H(X|YZ)] = \\ &= I(X;Y) + I(X;Z|Y). \end{aligned}$$

Второе тождество доказывается аналогично.

Частный случай системы обработки информации, в которой мы имеем дело с тремя вероятностными ансамблями, показан на Рис.3.3.

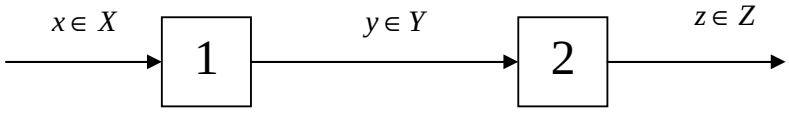


Рис. 3.3: Система обработки информации

Входом системы являются элементы ансамбля X, выход первого устройства Y является входом второго, а выход второго устройства Z является выходом всей системы. Специфической особенностью распределения вероятностей на множестве XYZ является условная независимость X и Z при известном Y. Следствием этой независимости является теорема, называемая *теоремой о переработке информации.*

Теорема 3.1 *Пусть X, Y и Z – вероятностные ансамбли, формируемые системой последовательной обработки информации, показанной на Рис.3.3. Тогда имеют места неравенства*

$$I(X;Y) \geq I(X;Z), \tag{3.8}$$
$$I(Y;Z) \geq I(X;Z). \tag{3.9}$$

Доказательство. Воспользуемся (3.6) и (3.7):

$$I(X;YZ) = I(X;Y) + I(X;Z|Y), \tag{3.10}$$
$$I(X;YZ) = I(X;Z) + I(X;Y|Z). \tag{3.11}$$

В силу условной независимости X и Z при известном Y имеем $I(X;Z|Y) = 0$. Приравнивая правые части (3.10) и (3.11), получаем

$$I(X;Y) = I(X;Z) + I(X;Y|Z).$$

Поскольку второе слагаемое неотрицательно, получаем неравенство (3.8). Аналогично доказывается (3.9). □

Неравенство (3.8) имеет следующее истолкование. Первое устройство можно рассматривать как канал связи, второе – как устройство обработки информации на выходе канала. Теорема утверждает, что при такой обработке взаимная информации между входом и выходом системы не увеличивается. Второе неравенство теоремы говорит о том, что обработка информации на входе системы также не приводит к увеличению информации между входом и выходом. Иными словами никакая обработка информации (детерминированная или случайная) не позволяет увеличить количество информации о входе системы, получаемой на ее выходе.

Можно также рассматривать систему, показанную на рис.3.3, как последовательное соединение двух каналов. В этом случае теорема устанавливает интуитивно понятный факт: количество, проходящей через последовательное соединение каналов, не может быть больше количества информации, проходящей через каждый отдельный канал.

3.5 Информационная емкость и пропускная способность

Задача, которые мы решаем в данном параграфе, состоит в том, чтобы выразить достижимую скорость передачи информации по каналу через известные нам информационные меры, такие как энтропия, взаимная информация и т.п.

Рассмотрим дискретный стационарный канал. Мы знаем, что количество информации о входных символах X канала, содержащееся в выходных символах Y определяется средней взаимной информацией $I(X;Y)$. Это верно при передаче одного символа канала. Из опыта изучения источников информации известно, что кодирование последовательностей сообщений потенциально более эффективно, чем кодирование отдельных сообщений. Точно также при кодировании для каналов мы объединяем последовательные входные символы канала в блоки, представляющие собой кодовые слова некоторого кода.

Таким образом, при использовании кодов длины n количество информации, получаемой декодером при передаче одного кодового слова, в среднем составит $I(X^n;Y^n)$ бит, что соответствует скорости передачи информации

$$\frac{1}{n}I(X^n;Y^n) \quad \text{бит/символ канала} \quad .$$

Эта величина зависит от переходных вероятностей $\{p(\boldsymbol{y}|\boldsymbol{x}),\ \boldsymbol{y}\in Y^n,\ \boldsymbol{x}\in X^n\}$ и от распределения вероятностей на входе канала $\{p(\boldsymbol{x}),\ \boldsymbol{x}\in X^n\}$. Входное распределение следует выбрать таким, чтобы максимизировать скорость передачи информации. Результирующую скорость запишем в виде

$$\max_{\{p(\boldsymbol{x})\}}\frac{1}{n}I(X^n;Y^n) \quad \text{бит/символ канала} \quad .$$

Заметим, что длина кода n также является свободным параметром, она может быть выбрана такой, чтобы скорость передачи была как можно больше. Эти неформальные рассуждения приводят нас к следующему определению.

Величина

$$C_0=\sup_n\max_{\{p(\boldsymbol{x})\}}\frac{1}{n}I(X^n;Y^n) \tag{3.12}$$

называется *информационной емкостью* канала.

Напомним, что sup (supremum) обозначает наименьшую верхнюю грань множества, т.е. наименьшее число, не меньшее, чем любой из

элементов множества. В отличие от максимального элемента, верхняя грань может не принадлежать множеству. Пусть, например $A = \{(n-1)/n, n = 1, 2, ...\}$. Максимального числа в этом множестве не существует, но $\sup A = 1$, причем эта верхняя грань совпадает с пределом последовательности при $n \to \infty$. Аналогично в определении информационной емкости канала наибольшее значение средней взаимной информации на букву канала может достигаться при бесконечной длине кода, поэтому в (3.12) нас интересует именно верхняя грань, а не максимальное значение.

Итак, мы ввели в рассмотрение информационную емкость канала C_0 как некоторую функцию его переходных вероятностей. Мы предполагаем, что именно эта величина характеризует потенциально достижимую скорость передачи по каналу. Выше, в параграфе 3.1 мы определили пропускную способность канала C как максимальную скорость, при которой возможна передача со сколь угодно малой вероятностью ошибки. Естественно, возникает вопрос о том, как соотносятся между собой эти две характеристики канала.

В разделе 1, были введены два понятия: скорость создания информации источником H и предел энтропии на сообщение источника $H_\infty(X) = H(X|X^\infty)$. Было доказано, что для произвольного стационарного источника эти две величины равны.

Вопрос о соотношении меду информационной емкостью канала и его пропускной способностью несколько сложней. Ниже мы докажем обратную теорему кодирования, утверждающую, что информационная емкость C_0 ограничивает сверху скорость, при которой достижима сколь угодно малая вероятность ошибки. Для дискретного постоянного канала будет доказана прямая теорема кодирования, согласно которой при скорости сколь угодно близкой к C_0 достижима сколь угодно малая вероятность ошибки.

3.6 Неравенство Фано

Для доказательства теорем кодирования нам нужно будет установить связь между информационными характеристиками канала и источника с одной стороны и практически важными параметрами – скоростью передачи информации и вероятностью ошибки с другой стороны. Неравенство Фано – ключ к решению этой задачи.

На рис. 3.4 для иллюстрации вводимых ниже обозначений показана обобщенная схема системы передачи сообщений,

Сообщениями являются элементы множества $U = \{u\} = \{0, ..., M-$

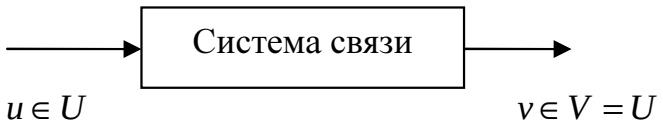


Рис. 3.4: Система передачи информации

1}. Получателю выдаются оценки сообщений. Множество оценок обозначено через $V = \{v\}$. Множества U и V совпадают. Всякий раз, когда $u \neq v$ имеет место ошибка декодирования.

Если задана модель системы связи, т.е. точно известны алгоритмы работы кодера и декодера и математическая модель канала, тем самым определено произведение ансамблей $UV = \{(u, v), p(u, v)\}$. С помощью распределения $p(u, v)$ вероятность ошибки P_e может быть вычислена по формуле

$$P_e = \sum_u \sum_{v \neq u} p(u, v). \tag{3.13}$$

Вероятность правильного решения равна

$$P_c = 1 - P_e = \sum_u \sum_{v=u} p(u, v). \tag{3.14}$$

Теорема 3.2 *(Неравенство Фано)*

$$H(U|V) \leq \eta(P_e) + P_e \log(M - 1), \tag{3.15}$$

где $\eta(\cdot)$ обозначает энтропию двоичного ансамбля (см.(1.3)).

Обсуждение. Прежде чем мы приступим к доказательству этого неравенства, постараемся уяснить его смысл. В левой части неравенства имеем условную энтропию источника сообщений при условии, что известно вынесенное декодером решение. Эта величина характеризует неопределенность о переданном сообщении, оставшуюся на приемной стороне после вынесения решения. Каким образом можно было бы помочь декодеру устранить эту неопределенность? Предположим, что в распоряжении декодера имеется добрый, но практичный джин. Он знает истинное переданное сообщение, но требует плату за каждый бит полученной от него информации. Какие вопросы нужно задать джину?

Одна из нетривиальных стратегий – следующая. Сначала спросим джина, правильно ли принятое решение. Поскольку множество возможных ответов состоит из двух элементов, имеющих вероятности P_e и $P_c = 1 - P_e$, ответ джина обойдется нам в среднем в $\eta(P_e)$ бит. Если решение было правильным, (вероятность этого события $1 - P_e$), то дополнительной информации не требуется. Если же решение ошибочно, то достаточно узнать, какое из остальных $M-1$ возможных сообщений было передано. Эта вторая ситуация имеет вероятность P_e и обойдется нам в худшем случае в $\log(M-1)$ бит. Итого для разрешения неопределенности $H(U|V)$ нам понадобится не более

$$\eta(P_e) + (1 - P_e) \times 0 + P_e \log(M-1)$$

бит, что в точности совпадает с правой частью (3.15).

Эти эвристические рассуждения позволяют понять, почему неравенство Фано имеет место, и облегчают запоминание формулы в его правой части. Конечно, эти рассуждения нельзя считать доказательством неравенства Фано.

Обсудим коротко, какую полезную информацию мы можем извлечь из соотношения (3.15). Введем специальное обозначение $\gamma(P_e)$ для правой части неравенства Фано

$$\gamma(P_e) = \eta(P_e) + P_e \log(M-1) \tag{3.16}$$

и изучим поведение этой функции. В крайних точках 0 и 1 она равна соответственно 0 и $\log(M-1)$. Беря последовательно первую и вторую производную, убеждаемся в том, что функция $\gamma(P_e)$ строго выпукла $\cap$ и имеет максимум в точке $(M-1)/M$. В этой точке $\gamma(P_e) = \log M$. График функции $\gamma(P_e)$ показан на Рис. 3.5.

При доказательстве обратной теоремы кодирования нам понадобится следующее свойство функции $\gamma(P_e)$, которое поясняется представленным на рис. 3.5 графиком: для любого $\delta > 0$ существует положительное $\varepsilon > 0$ такое, что из неравенства $\gamma(P_e) > \delta$ следует неравенство $P_e > \varepsilon$.

Доказательство теоремы 3.2. Используя (3.13) и (3.14), запишем выражения, участвующие в неравенстве Фано (3.15), в следующем виде:

$$H(U|V) = -\sum_u \sum_{v \neq u} p(u,v) \log p(u|v) - \sum_u \sum_{v=u} p(u,v) \log p(u|v), \tag{3.17}$$

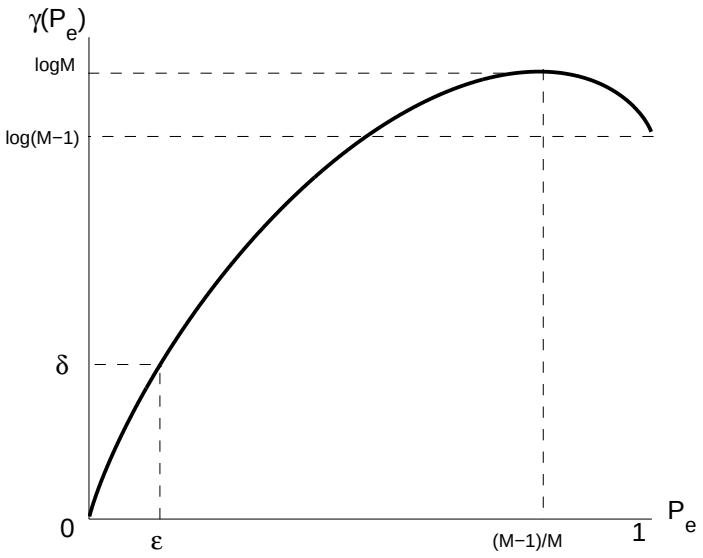


Рис. 3.5: Правая часть неравенства Фано

$$\eta(P_e) = -\sum_u \sum_{v \neq u} p(u,v) \log P_e - \sum_u \sum_{v \neq u} p(u,v) \log P_c, \tag{3.18}$$

$$P_e \log(M-1) = \sum_u \sum_{v \neq u} p(u,v) \log(M-1). \tag{3.19}$$

Рассмотрим разность

$$\Delta = H(U|V) - \eta(P_e) - P_e \log(M-1).$$

Чтобы доказать (3.15), нужно доказать, что $\Delta \leq 0$. Для этого вычтем из правой и левой части (3.17) соответствующие части тождеств (3.18) и (3.19). После элементарных преобразований придем к равенству

$$\Delta = \sum_u \sum_{v \neq u} p(u,v) \log \frac{P_e}{p(u|v)(M-1)} + \sum_u \sum_{v=u} p(u,v) \log \frac{P_c}{p(u|v)}.$$

Следующий шаг основан на использовании неравенства

$$\log x \leq (x-1) \log e.$$

Применив его к обоим слагаемым, получим

$$\Delta \leq (\log e) \left[\sum_u \sum_{v \neq u} p(u,v) \frac{P_e}{p(u|v)(M-1)} - \sum_u \sum_{v \neq u} p(u,v) + \right.$$

$$+\sum_u \sum_{v=u} p(u,v)\frac{P_c}{p(u|v)} - \sum_u \sum_{v=u} p(u,v)\Bigg].$$

Воспользуемся тем, что $p(u,v) = p(v)p(u|v)$, и обозначениями (3.13) и (3.14). Результат легко привести к виду

$$\Delta \le \log e \times \left[\frac{P_e}{M-1}\sum_u \sum_{v\neq u} p(v) - P_e + P_c \sum_u \sum_{v=u} p(v) - P_c\right]. \qquad (3.20)$$

Заметим теперь, что

$$\sum_u \sum_{v\neq u} p(v) = (M-1)\sum_v p(v) = (M-1). \qquad (3.21)$$

Здесь мы использовали то, что сумма в левой части содержит всего M различных слагаемых, и каждое из них встречается в этой сумме ровно $(M-1)$ раз. Кроме того, справедливо равенство

$$\sum_u \sum_{v=u} p(v) = \sum_u p(u) = 1. \qquad (3.22)$$

Подстановка (3.21) и (3.22) в (3.20) приводит к неравенству $\Delta \le 0$. Тем самым неравенство Фано доказано. □

Обобщим теорему 3.2 на случай передачи последовательностей сообщений. Для этого рассмотрим систему связи, показанную на Рис. 3.6. На этот раз входом системы является последовательность сообщений $\boldsymbol{u} = (u_1, ..., u_N)$, а выходом – последовательность решений $\boldsymbol{v} = (v_1, ..., v_N)$, $u_i, v_i \in U = V = \{0, ..., M-1\}$, $i = 1, ..., N$. Вероятность ошибки при передаче i-го сообщения обозначается как $P_{ei} = P(u_i \neq v_i)$.

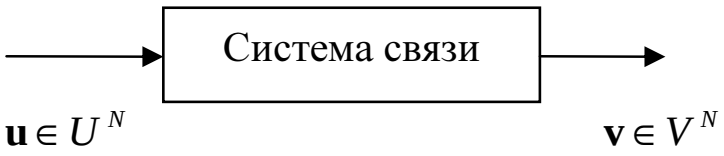


Рис. 3.6: Система передачи информации

Введем в рассмотрение среднюю вероятность ошибки для последовательности длины N

$$\bar{P}_e = \frac{1}{N}\sum_{i=1}^{N} P_{ei}.$$

Теорема 3.3 *Для последовательностей* $(\boldsymbol{u}, \boldsymbol{v}) \in U^N V^N$, *составленных из элементов множества объема* M, *имеет место неравенство*

$$\frac{1}{N} H(U^N|V^N) \le \eta(\bar{P}_e) + \bar{P}_e \log(M-1) \tag{3.23}$$

Доказательство. Напомним, что условная энтропия не возрастает с увеличением числа условий. Поэтому

$$H(U^N|V^N) = \sum_{i=1}^{N} H(U_i|U_1 ... U_{i-1} V^N) \le \sum_{i=1}^{N} H(U_i|V_i).$$

Поделим обе части на N и к каждому слагаемому применим неравенство Фано. Получим

$$\frac{1}{N} H(U^N|V^N) \le \frac{1}{N} \sum_{i=1}^{N} \eta(P_{ei}) + \frac{1}{N} \sum_{i=1}^{N} P_{ei} \log(M-1).$$

Поскольку энтропия – выпуклая $\cap$ функция, средняя по i энтропия не меньше энтропии средней вероятности ошибки. Учитывая это, из последнего неравенства получаем доказываемое неравенство (3.23). $\square$

3.7 Обратная теорема кодирования

В этом параграфе для произвольного дискретного стационарного канала будет доказано, что надежная передача информации со скоростью, превышающей информационную емкость канала, невозможна.

Предполагается, что для передачи используется код C длины n, состоящий из $M = |C|$ кодовых слов. При этом скорость кода равна $R = \log|M|/n$, т.е. число кодовых слов равно $M = 2^{nR}$. Мы считаем сообщениями источника номера кодовых слов кода C. Чтобы упростить запись доказательства, сообщения считаем равновероятными. Такое предположение не сужает общности, поскольку в противном случае, применив кодирование без потерь, можно было бы использовать для передачи этих сообщений коды с меньшей скоростью.

Теорема 3.4 *Обратная теорема кодирования. Для дискретного стационарного канала с информационной емкостью* C_0 *для любого* $\delta > 0$ *существует число* $\varepsilon > 0$, *такое, что для любого кода со скоростью* $R > C_0 + \delta$ *средняя вероятность ошибки удовлетворяет неравенству*

$$\bar{P}_e \ge \varepsilon.$$

Доказательство. Рассмотрим произвольный код C длины n со скоростью $R = \log|C|/n$. Обозначим через N длину последовательностей сообщений $\boldsymbol{u} \in U^N$, сопоставляемых словам кода C, через $\boldsymbol{v} \in V^N$ обозначим последовательности решений, принимаемых декодером. Имеет место цепочка соотношений

$$\begin{aligned} nR &= \log|C| = \\ &\overset{\text{(a)}}{=} H(X^n) \overset{\text{(b)}}{\leq} H(U^N) = \\ &= H(U^N) - H(U^n|V^N) + H(U^N|V^N) = \\ &\overset{\text{(c)}}{=} I(U^N;V^N) + H(U^N|V^N) \leq \\ &\overset{\text{(d)}}{=} I(X^n;Y^n) + H(U^N|V^N) \leq \\ &\overset{\text{(e)}}{\leq} nC_0 + n\gamma(\bar{P}_e). \end{aligned}$$

Здесь мы сначала в (a) использовали предположение о равновероятности кодовых слов, затем в(b) свойство энтропии 1.2.7 (при преобразовании случайного ансамбля U^N в X^n энтропия либо сохранилась либо уменьшилась), далее (c) – свойство взаимной информации, затем в (d) теорему о переработке информации 3.1, в (e) неравенство Фано и определение информационной емкости канала. Функция $\gamma(\cdot)$ определена соотношением (3.16).

Из последнего неравенства получаем

$$\gamma(\bar{P}_e) \geq R - C_0 > \delta.$$

Из свойств функции $\gamma(\cdot)$ (см. обсуждение перед доказательством Теоремы 3.2) следует, что из неравенства $\gamma(\bar{P}_e) > \delta$ следует существование положительного $\varepsilon > 0$ такого, что $\bar{P}_e \geq \varepsilon$. Тем самым теорема доказана. □

Итак, доказанная теорема утверждает, что, если скорость передачи информации хотя бы ненамного (на любую положительную величину δ) превышает информационную емкость канала, вероятность ошибки уже не может быть сколь угодно малой. Она ограничена снизу положительной величиной ε, независящей от длины используемых кодов. Отметим особо, что результат получен для произвольного стационарного канала.

Чтобы доказать прямую теорему кодирования, нам придется существенно сузить множество рассматриваемых моделей. В следующих параграфах мы получим относительно простые формулы для информационной емкости некоторых каналов без памяти и уже затем докажем прямую теорему кодирования для дискретных постоянных каналов.

3.8 Вычисление информационной емкости каналов без памяти

Рассмотрим дискретный постоянный канал (стационарный канал без памяти), заданный входным и выходным алфавитами $X = \{x\}$ и $Y = \{y\}$ и матрицей условных распределений $P = \{p(y|x)\}$. Для данной модели канала для произвольных последовательностей $\boldsymbol{x} = (x_1, ..., x_n)$ и $\boldsymbol{y} = (y_1, ..., y_n)$ условные вероятности $p(\boldsymbol{y}|\boldsymbol{x})$ вычисляются по формуле

$$p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{n} p(y_i|x_i). \tag{3.24}$$

По определению, информационная емкость канала равна

$$C_0 = \sup_n \max_{\{p(\boldsymbol{x})\}} \frac{1}{n} I(X^n; Y^n). \tag{3.25}$$

Эта формула непригодна для вычисления информационной емкости по заданной матрице P, поскольку предполагает вычисление экстремума функции по счетному числу параметров. Нам предстоит дать ответ на вопрос, нельзя ли упростить (3.25), используя (3.24)? Положительный ответ на этот вопрос сформулируем в виде следующей теоремы.

Теорема 3.5 *Информационная емкость дискретного постоянного канала вычисляется по формуле*

$$C_0 = \max_{\{p(x)\}} I(X; Y). \tag{3.26}$$

Доказательство. Доказательство состоит из двух шагов. Сначала мы докажем, что правая часть (3.26) является границей сверху на информационную емкость канала. Затем мы докажем, что эта граница достигается при некотором распределении $p(\boldsymbol{x})$.

При произвольном распределении $\{p(\boldsymbol{x}), \boldsymbol{x} \in X^n\}$ запишем взаимную информацию между входными и выходными последовательностями в виде

$$I\left(X^n; Y^n\right) = H\left(Y^n\right) - H\left(Y^n|X^n\right). \tag{3.27}$$

Воспользовавшись (3.24) и свойствами логарифма и математического ожидания, выполним простые преобразования.

$$\begin{aligned} H\left(Y^n|X^n\right) &= \mathbf{M}\left[-\log p(\boldsymbol{y}|\boldsymbol{x})\right] = \\ &= \mathbf{M}\left[-\log \prod_{i=1}^{n} p(y_i|x_i)\right] = \\ &= \sum_{i=1}^{n} \mathbf{M}\left[-\log p(y_i|x_i)\right] = \\ &= \sum_{i=1}^{n} H(Y_i|X_i). \end{aligned}$$

Из свойств энтропии мы помним, что

$$H\left(Y^n\right) \leq \sum_{i=1}^{n} H(Y_i), \tag{3.28}$$

причем равенство имеет место только для независимых ансамблей. С учетом выполненных выкладок вместо (3.27) получаем неравенство

$$I\left(X^n; Y^n\right) \leq \sum_{i=1}^{n} \left[H\left(Y_i\right) - H\left(Y_i|X_i\right)\right] = \sum_{i=1}^{n} I\left(X_i; Y_i\right). \tag{3.29}$$

Тем самым мы, по сути, завершили первый шаг нашего доказательства. Предположим теперь, что буквы на входе канала независимы и докажем, что при этом предположении в (3.29) имеет мест равенство. Это будет означать, что максимум в (3.25) следует искать только среди таких распределений на X^n, которые порождают последовательности независимых букв на входе канала. Доказательство сводится к доказательству того, что при этом предположении имеет место равенство в (3.28). Проще говоря, нужно доказать, что для дискретного канала без памяти из независимости букв на входе канала следует независимость символов на выходе канала.

Выходное распределение по формуле полной вероятности вычисляется как

$$p(\boldsymbol{y}) = \sum_{\boldsymbol{x} \in X^n} p(\boldsymbol{x}) p(\boldsymbol{y}|\boldsymbol{x}).$$

В предположении о независимости входных символов с учетом (3.24) получаем

$$\begin{aligned} p(\boldsymbol{y}) &= \sum_{\boldsymbol{x}\in X^n} \prod_{i=1}^{n} p(x_i) \prod_{i=1}^{n} p(y_i|x_i) = \\ &= \sum_{\boldsymbol{x}\in X^n} \prod_{i=1}^{n} p(x_i) p(y_i|x_i) = \\ &= \sum_{x_1\in X} \sum_{x_2\in X} \cdots \sum_{x_n\in X} p(x_1)p(y_1|x_1) \cdot p(x_2)p(y_2|x_2) \cdot \ldots \\ &\quad \ldots \cdot p(x_n)p(y_n|x_n). \end{aligned}$$

Поочередно вынося сомножители за знаки сумм, придем к равенству

$$p(\boldsymbol{y}) = \prod_{i=1}^{n} \sum_{x_i\in X} p(x_i)p(y_i|x_i) = \prod_{i=1}^{n} p(y_i),$$

из которого и вытекает доказываемое утверждение о независимости выходных символов канала.

Теперь мы знаем, что при независимых входных символах в (3.29) имеет место равенство. Подстановка этого равенства в определение информационной емкости (3.25) дает

$$C_0 = \sup_n \max_{\{p(\boldsymbol{x})\}} \frac{1}{n} \sum_{i=1}^{n} I(X_i; Y_i).$$

Заметим, что каждое слагаемое зависит только от распределения вероятностей для соответствующего входного символа. Следовательно, поиск максимума может быть выполнен для каждого отдельного слагаемого независимо от других. Получаем

$$C_0 = \sup_n \frac{1}{n} \sum_{i=1}^{n} \max_{\{p(x_i)\}} I(X_i; Y_i).$$

В силу стационарности канала все слагаемые суммы будут одинаковыми. Поэтому

$$C_0 = \sup_n \max_{\{p(x)\}} I(X; Y) = \max_{\{p(x)\}} I(X; Y).$$

□

Мы пришли к вполне естественному результату: для стационарного канала без памяти вычисление информационной емкости сводится к максимизации средней взаимной информации между отдельными буквами по одномерным распределениям.

Важен также доказанный попутно факт: пропускная способность канала без памяти достигается при независимых буквах на входе канала. В этом смысле распределения, порождающие независимые буквы на входе канала, являются оптимальными.

Формула (3.26) много проще исходной формулы (3.25). При небольшом объеме алфавита оптимизация по входным распределениям может быть выполнена аналитически или численными методами с помощью стандартных компьютерных программ. Известен алгоритм Блэйхута [32], решающий эту задачу весьма эффективно.

Тем не менее, хотелось бы иметь явное выражение, связывающее информационную емкость с параметрами канала. Для некоторых простых случаев такие выражения получить несложно, и рассмотрению этих случаев посвящен следующий параграф.

3.9 Симметричные каналы

Нашей задачей является дальнейшее упрощение формулы для информационной емкости канала. Сузив класс каналов до дискретных постоянных каналов (ДПК), задаваемых матрицей условных вероятностей $P = \{p(y|x), x \in X, y \in Y\}$, мы получили формулу

$$C_0 = \max_{\{p(x)\}} I(X;Y). \tag{3.30}$$

Рассмотрим некоторые частные случаи вида матрицы P.

ДПК называется *симметричным по входу*, если все строки его матрицы переходных вероятностей могут быть получены перестановками элементов первой строки.

ДПК называется *симметричным по выходу*, если все столбцы его матрицы переходных вероятностей могут быть получены перестановками элементов первого столбца.

ДПК называется *полностью симметричным*, если он симметричен одновременно по входу и по выходу.

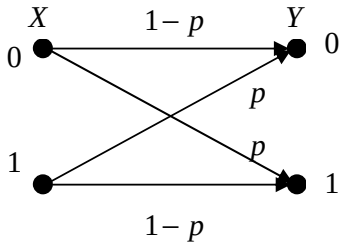
Пример 3.9.1 На Рис. 3.7 показаны три диаграммы переходных вероятностей каналов и соответствующие матрицы переходных вероятностей.

Эти модели описывают полностью симметричный , симметричный только по входу и симметричный только по выходу каналы.

Сформулируем те свойства симметричных каналов, которые упрощают вычисление их информационной емкости.

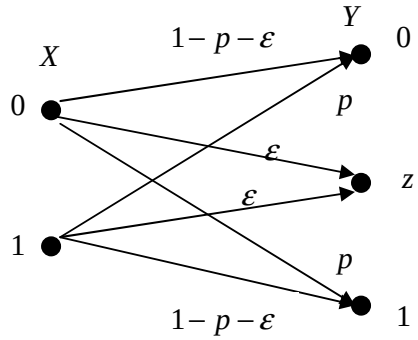
Свойство 3.9.1 *Для симметричного по входу канала без памяти*

$$C_0 = \max_{\{p(x)\}} \{H(Y)\} - H(Y|x), \quad x \in X.$$



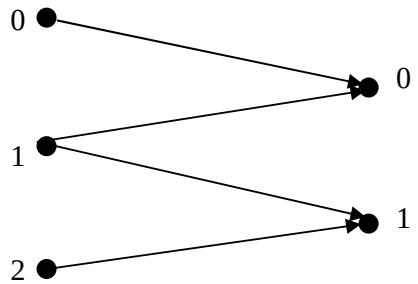
$$P = \begin{Vmatrix} 1-p & p \\ p & 1-p \end{Vmatrix}$$

а) ДСК



$$P = \begin{Vmatrix} 1-p-\varepsilon & \varepsilon & p \\ p & \varepsilon & 1-p-\varepsilon \end{Vmatrix}$$

б) ДСтК



$$P = \begin{Vmatrix} 1 & 0 \\ 1/2 & 1/2 \\ 0 & 1 \end{Vmatrix}$$

в) Симметричный по выходу канал

Рис. 3.7: Симметричные каналы

Свойство 3.9.2 *Для симметричного по входу канала без памяти*

$$C_0 \le \log L - H(Y|x), \quad x \in X.$$

Свойство 3.9.3 *Для симметричного по выходу канала без памяти при равновероятных входных символах выходные символы также равновероятны.*

Свойство 3.9.4 *Для полностью симметричного канала без памяти*

$$C_0 = \log |Y| - H(Y|x), \quad x \in X.$$

Доказательства. Начнем с первого свойства. Напомним, что

$$I(X,Y) = H(Y) - H(Y|X). \tag{3.31}$$

В свою очередь вычитаемое можно записать как

$$H(Y|X) = \sum_x p(x) H(Y|x).$$

Поскольку все строки матрицы переходных вероятностей одинаковы с точностью до нумерации элементов, все условные энтропии $H(Y|x)$ одинаковы. Следовательно, вычитаемое в (3.31) не зависит от входного распределения. Максимизация взаимной информации сводится к максимизации энтропии $H(Y)$, что и утверждается в свойстве 3.9.1.

Второе свойство следует из первого с учетом того, что энтропия ансамбля не превышает логарифма числа его элементов. Больше того, нам известно, что равенство $H(Y) = \log |Y|$ имеет место при равновероятных буквах ансамбля Y. Свойство 3.9.3 доказывается с помощью формулы полной вероятности

$$p(y) = \sum_x p(x) p(y|x).$$

При равновероятных буквах на входе

$$p(y) = \frac{1}{|X|} \sum_x p(y|x), \quad y \in Y$$

Сумма в правой части представляет собой сумму элементов столбца матрицы переходных вероятностей. Поскольку в случае симметричного по выходу канала столбцы одинаковы с точностью до перестановки их элементов, $p(y)$ не зависит от y и имеет место равенство $p(y) = 1/|Y|$. Из первых трех свойств немедленно следует свойство 3.9.4 . □

Пример 3.9.2 *Двоичный симметричный канал.* Из свойства 3.9.4 получаем формулу для информационной емкости двоичного симметричного канала (Рис. 3.7а)

$$C_0 = 1 - \eta(p).$$

График зависимости информационной емкости от переходной вероятности p показан на Рис. 3.8.

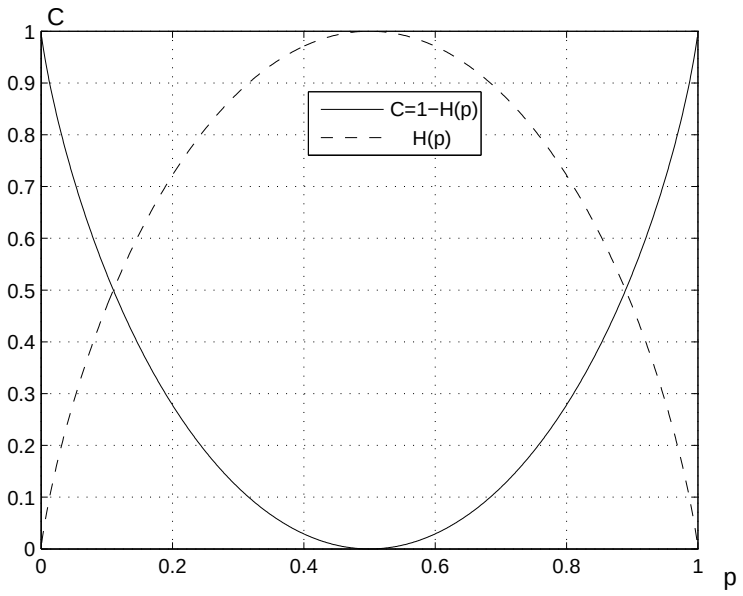


Рис. 3.8: Пропускная способность ДСК

Информационная емкость равна нулю при $p = 1/2$. В этом случае вход и выход канала независимы. Естественно, с уменьшением p информационная емкость растет до значения $C_0 = 1$ при $p = 0$. Информационная емкость симметрична относительно точки $p = 1/2$. Это понятно, ведь при $p > 1/2$, переобозначив выходные символы, мы получим канал с переходной вероятностью $p < 1/2$.

К сожалению, сформулированных свойств недостаточно для того, чтобы получить явные выражения для двоичного стирающего канала, показанного на рис. 3.7б. Для этого симметричного по входу канала имеет место только свойство 3.9.1, предполагающее оптимизацию по входному распределению при вычислении информационной емкости. Однако, данная модель принадлежит классу еще одному широкому классу симметричных каналов, для которых задача вычисления информационной емкости решается просто.

Канал называется *симметричным в широком смысле*, если перенумерацией выходных символов его матрица может быть представлена в форме клеточной матрицы

$$P = [P_1|\ P_2|\ ...\ |P_M]\,, \tag{3.32}$$

в которой каждая из подматриц P_i полностью симметрична (по входу и по выходу).

Пример 3.9.3 *Двоичный канал со стираниями.* Матрица переходных вероятностей ДСтК (рис. 3.7б) перенумерацией выходных символов (перестановкой столбцов) легко приводится к виду

$$P' = \left[\begin{array}{cc|c} 1-p-\varepsilon & p & \varepsilon \\ p & 1-p-\varepsilon & \varepsilon \end{array}\right].$$

Обе подматрицы этой матрицы симметричны по входу и по выходу, следовательно, ДСтК симметричен в широком смысле.

Свойство 3.9.5 *Для симметричного в широком смысле канала максимум взаимной информации между входом и выходом (см. (3.30)) достигается при равновероятных буквах входного алфавита.*

Доказательство. Поскольку симметричный в широком смысле канал заведомо симметричен по входу, достаточно доказать, что равномерное распределение на входе канала максимизирует энтропию выходного алфавита Y.

Рассмотрим представление матрицы переходных вероятностей в форме (3.32) и обозначим через $Y_1, ..., Y_M$ подмножества символов выходного алфавита Y, соответствующие подматрицам $P_1, ..., P_M$. Энтропию ансамбля Y запишем в виде

$$H(Y) = -\sum_{i=1}^{M} \sum_{y \in Y_i} p(y) \log p(y). \tag{3.33}$$

Введем обозначение

$$q_i = \sum_{y \in Y_i} p(y)$$

для вероятности подмножества Y_i. Набор чисел $q_1, ..., q_M$ образует распределение вероятностей на множестве индексов $I = \{1, ..., M\}$. Заметим, что для рассматриваемой модели канала вероятности $q_1, ..., q_M$ не зависят от распределения $\{p(x)\}$ на множестве X. Чтобы убедиться в этом, запишем условную вероятность подмножества Y_i при известном символе x:

$$\mathrm{P}(Y_i|x) = \sum_{y \in Y_i} p(y|x)$$

Сумма в правой части по предположению не зависит от конкретного значения x, следовательно, при всех x имеет место равенство $\mathrm{P}(Y_i|x) = q_i$.

Преобразуем (3.33) к виду

$$H(Y) = -\sum_{i=1}^{M} q_i \sum_{y \in Y_i} \frac{p(y)}{q_i} \log \left(\frac{p(y)}{q_i} q_i \right) = H(I) + \sum_{i=1}^{M} q_i H(Y_i), \qquad (3.34)$$

где

$$H(I) = -\sum_{i=1}^{M} q_i \log q_i \quad -$$

энтропия множества индексов I, а

$$H(Y_i) = -\sum_{y \in Y_i} \frac{p(y)}{q_i} \log \frac{p(y)}{q_i} \quad -$$

энтропия подмножества Y_i. Поскольку первое слагаемое в (3.34) не зависит от входного распределения, достаточно рассматривать второе слагаемое. Заметим, что, если все элементы подматрицы P_i поделить на вероятность q_i, то результатом будет стохастическая матрица, которая является матрицей переходных вероятностей полностью симметричного канала. Согласно свойству 3.9.3 максимальная энтропия выходного алфавита (равномерное распределение на выходе канала) достигается при равновероятных входных символах. Таким образом, при равновероятных символах на входе достигается максимум каждого слагаемого в правой части (3.34) и, следовательно, максимум энтропии $H(Y)$. □

Пример 3.9.4 Вернемся к модели ДСтК, представленной на рис. 3.7б. Благодаря свойству 3.9.5 мы знаем, что информационная емкость этого канала равна взаимной информации между входом и выходом при равновероятных входных символах $p_x(0) = p_x(1) = 1/2$. Используя формулу полной вероятности, находим, что

$$p_y(0) = p_y(1) = \frac{1-\varepsilon}{2}, \quad p_y(z) = \varepsilon.$$

Подставив эти значения в формулу для взаимной информации, после несложных преобразований можно получить формулу для информационной емкости

$$C_0 = (1-\varepsilon)\left(1 - \eta\left(\frac{p}{1-\varepsilon}\right)\right).$$

Из этой формулы, в частности следует очень простая формула для информационной емкости канала со стираниями и без ошибок. При $p = 0$ имеем

$$C_0 = 1 - \varepsilon.$$

Результат кажется очевидным. Поскольку доля "стертых" символов равна ε, то доля успешно переданных и при этом (в данном случае) заведомо правильно переданных символов как раз равна $1-\varepsilon$. Проблема, однако, состоит в том, что кодер не знает заранее, какие именно символы будут стерты. Тем не менее, полученная формула для информационной емкости говорит о том, что возможно такое кодирование, при котором скорость кода будет такой же, как если бы позиции стертых символов были известны заранее. При этом вероятность ошибки, как следует из доказываемой ниже прямой теоремы кодирования, может быть сделана сколь угодно малой.

3.10 Прямая теорема кодирования для дискретных постоянных каналов

Нам предстоит доказать прямую теорему кодирования, утверждающую, что при скорости меньшей информационной емкости вероятность ошибки декодирования может быть сделана сколь угодно малой. Прямой путь к цели состоит в построении последовательности кодов возрастающей длины n с убывающей по мере роста длины кодов вероятностью ошибки. Парадокс состоит в том, что таких конструкций кодов до сих пор не найдено, при том, что теорема кодирования, устанавливающая их существование, была успешно доказана Шенноном более полувека назад.

Метод, использованный Шенноном, называется *методом случайного кодирования*. Идея его очень проста. При заданной длине и скорости кода множество кодовых слов кода строится случайным выбором символов в соответствии с некоторым распределением вероятностей. При таком построении кода сам код может рассматриваться как случайное событие, а его вероятность ошибки – как случайная величина. Предположим, что удалось найти математическое ожидание вероятности ошибки по множеству кодов и это значение оказалось равным некоторой величине ε. Тогда, очевидно, мы сможем утверждать, что, по меньшей мере, для одного из кодов вероятность ошибки не превышает ε.

Итак, воспользуемся методом случайного кодирования для доказательства прямой теоремы кодирования для дискретных постоянных каналов. Напомним, что рассматриваемый канал задается матрицей условных вероятностей $P = \{p(y|x), x \in X, y \in Y\}$ получения выходных символов $y \in Y$ при передаче по каналу символов $x \in X$. Поскольку ДПК – канал без памяти, для любой пары последовательностей $\boldsymbol{x} = (x_1, ..., x_n) \in X^n$, $\boldsymbol{y} = (y_1, ..., y_n) \in Y^n$ условная вероятность $p(\boldsymbol{y}|\boldsymbol{x})$

вычисляется как произведение побуквенных условных вероятностей:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{n} p(y_i|x_i).$$

Обозначим через $\hat{m}$ принятое декодером решение о номере переданного кодового слова при передаче кодового слова $\boldsymbol{x}_m$. Событие $\hat{m} \neq m$ представляет собой ошибку декодирования при передаче сообщения m, вероятность этого события обозначим как P_{em}. Считая все сообщения (номера кодовых слов) равновероятными, среднюю вероятность ошибки определим соотношением

$$P_e = \frac{1}{M} \sum_{m=1}^{M} P_{em}.$$

Теорема 3.6 *Для дискретного постоянного канала с информационной емкостью C_0 для любых ε,$\delta > 0$ существует достаточно большое число n_0 такое, что для любого натурального числа $n \geq n_0$ существует код длины n со скоростью $R \geq C_0 - \delta$, средняя вероятность ошибки которого $P_e \leq \varepsilon$.*

Иными словами, мы утверждаем, что при скорости кода сколь угодно близкой к информационной емкости C_0 (но, конечно, меньшей C_0 хотя бы на малую величину δ) увеличением длины кодовых слов можно добиться сколь угодно малой вероятности ошибки (меньше любого ε).

Доказательство. Доказательство состоит из следующих шагов:

1. Построить ансамбль случайных кодов с заданной длиной и скоростью.

2. Указать правило декодирования.

3. Оценить среднюю по ансамблю кодов вероятность ошибки и доказать, что вероятность ошибки убывает с увеличением длины кодов.

Полный текст доказательства приведен в книге [12]. □

В результате доказательства прямой и обратной теорем кодирования установлено, что скорость сколь угодно близкая к информационной емкости C_0 (но меньшая C_0) достижима при сколь угодно малой вероятности ошибки, а при скорости большей C_0 вероятность ошибки не может быть сделана произвольно малой. Таким образом, C_0 совпадает с пропускной способностью канала C.

3.11 Непрерывные каналы дискретного времени

В разделе 3 мы рассмотрели дискретные каналы и убедились в том, что информационная емкость (максимум взаимной информации по входным распределениям) определяет максимальную достижимую скорость передачи информации, при которой может быть обеспечена сколь угодно высокая надежность. Напомним, что скорость передачи измерялась в битах на один входной символ канала.

Потребителю системы связи не очень интересна скорость в битах на символ. Гораздо информативнее был бы ответ, выраженный в битах в секунду. Понятно, что этот ответ зависел бы уже не только от способа кодирования и декодирования, но и от устройства модулятора и демодулятора. Такая постановка задачи намного сложнее. Однако теперь, после изучения принципов измерения информации непрерывных источников (раздел 4) и обретения опыта их анализа, мы готовы к тому, чтобы установить зависимость предельно достижимой скорости передачи данных от наиболее важных характеристик канала: отношения сигнал/шум и ширины полосы частот.

Как и для дискретных каналов, все результаты носят теоретический характер и не конструктивны. Методам модуляции и кодирования посвящены специальные разделы теории систем связи, эти вопросы выходят далеко за рамки курса теории информации. Тем не менее, в конце данного разделы будет приведено сравнение известных на момент написания учебника способов модуляции и кодирования с предельными характеристиками, которые традиционно называют пределами Шеннона.

Обозначим через X и Y соответственно входной и выходной алфавиты канала связи. Оба алфавита считаем непрерывными, без потери общности можно считать, что оба множества совпадают с множеством всех вещественных чисел.

В этом параграфе мы рассматриваем каналы дискретного времени. Это означает, что входом канала является последовательность $\boldsymbol{x} = (x_1, x_2, ..., x_n)$, ей соответствует выходная последовательность $\boldsymbol{y} = (y_1, y_2, ..., y_n)$, а модель канала задается условными плотностями распределения вида $f(\boldsymbol{y}|\boldsymbol{x}) = f(y_1, \ldots, y_n|x_1, \ldots, x_n)$.

Как и в случае дискретных каналов, мы будем считать канал стационарным, т.е. условные плотности $f(\boldsymbol{y}|\boldsymbol{x})$ независящими от положения последовательностей $\boldsymbol{x}$ и $\boldsymbol{y}$ во времени. Помимо этого, мы изучаем толь-

ко каналы без памяти, т.е.

$$f(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{n} f(y_i|x_i).$$

Еще одно разумное упрощение модели состоит в том, что искажение сигнала в канале не зависит от передаваемого сигнала. Мы рассматриваем модель, показанную на рис. 3.9. Выход канала y связан с входом x соотношением

$$y = x + z,$$

в котором случайная величина z не зависит от x. Аналогичное соотношение имеет место и для последовательностей, т.е. $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{z}$. Из этого равенства следует, что

$$f(\boldsymbol{y}|\boldsymbol{x}) = f(\boldsymbol{y} - \boldsymbol{x}|\boldsymbol{x}) = f(\boldsymbol{z}|\boldsymbol{x}) = f(\boldsymbol{z}).$$

Такая модель канала называется *каналом с аддитивным шумом.*

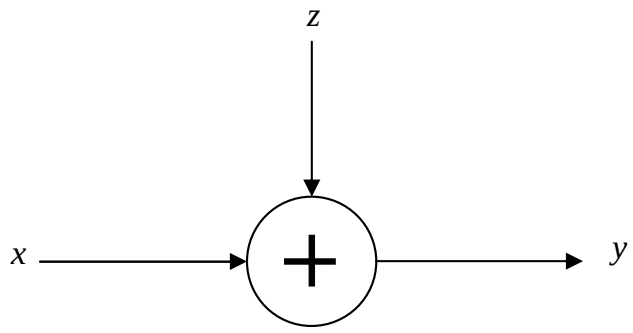


Рис. 3.9: Канал дискретного времени с аддитивным шумом

Для канала без памяти с аддитивным шумом имеет место соотношение

$$f(\boldsymbol{z}) = \prod_{i=1}^{n} f(z_i).$$

Действуя по аналогии с дискретными каналами, мы должны были бы определить информационную емкость канала как максимум взаимной информации $I(X;Y)$ по всем распределениям на входе канала. Однако, в случае канала с непрерывным входом этот максимум будет бесконечным. Чтобы убедиться в этом, рассмотрим сначала канал без шума. Взаимная информация в этом случае равна дифференциальной энтропии входного распределения, и ее максимум может быть сколь угодно

большим. Например, в качестве входного алфавита можно выбрать все множество целых чисел $\mathbb{Z}$. Приписывая числам этого множества одинаковые вероятности, получим ансамбль с бесконечной энтропией, буквы которого передаются абсолютно надежно.

Если же шум присутствует, то в качестве входа канала можно выбрать решетку $d\mathbb{Z}$. с таким большим шагом решетки d, чтобы влиянием шума можно было бы пренебречь. И в этом случае, как мы видим, энтропия входного алфавита тоже может быть сделана сколь угодно большой.

Чтобы придать смысл решаемой задаче, нам следует наложить дополнительные ограничения на множество входных сигналов. Наиболее естественным ограничением является ограничение на мощность входного сигнала.

Энергией входной последовательности $\boldsymbol{x} = (x_1, ..., x_n)$ называется величина $\sum_{i=1}^{n} x_i^2$.

Мощностью или средней энергией на отсчет называют величину

$$E(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} x_i^2}{n}$$

Информационной емкостью непрерывного стационарного канала дискретного времени с ограничением E на мощность входных сигналов называется величина

$$C_0 = \sup_n \max_{f(\boldsymbol{x}):\mathbf{M}[E(\boldsymbol{x})]\le E} \frac{1}{n} I(X^n; Y^n).$$

Как естественное обобщение результатов, полученных в главе 3 для дискретных каналов, приведем формулы для информационной емкости самых простых и важных для теории и практики моделей каналов.

Теорема 3.7 *Информационная емкость непрерывного стационарного канала дискретного времени без памяти с ограничением E на мощность входных сигналов равна*

$$C_0 = \max_{f(x):\mathbf{M}[E(x)]\le E} I(X; Y). \tag{3.35}$$

Доказательство. По аналогии с дискретными каналами нужно сначала показать, что правая часть (3.35) является оценкой сверху для информационной емкости. Затем нужно показать, что эта оценка достигается в том случае, когда последовательность на входе канала – последовательность независимых одинаково распределенных величин. Доказательство

опирается, в частности, на выпуклость взаимной информации как функции распределения вероятностей на входном алфавите. Формальную запись доказательства предоставляем читателю в качестве упражнения. □

Мы называем канал *гауссовским*, если шум в канале имеет гауссовское распределение, вообще говоря, многомерное, описываемое (4.8). В данном разделе мы рассматриваем только гауссовские каналы без памяти, в которых одномерное распределение шума имеет вид

$$f(z) = \frac{1}{\sqrt{2\pi N_0}} e^{-\frac{z^2}{2N_0}}$$

Теорема 3.8 *Информационная емкость гауссовского стационарного канала дискретного времени без памяти с ограничением E на мощность входных сигналов равна*

$$C_0 = \frac{1}{2}\log\left(1 + \frac{E}{N_0}\right) \tag{3.36}$$

Доказательство. □

Поскольку дифференциальная энтропия гауссовского распределения при заданной мощности (дисперсии) больше, чем дифференциальная энтропия любого другого распределения с той же мощностью (Свойство 4.1.6) из Теоремы 3.8 вытекает следующее утверждение.

Следствие 3.9 *Информационная емкость стационарного канала дискретного времени без памяти с ограничением E на мощность входных сигналов*

$$C_0 \leq \frac{1}{2}\log\left(1 + \frac{E}{N_0}\right)$$

Отношение E/N_0 называют *отношением сигнал/шум*. Как мы видим, эта величина определяет информационную емкость и, в конечном счете, достижимую скорость надежной передачи информации по каналу.

Для того, чтобы утверждать, что информационная скорость действительно является пропускной способностью канала, нужно доказать прямую и обратную теоремы кодирования. Доказательство теорем кодирования для непрерывных каналов выходит за рамки нашего курса. Тем не менее, в дальнейшем вместо информационной емкости будем использовать более часто используемое понятие “пропускная способность”, принимая на веру справедливость теорем кодирования.

3.12 Канал непрерывного времени с аддитивным белым гауссовским шумом

В этом параграфе нам предстоит получить формулу для пропускной способности канала непрерывного времени. Строгое изложение этих вопросов потребовало бы использования большого числа понятий, которые не использовались раньше в данном учебнике. Поэтому мы ограничимся эвристическими рассуждениями, обосновывающими фундаментальный результат Шеннона, полученный им еще в 1948 году [21].

Рассмотрим стационарный канал с аддитивным гауссовским шумом. Для описания модели гауссовского шума $z(t)$ достаточно задать его корреляционную функцию

$$K_z(t) = \mathbf{M}[z(t)z(t+\tau].$$

Преобразование Фурье от корреляционной функции

$$S_z(f) = \int_{-\infty}^{\infty} K(\tau)e^{-2\pi i f\tau}d\tau$$

представляет собой *спектральную плотность мощности* стационарного процесса непрерывного времени. (Сравните с определением спектральной плотности мощности дискретного процесса (4.15)). Частота f измеряется в Герцах и, вообще говоря, принимает произвольные вещественные значения из интервала $(-\infty, \infty)$.

Функция $S_z(f)$ – вещественная неотрицательная и четная (см. задачу **??** главы 4).

Аналогично можно записать спектральную плотность мощности $S_x(f)$ процесса $x(t)$ на входе канала. Тогда ограничение на мощность E входного сигнала примет вид

$$E_x = \int_{-\infty}^{\infty} S_x(f)df \le E \tag{3.37}$$

Будем интерпретировать значения $S_x(f)$ и $S_z(f)$ как мощность соответственно сигнала и шума на частоте f (правильнее было бы говорить о неперекрывающихся малых интервалах частот а не об отдельных частотах). При бесконечной длине сигналов их проекции на гармонические функции становятся некоррелированными (см. [3], [7], [39]). Приходим к совокупности независимых параллельных каналов, причем, согласно Теореме 3.8 пропускная способность каждого канала равна

$$\frac{1}{2}\log\left(1+\frac{S_x(f)}{S_z(f)}\right).$$

Интегрируя по частоте, приходим к формуле для пропускной способности стационарного канала с аддитивным гауссовским шумом

$$C = \sup_{S_x(f)} \frac{1}{2} \int_{-\infty}^{\infty} \log \left(1 + \frac{S_x(f)}{S_z(f)} \right) df. \tag{3.38}$$

где точная верхняя грань вычисляется по всем по всем спектральным плотностям $S_x(f)$, удовлетворяющим ограничению (3.37) на мощность процессса.

Поиск экстремума методом множителей Лагранжа приводит к следующему результату:

$$C = \frac{1}{2} \int_{-\infty}^{\infty} \log \left(1 + \frac{\max\{(\theta - S_z(f)), 0\}}{S_z(f)} \right) df. \tag{3.39}$$

где θ выбирается из условия

$$\int_{-\infty}^{\infty} \max\{(\theta - S_z(f)), 0\} df = E.$$

Такое распределение энергии по частотам называют *принципом разливания воды*: суммарная мощность сигнала и шума "растекается" таким образом, что уровень везде одинаков, за исключением тех участков спектра, где мощность шума превышает этот уровень (в этих участках энергия сигнала равна нулю). Пример приведен на рис. 3.10. Параметр θ подбирается так, чтобы площадь заштрихованной области ("объем воды") была равна разрешенной энергии сигнала. При вычислении пропускной способности по формуле (3.39) интегрирование выполняется по интервалам частот, в которых энергия сигнала положительна ($\theta > S_z(f)$).

Рассмотрим теперь важные частные случаи. Прежде всего, предположим, что спектральная плотность шума постоянна и равна $S_z(f)=N_0/2$ во всем диапазоне частот. Такой процесс называют *белым шумом*. Физически белый шум существовать не может, поскольку его мощность бесконечна.

С помощью обратного преобразования Фурье легко убедиться в том, что корреляционная функция белого шума имеет вид

$$K_z(t) = \frac{N_0}{2} \delta(t),$$

где функция $\delta(t)$ – дельта-функция Дирака, определяемая соотношени-

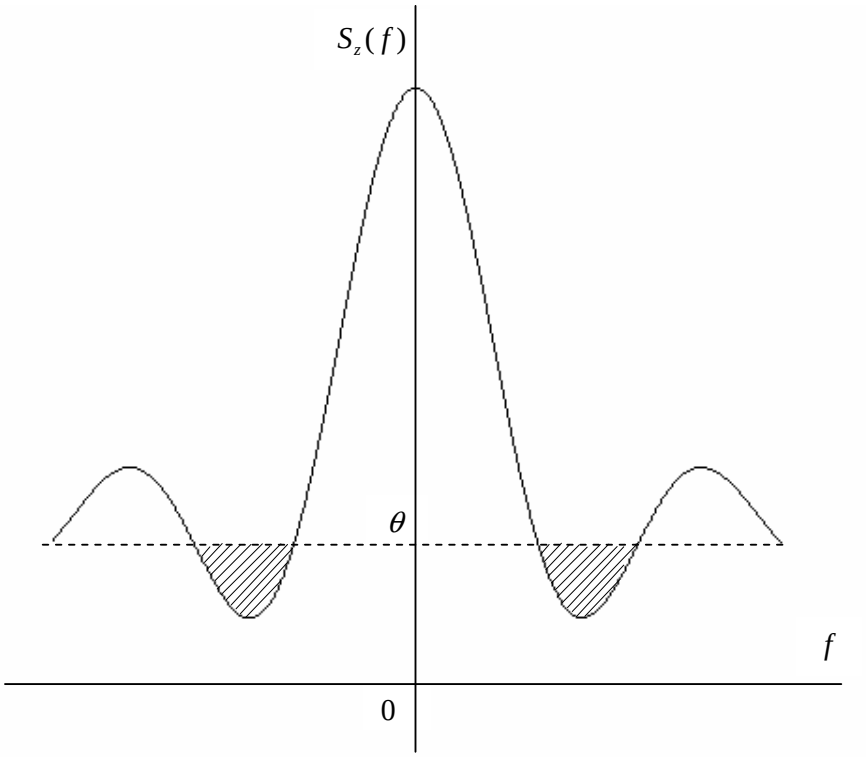


Рис. 3.10: Принцип разливания воды

ями

$$\delta(t) = \begin{cases} 0 & t \neq 0 \\ \infty & t = 0 \end{cases},$$
$$\int_{-\infty}^{\infty} \delta(t) = 1.$$

Канал с аддитивным белым гауссовским шумом (АБГШ) – модель непрерывного по времени канала, аналогичная дискретной по времени модели канала без памяти. Соседние, сколь угодно близкие по времени отсчеты шума независимы, поскольку корреляционная функция равна нулю во всех точках за исключением нуля.

Из сопоставления (3.38) с (3.39) следует, что при равномерной плотности мощности шума $S_z(f)$ оптимальное распределение мощности сигнала тоже будет равномерным. Наложим теперь физически оправданное ограничение на ширину полосы частот сигнала W Гц. Это означает, что спектральная плотность мощности сигнала равна $S_x(f) = E/(2W)$.

Заметим, что любой сигнал конечной длительности имеет, строго говоря, бесконечный спектр. В этом смысле предположение об ограниченности спектра не может выполняться в точности. Мы предполагаем, что доля энергии сигнала в части его спектра за пределами полосы частот

$[-W, W]$ ничтожно мала. Подчеркнем еще раз, что, говоря о полосе частот ширины W, мы подразумеваем сигнал, спектр которого симметричен в интервале частот $[-W, W]$.

С учетом сделанных замечаний получаем формулу для пропускной способности канала с АБГШ с дисперсией шума $N_0/2$ и ограничениями E и W соответственно на мощность и полосу сигнала

$$C = W \log \left(1 + \frac{E}{WN_0}\right) \quad \text{бит/с.} \tag{3.40}$$

Эта формула – пожалуй, один из главных результатов Шеннона.

Если мы теперь откажемся от ограничения на полосу частот, то получим

$$\begin{aligned} C &= \lim_{W\to\infty} W \log \left(1 + \frac{E}{WN_0}\right) \\ &= \frac{E}{N_0 \ln 2} \quad \text{бит/с.} \end{aligned} \tag{3.41}$$

Чтобы получить этот результат, мы воспользовались правилом Лопиталя.

Полученные в данном параграфе формулы позволяют по заданным параметрам канала подсчитать потенциально достижимую скорость передачи информации.

Пример 3.12.1 Телефонный канал. Хотя по паре телефонных проводов можно передавать информацию в очень большом диапазоне частот, модем для телефонного канала спроектирован так, чтобы весь сигнал помещался в полосе речевого сигнала, т.е. в полосе $W = 3400$ Гц. Это ограничение связано с использованием на телефонных станциях аппаратуры уплотнения каналов, в которой именно такая полоса отводится на каждый канал. При отношении сигнал-шум на Гц равном $E/(WN_0) = 100$ из формулы (3.40) находим, что $C \approx 22000$ бит/с. Мы знаем, что современные модемы надежно работают и при более высоких скоростях. В этом нет противоречия. Дело в том, что реальные характеристики каналов несколько лучше, чем те, что мы использовали при расчетах. Кроме того, современные модемы для телефонных линий используют очень непростые и очень эффективные методы модуляции и корректирующие коды.

Как мы уже отмечали, пропускная способность – лишь теоретический предел скорости, при которой потенциально возможна надежная

передача. Существующие доказательства теорем кодирования не конструктивны, они не указывают способов построения кода. Более того, известные сегодня разнообразные конструкции кодов по своим характеристикам заметно уступают теоретически существующим оптимальным кодам.

В следующем параграфе мы покажем, как теорема Шеннона используется на практике для анализа эффективности кодирования и сравним характеристики существующих способов кодирования соотносятся с теоретическими пределами.

3.13 Энергетический выигрыш кодирования

Пропускная способность, вычислению которой посвящены предыдущие параграфы, является характеристикой канала связи. Цель данного параграфа – анализ характеристик способов кодирования. Традиционной мерой эффективности служат затраты энергии на передачу одного бита информации. Ниже мы сформулируем критерий эффективности более точно и посмотрим, насколько близки характеристики применяемых на практике кодов к теоретическому пределу.

Рассмотрим канал с АБГШ без ограничения на полосу сигнала. Предположим, что спектральная плотность мощности шума равна $N_0/2$, мощность передаваемого сигнала равна E *ватт* или *джоулей/с*, а скорость передачи составляет R *бит/с*. Это означает, что затраты энергии на бит состаявляют

$$E_b = \frac{E}{R} \quad \text{дж/бит.}$$

Величину E_b/N_0 называют отношением *сигнал-шум на бит*. Из (3.36) следует, что при больших отношениях сигнал-шум пропускная способность пропорциональна логарифму отношения сигнал-шум, поэтому значения этой величины приводят в децибеллах (дБ). Для числа A его значение в децибеллах равно $10 \log_{10} A$ дБ.

Согласно (3.41) надежная передача возможно только при

$$R < C = \frac{E}{N_0 \ln 2}$$

или

$$\frac{E_b}{N_0} > \ln 2 = 0.693 = -1.59 \quad \text{дБ.} \tag{3.42}$$

Итак, из теоремы кодирования для канала с АБГШ следует: *надежная передача информации возможно только тогда, когда отношение*

сигнал/шум на бит не меньше величины −1.59 дБ.

Насколько трудно достичь надежной передачи при таком отношении сигнал-шум?

Чтобы ответить на этот вопрос, рассмотрим передачу информации двоичными сигналами без кодирования. Надежность передачи зависит от конкретной физической среды распространения сигналов и выбранного соответствующего метода модуляции и демодуляции. При использовании для передачи информации двоичных противоположных сигналов при передаче символов 0 и 1 на выходе демодулятора наблюдаются случайные величины с плотностями распределения

$$f(y|0) = \frac{1}{\sqrt{\pi N_0}} e^{-\frac{(y+\sqrt{E})^2}{N_0}}$$

и

$$f(y|1) = \frac{1}{\sqrt{\pi N_0}} e^{-\frac{(y-\sqrt{E})^2}{N_0}}$$

соответственно. Поскольку в отсутствие кодирования каждый сигнал несет один бит информации, имеем $E = E_b$. Если приемник выносит решение о том, что передавался 0, если $y < 0$ и решение в пользу 1 в противном случае, то вероятность ошибки в одном бите равна

$$P_b = \int_0^\infty f(y|0)dy = \tag{3.43}$$

$$= \int_{-\infty}^0 f(y|1)dy = \tag{3.44}$$

$$= Q\left(\sqrt{\frac{2E_b}{N_0}}\right), \tag{3.45}$$

где использовано обозначение

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{y^2}{2}} dy. \tag{3.46}$$

График зависимости вероятности ошибки при некодированной передаче от отношения сигнал-шум приведен на Рис.3.11. На этом же рисунке показана асимптота Шеннона $E_b/N_0 = -1.59$ дБ. Из графиков, в частности, можно заметить, что для достижения вероятности ошибки 10^{-6} требуется $E_b/N_0 = 10.53$ дБ. Следовательно, при такой вероятности ошибки потенциально достижимый энергетический выигрыш кодирования составляет $1.59 + 10.53 = 12.06$ дБ, или в 16 раз! Цитируя Блэйхута,

можно сказать, что улучшать каналы связи – выбрасывать деньги на ветер. Лучше просто применить кодирование!

К сожалению, если принять во внимание некоторые важные практические ограничения, то передача при $E_b/N_0 = -1.59$ дБ окажется невозможной. Например, предположим, что информация передается двоичными сигналами, и на приемной стороне сначала выносится решение (0 или 1) о каждом сигнале, а уже затем выполняется декодирование, т.е. выбор кодового слова ближайшего к принятой из канала двоичной последовательности. Такой метод обработки сигналов называют *декодированием с жесткими решениями.*

Анализ этой ситуации (см. [12]) приводит к следующему результату: для отношения сигнал/шум на бит $E_b/N_0 = E/(N_0 R_c)$ имеем неравенство

$$\frac{E_b}{N_0} > \frac{\pi}{2} \ln 2 = 1.09 = 0.37 \text{ дБ} \tag{3.47}$$

Эти подсчеты показывают, что потери, связанные с принятием жестких решений на выходе демодулятора, составляют приблизительно 2 дБ. Асимптота, соответствующая декодированию с жесткими решениями, показана на Рис. 3.11

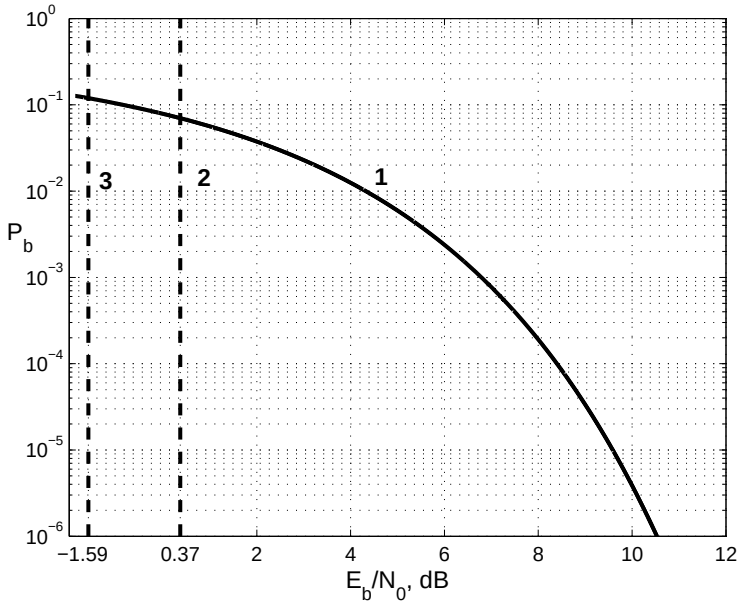


Рис. 3.11: Характеристики некодированной передачи (1) и пределы Шеннона для декодирования с жесткими (2) и мягкими (3) решениями

Анализ каналов с мягкими решениями и сравнения с современными методами кодирования можно найти в [12].

Глава 4

Непрерывные источники. Кодирование с заданным критерием качества

Дискретными мы называем множества, число элементов в которых конечно или счетно. Все остальные множества мы классифицируем как *непрерывные*. Пример – отрезок числовой оси. В этом разделе мы изучаем понятие дифференциальной энтропии, которая в данном случае уже не является характеристикой достижимой скорости кодирования, но позволяет объективно сравнивать "информативность" разных аналоговых источников. Это важный шаг на пути к изучению функций скорость-искажение для кодирования источников с заданным уровнем искажений, а также к вычислению пропускной способности непрерывных каналов

4.1 Дифференциальная энтропия. Взаимная информация для непрерывных ансамблей

Собственная информация для непрерывных источников неопределена, поскольку вероятность каждого отдельного значения стремится к нулю. Не будет ошибкой сказать, что собственная информация значений (а значит и энтропия) непрерывного источника бесконечна. Возьмем, например, число π. Оно известно математикам с точностью до миллионов разрядов, но поиск продолжается и каждый следующий разряд несет новую информацию.

Итак, формула энтропии

$$H(X) = -\sum_{x \in X} p(x) \log p(x)$$

лишена смысла для непрерывных с.в. Возникает искушение заменить сумму на интеграл, а вероятность на плотность и подумать над тем, какой смысл несет вычисленная таким образом функция.

Дифференциальной энтропией непрерывного ансамбля X с заданной на нем плотностью распределения $f(x)$ называется величина

$$h(X) = -\int_X f(x) \log f(x) dx \tag{4.1}$$

Примеры вычисления дифференциальной энтропии для наиболее часто встречающихся распределений вероятностей даны в таблице 4.1. В процессе изучения свойств дифференциальной энтропии мы поймем, почему она, с одной стороны, тоже "энтропия" (мера средней информации), хотя, с другой стороны, "дифференциальная".

При описании обобщенного гауссовского распределения использована гамма-функция $\Gamma(z)$, определяемая как

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt.$$

Легко проверить, что при $\alpha = 1$ обобщенное гауссовское распределение совпадает с экспоненциальным, при $\alpha = 2$ из него получается нормальное распределение, а при $\alpha \to \infty$ оно стремится к равномерному.

Таблица 4.1: Примеры непрерывных случайных величин

Распределение вероятностей	Плотность $f(x)$	Характеристики		
		m	σ^2	$h(X)$
Равномерное	$\frac{1}{b-a}, \quad x \in [a,b]$ $0, \quad x < a, x > b$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$	$\log(b-a)$
Экспоненциальное	$\lambda e^{-\lambda x}, \quad x \geq 0$ $0, \quad x < 0$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$	$\log \frac{e}{\lambda}$
Лапласа	$\frac{\lambda}{2} e^{-\lambda \lvert x-m \rvert}$	m	$\frac{1}{\lambda^2}$	$\log \frac{2e}{\lambda}$
Нормальное (гауссовское)	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2\sigma^2}}$	m	σ^2	$\log \sqrt{2\pi e \sigma^2}$
Обобщенное гауссовское	$\frac{\alpha\eta(\alpha,\sigma)}{2\Gamma(1/\alpha)} e^{-\eta(\alpha,\sigma)\lvert x-m \rvert^\alpha}$ $\eta(\alpha,\sigma) = \frac{1}{\sigma}\sqrt{\frac{\Gamma(3/\alpha)}{\Gamma(1/\alpha)}}$	m	σ^2	$\log \frac{2\Gamma(1/\alpha)e^{1/\alpha}}{\alpha\eta(\alpha,\sigma)}$

Свойство 4.1.1 *Дифференциальная энтропия может быть как положительной, так и отрицательной.*

Доказательство. Согласно формуле для дифференциальной энтропии равномерного распределения (см. таблицу 4.1), для с.в. распределенной равномерно на интервале длины a имеем

$$h(X) = \log a \begin{cases} \leq 0, & a \leq 1, \\ > 0, & a > 1. \end{cases}$$

□

Условимся под aX понимать с.в., полученную из X умножением ее значений на a, аналогично $a + X$ означает прибавление a к значениям из X.

Свойство 4.1.2

$$\begin{aligned} h(a + X) &= h(X); \\ h(aX) &= h(X) + \log |a|. \end{aligned}$$

Доказательство. Первое равенство легко получить из определения (4.1) с помощью простой замены переменных при интегрировании.

Чтобы получить второе тождество, вспомним известное из теории вероятностей правило вычисления плотности распределения для функций от случайных величин.

Если $Y = \{y\}$ получена из $X = \{x\}$ взаимно-однозначным преобразованием $y(x)$, то плотность $f_y(y)$ вычисляется по известной плотности $f_x(x)$ по формуле

$$f_y(y) = f_x(x(y)) \left| \frac{dx(y)}{dy} \right|. \tag{4.2}$$

Поэтому, если плотностью для X была $f(x)$, то aX будет иметь плотность $\frac{1}{a} f(\frac{x}{a})$. Подстановка этой плотности в (4.1) приводит к требуемому результату. □

Мы доказали, что изменение математического ожидания (прибавление константы к каждому значению с.в.) не изменяет дифференциальной энтропии, а умножение на константу a увеличивает ее на $\log a$. Первый из этих фактов понятен, поскольку смещение с.в. напоминает изменение нумерации или переобозначение элементов дискретного множества. Второй факт объяснить сложнее, поскольку мы привыкли к тому, что взаимно-однозначное преобразование дискретного ансамбля не изменяет его энтропии.

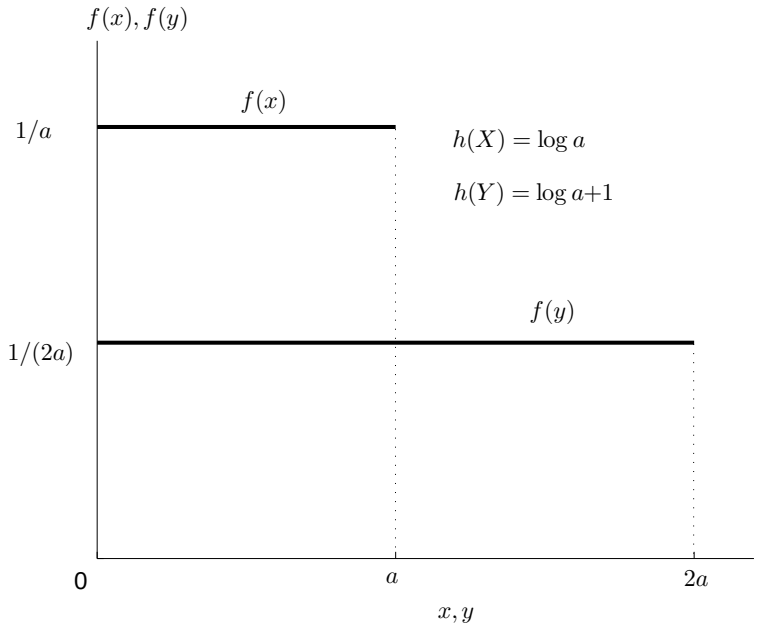


Рис. 4.1: Пример преобразования случайной величины

Рис. 4.1 поясняет изменение энтропии при масштабировании. На нем показана плотность с.в. X равномерно распределенной на интервале $[0, a]$ и с.в. $Y = 2X$. Дифференциальные энтропии этих с.в. связаны соотношением

$$h(Y) = h(X) + 1.$$

В чем смысл дополнительного бита?

Рассмотрим некоторое конкретное значение $y \in Y$. Заметим, что 1 бита достаточно, чтобы указать в каком из двух отрезков $[0, a)$ или $[a, 2a]$ находится y. После того, как значение этого бита известно, информация, содержащаяся в y совпадает с информацией, содержащейся в любом значении x. Следовательно, для передачи значений из Y нужно потратить на 1 бит больше, чем для значений ансамбля X.

Этот пример подсказывает, что дифференциальная энтропия не указывает значение среднего количества информации в сообщениях непрерывного множества, но зато дает возможность соизмерять количество информации в различных множествах, а также измерять изменение количества информации при преобразованиях ансамблей. Отсюда и название: “дифференциальная” энтропия.

Если на $XY = \{(x, y)\}$ задана двумерная плотность $f(x, y) = f(x)f(y|x)$, то по аналогии с дискретными ансамблями определяем

условную дифференциальную энтропию как

$$h(Y|X) = -\int_X \int_Y f(x,y) \log f(y|x) dxdy.$$

В точности как и для дискретного случая, имеют место следующие утверждения.

Свойство 4.1.3

$$\begin{array}{rcl} h(XY) & = & h(X) + h(Y|X), \\ h(X|Y) & \leq & h(X), \end{array}$$

равенство имеет место только для независимых ансамблей X и Y.

Относительной энтропией Кульбака-Лейблера плотности $f_1(x)$ по отношению к плотности $f_2(x)$ называется функция

$$L(f_1||f_2) = \int_X f_1(x) \log \frac{f_1(x)}{f_2(x)} dx. \tag{4.3}$$

Свойство 4.1.4 *Для любых плотностей $f_1(x)$ и $f_2(x)$*

$$L(f_1||f_2) \geq 0,$$

с равенством в том и только в том случае, когда плотности совпадают.

Доказательство аналогично доказательству для дискретных распределений (см. [12]).

Опираясь на это свойство, мы установим верхние пределы для дифференциальной энтропии для некоторых важных классов распределений вероятностей.

Свойство 4.1.5 *Если областью определения с.в. служит отрезок длины a, то для любой плотности $f(x)$*

$$h(X) \leq \log a,$$

с равенством в случае равномерного распределения.

Свойство 4.1.6 *Для неотрицательных случайных величин с математическим ожиданием m при любой плотности распределения $f(x)$*

$$h(X) \leq \log(em),$$

с равенством в случае экспоненциального распределения.

Свойство 4.1.7 *Для случайных величин с дисперсией σ^2 при любой плотности распределения $f(x)$*

$$h(X) \leq \log \sqrt{2\pi e \sigma^2},$$

с равенством в случае гауссовского распределения.

Последние три свойства устанавливают, что равномерное, экспоненциальное и гауссовское распределения являются *экстремальными* в том смысле, что они максимизируют дифференциальную энтропию при различных ограничениях на распределения вероятностей.

Мы докажем свойство 4.1.5. Свойства 4.1.6 и 4.1.7 рекомендуем доказать самостоятельно.

Доказательство свойства 4.1.5. Согласно свойству 4.1.2 без потери общности можно рассматривать интервал $[0, a]$ в качестве области определения с.в. Пусть $f(x)$ и $f_0(x)$ обозначают соответственно произвольную плотность и равномерную плотность на $[0, a]$. Рассмотрим разность

$$\begin{aligned} \log a - h(X) &= \int_0^a f(x) \log a dx + \int_0^a f(x) \log f(x) dx = \\ &= \int_0^a f(x) \log \frac{f(x)}{1/a} dx = \\ &= L(f||f_0). \end{aligned}$$

Правая часть неотрицательна в силу свойства относительной энтропии 4.1.4. □

Продолжая аналогию с дискретными ансамблями, введем понятие *средней взаимной информации между непрерывными ансамблями* X и Y

$$I(X;Y) = \int_X \int_Y f(x,y) \log \frac{f(y|x)}{f(y)} dxdy.$$

Средняя взаимная информация для непрерывных величин имеет ту же интерпретацию и те же свойства, что и для дискретных величин. Напомним те свойства, которые будут востребованы в дальнейшем.

Свойство 4.1.8

$$I(X;Y) = I(Y;X) = h(Y) - h(Y|X) = h(X) - h(X|Y);$$

Свойство 4.1.9

$$I(X;Y) \geq 0$$

с равенством только для независимых ансамблей X и Y.

Свойство 4.1.10 *I(X;Y) – выпукла вниз как функция условного распределения вероятностей* $f(y|x)$

Еще одно замечательное свойство:

Свойство 4.1.11

$$I(aX;Y) = I(X;Y).$$

Оно легко доказывается с использованием свойства 4.1.2. Больше того с помощью формулы преобразования плотностей (4.2), легко установить, что средняя взаимная информация (в отличие от дифференциальной энтропии) не изменяется при обратимом преобразовании ансамблей.

4.2 Дифференциальная энтропия случайных векторов

В данном параграфе мы будем решать задачу вычисления дифференциальной энтропии для случайных последовательностей (векторов), элементы которых – непрерывные с. в, а затем и для случайных процессов.

Рассмотрим последовательность $\boldsymbol{x} = (x_1, ..., x_n) \in X^n$. Предположим, что известна n-мерная плотность $f(\boldsymbol{x})$. Нас интересует

$$h(X^n) = -\int_{X^n} f(\boldsymbol{x}) \log f(\boldsymbol{x}) d\boldsymbol{x}.$$

Для независимых с.в. ответ прост: $h(X^n) = \sum_{i=1}^n h(X_i)$. Интересно найти нетривиальное решение задачи для содержательных моделей зависимых случайных величин.

Нам пригодится следующее правило пересчета многомерных распределений вероятностей при линейных преобразованиях случайных векторов.

Предположим, что вектор $\boldsymbol{y} = (y_1, ..., y_n) \in Y^n$ получен из $\boldsymbol{x} \in X^n$ обратимым преобразованием с помощью дифференцируемых функций $y_i = f_i(\boldsymbol{x})$. Тогда распределением вероятностей $g(\boldsymbol{y})$ для $\boldsymbol{y}$ будет

$$g(\boldsymbol{y}) = |J|^{-1} f(\boldsymbol{x}), \tag{4.4}$$

где J – якобиан преобразования

$$J = \det \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}.$$

Из этого правила получаем обобщение свойства 4.1.2 на произвольные линейные преобразования векторов.

Теорема 4.1 *Для любой невырожденной матрицы A размера $n \times n$ дифференциальная энтропия ансамбля $Y^n = \{\boldsymbol{y} = \boldsymbol{x}A, \boldsymbol{x} \in X^n\}$ равна*

$$h(Y^n) = h(X^n) + \log|\det A|.$$

Доказательство. На основании (4.4) имеем

$$\begin{aligned} h(Y^n) &= \mathbf{M}\left[-\log g(\boldsymbol{y})\right] = \\ &= \mathbf{M}\left[-\log\left(f(\boldsymbol{x})|\det A|^{-1}\right)\right] = \\ &= h(X^n) + \log|\det A|. \end{aligned}$$

□

Поскольку, как мы уже выяснили, энтропия не зависит от математического ожидания, временно ограничимся рассмотрением случайных векторов с нулевым математическим ожиданием. Обозначим через $K_{\boldsymbol{x}}$ корреляционную матрицу такого вектора $\boldsymbol{x}$,

$$K_{\boldsymbol{x}} = \mathbf{M}[\boldsymbol{x}'\boldsymbol{x}].$$

Ее элементами являются корреляционные моменты $K_{ij} = \mathbf{M}[x_i x_j]$, $i, j = 1, \ldots, j$.

При линейном преобразовании $\boldsymbol{y} = \boldsymbol{x}A$ корреляционная матрица преобразованного вектора примет вид

$$K_{\boldsymbol{y}} = \mathbf{M}\left[A'\boldsymbol{x}'\boldsymbol{x}A\right] = A'K_{\boldsymbol{x}}A. \tag{4.5}$$

Например, если компоненты вектора $\boldsymbol{x}$ были независимы и имели единичные дисперсии, т.е. $K_{\boldsymbol{x}} = I_n$, где I_n – единичная матрица размера $n \times n$, то после преобразования $\boldsymbol{y} = \boldsymbol{x}A$ получим вектор $\boldsymbol{y}$ с корреляционной матрицей

$$K_{\boldsymbol{y}} = A'A, \quad \det K_{\boldsymbol{y}} = (\det A)^2. \tag{4.6}$$

Из теории вероятностей мы знаем, что линейная комбинация гауссовских с.в. – гауссовская с.в., а из линейной алгебры известно, что симметрическая матрица K может быть представлена в виде произведения матриц

$$K = T'\Lambda T, \tag{4.7}$$

в котором матрица T составлена из собственных векторов матрицы K, а Λ – диагональная матрица, на главной диагонали которой располагаются собственные числа матрицы K.

Сопоставляя (4.5), (4.6) и (4.7), приходим к заключению, что гауссовский вектор с любой заданной наперед корреляционной матрицей может быть получен с помощью подходящего линейного преобразования из вектора независимых гауссовских с.в. с единичными дисперсиями, и, наоборот, найдется преобразование, приводящее произвольный гауссовский вектор к вектору из независимых гауссовских с.в. с единичными дисперсиями.

Напомним общее выражение для n-мерной плотности гауссовского вектора с корреляционной матрицей K и вектором математических ожиданий $\boldsymbol{m}$

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}\det K^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})K^{-1}(\boldsymbol{x}-\boldsymbol{m})^T}, \tag{4.8}$$

где T – символ транспонирования. Теорема 4.1 указывает простой путь к получению дифференциальной энтропии гауссовского вектора. Результат вычислений сформулируем в виде следующей теоремы.

Теорема 4.2 *Дифференциальная энтропия гауссовского вектора с корреляционной матрицей K равна*

$$h(X^n) = \frac{1}{2}\log\left((2\pi e)^n \det K\right). \tag{4.9}$$

Доказательство. Найдем разложение вида (4.7) для K. Обозначим через $\lambda_1, ..., \lambda_n$ диагональные элементы матрицы Λ, т.е. собственные числа матрицы K. Применим преобразование $\boldsymbol{x} = \boldsymbol{y}A$ к вектору $\boldsymbol{y}$, компоненты которого – независимые гауссовские с.в. с дисперсиями $\lambda_1, ..., \lambda_n$. Применив Теорему 4.1 после простых выкладок получим (4.9). □

Следствие 4.3 *Дифференциальная энтропия гауссовского вектора из n независимых компонент с дисперсиями $\sigma_1^2, ..., \sigma_n^2$ равна*

$$h_n(X) = \frac{1}{2}\log(2\pi e) + \sum_{i=1}^{n}\log\sigma_i. \tag{4.10}$$

Напомним, что согласно свойству 4.1.7 гауссовская с.в. имеет наибольшую энтропию среди всех с.в. с заданной дисперсией. Оказывается, что аналогичным свойством обладает и многомерное гауссовское распределение.

Теорема 4.4 *Дифференциальная энтропия произвольного вектора с корреляционной матрицей K удовлетворяет неравенству*

$$h(X^n) \le \frac{1}{2}\log\left((2\pi e)^n \det K\right). \tag{4.11}$$

Доказательство. Без потери общности можно рассматривать векторы с нулевым математическим ожиданием. Пусть $f(\boldsymbol{x})$ и $g(\boldsymbol{x})$ – соответственно гауссовское и произвольное распределение вероятностей на X^n, причем обоим распределениям соответствуют одинаковые корреляционные матрицы $K = \{k_{ij}\}, i, j = 1, ..., n$. Дифференциальные энтропии этих распределений обозначим соответственно через $h(f)$ и $h(g)$. Прежде всего заметим, что

$$-\int_{X^n} g(\boldsymbol{x}) \log f(\boldsymbol{x}) d\boldsymbol{x} = \frac{1}{2} \log \left((2\pi e)^n \det K \right). \tag{4.12}$$

Действительно, подставив в левую часть (4.12) n-мерную плотность (4.8), получим

$$\begin{aligned}
& - \int_{X^n} g(\boldsymbol{x}) \log f(\boldsymbol{x}) d\boldsymbol{x} = \\
= & - \frac{1}{2} \log \left((2\pi)^n \det K \right) - \frac{\log e}{2} \int_{X^n} g(\boldsymbol{x}) \boldsymbol{x} K^{-1} \boldsymbol{x}^T d\boldsymbol{x} = \\
\overset{\text{(a)}}{=} & - \frac{1}{2} \log \left((2\pi)^n \det K \right) - \frac{\log e}{2} \int_{X^n} g(\boldsymbol{x}) \sum_{i=1}^{n} \sum_{j=1}^{n} x_i \tilde{k}_{ij} x_j d\boldsymbol{x} = \\
\overset{\text{(b)}}{=} & - \frac{1}{2} \log \left((2\pi)^n \det K \right) - \frac{\log e}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \tilde{k}_{ij} \int_{X^n} g(\boldsymbol{x}) x_i x_j d\boldsymbol{x} = \\
\overset{\text{(c)}}{=} & - \frac{1}{2} \log \left((2\pi)^n \det K \right) - \frac{\log e}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \tilde{k}_{ij} k_{ij} = \\
\overset{\text{(d)}}{=} & - \frac{1}{2} \log \left((2\pi)^n \det K \right) - \frac{n}{2} \log e = \\
= & \frac{1}{2} \log \left((2\pi e)^n \det K \right).
\end{aligned}$$

В (a) мы приняли обозначение $\tilde{k}_{ij}$ для элементов матрицы K^{-1}, в (b) поменяли порядок интегрирования и суммирования. Под знаком интеграла получили корреляционные моменты, это учтено в (c). Внутренняя сумма равна скалярному произведению строки матрицы K на столбец K^{-1} с тем же номером. Поэтому каждая внутренняя сумма равна 1. Поэтому имеет место переход (d).

Теперь из (4.12) вытекает, что разность левой и правой частей (4.11) есть

$$-\int_{X^n} g(\boldsymbol{x}) \log \frac{g(\boldsymbol{x})}{f(\boldsymbol{x})} = -L(g||f) \leq 0,$$

где неравенство основано на свойстве 4.1.4 относительной энтропии. Отсюда следует утверждение теоремы. □

4.3 Дифференциальная энтропия стационарных процессов дискретного времени

От энтропии векторов один шаг до энтропии стационарных случайных процессов дискретного времени. Напомним, как этот шаг был сделан для дискретных процессов. Мы ввели в рассмотрение энтропии $H_n(X) = H(X^n)/n$ и $H(X|X^n)$ (см. параграф 1.5) и рассматривали их пределы при $n \to \infty$. Эти пределы совпадают и были названы энтропией дискретного стационарного источника.

Точно также мы определяем *дифференциальную энтропию на сообщение* для стационарного процесса как

$$h_n(X) = \frac{h(X^n)}{n}$$

и *дифференциальной энтропией случайного процесса* дискретного времени называем предел

$$h_\infty(X) = \lim_{n\to\infty} h_n(X). \tag{4.13}$$

Введя обозначения $h(X|X^n) = h(X_{n+1}|X_1...X_n)$ и $h(X|X^\infty) = \lim_{n\to\infty} h(X|X^n)$, по аналогии с теоремой 1.4 можем установить, что для произвольного стационарного процесса

$$h_\infty(X) = h(X|X^\infty).$$

Корреляционной функцией стационарного процесса мы называем функцию

$$K(n) = M[(x_i - m_x)(x_{i+n} - m_x)].$$

Из теории вероятностей мы знаем, что корреляционная функция – четная функция аргумента n и принимает максимальное значение равное дисперсии процесса при $n = 0$, т.е. $K(0) = \sigma_x^2$.

По известной корреляционной функции для каждого n можно построить симметрическую корреляционную матрицу $K_n = K_{ij}, i, j, = 1, ..., n$, элементы которой равны $K_{ij} = K(i - j) = K(j - i)$

Теорема 4.2 дает возможность подсчитать $h_n(X)$ для гауссовского процесса. Подставив (4.9) в 4.13 и перейдя к пределу, получим

$$h_\infty(X) = \frac{1}{2}\log(2\pi e) + \lim_{n\to\infty} \frac{\log\det K_n}{n}. \tag{4.14}$$

Итак, подсчет энтропии сводится к подсчету последовательности определителей корреляционных матриц процесса. Если вспомнить о том, что определитель равен произведению собственных чисел матрицы, то, на самом деле, нужно построить последовательность собственных чисел и найти ее предел при $n \to \infty$, что тоже непросто.

Нетривиальным примером процесса, для которого вычисления можно выполнить по формуле (4.14), служит так называемый авторегрессионный процесс. Его анализ приведен в [12].

Рассмотрим более общий подход, предложенный более полувека назад А.Н. Колмогоровым.

Спектральной плотностью мощности стационарного случайного процесса с функцией корреляции $K(n)$ называется функция

$$S(\omega) = \sum_{n=-\infty}^{\infty} K(n) e^{-in\omega}, \quad \omega \in (-\pi, \pi]. \tag{4.15}$$

В соответствии с этим определением спектральная плотность мощности – преобразование Фурье корреляционной функции. Функция $S(\omega)$ – неотрицательная вещественная четная функция аргумента ω.

Обсудим физический смысл понятия спектральной плотности мощности по отношению к случайным процессам дискретного времени. Аргумент ω привычно было бы интерпретировать как круговую частоту сигнала и тогда она измерялась бы в радианах/с. Однако, в (4.15) ω – безразмерная величина. Если считать, что рассматриваемый стационарный процесс получен измерениями значений процесса непрерывного времени (дискретизацией) через равные интервалы времени (период дискретизации) T секунд, то значения ω связаны со значениями частот f, измеряемыми в Герцах, соотношением $\omega = 2\pi f T$. Частота дискретизации равна $f_d = 1/T$. Следовательно, интервалу $(-\pi, \pi]$ для ω соответствует интервал $(-f_d/2, f_d/2]$ для "настоящей" частоты f. Согласно теореме отсчетов (теореме Котельникова-Шеннона) этот интервал содержит весь спектр сигнала, который может быть представлен своими отсчетами, измеренными с периодом T. Итак, ω – круговая частота, нормированная по отношению к частоте дискретизации. Соответственно, интеграл от $S(\omega)$ по малому интервалу можно трактовать как дисперсию (энергию) случайного процесса в соответствующем частотном диапазоне.

В соответствии с Теоремой 4.1 обратимое линейное преобразование с определителем равным единице не изменяет дифференциальной энтропии ансамбля. В качестве такого преобразования можно выбрать преобразование Фурье, примененное к последовательностям большой длины

n. Известно, что с увеличением n коэффициенты корреляции спектральных компонент преобразования Фурье стремятся к нулю. Учитывая приведенные выше замечания о спектральной плотности мощности и применяя к спектральным компонентам Следствие 4.3, приходим к фундаментальному результату, впервые полученному А.Н. Колмогоровым [52].

Теорема 4.5 *Дифференциальная энтропия стационарного гауссовского процесса со спектральной плотностью мощности $S(\omega)$ равна*

$$h_\infty(X) = \frac{1}{2}\log(2\pi e) + \frac{1}{4\pi}\int_{-\pi}^{\pi}\log S(\omega)d\omega. \tag{4.16}$$

Строгое доказательство Теоремы 4.5 требует значительных усилий и опирается на довольное сложный математический аппарат. С разными подходами к доказательству можно познакомится в [18], [15] и [3].

Пример применения Теоремы 4.5 к авторегрессионному процессу первого порядка детально разобран в [12].

4.4 Меры искажения. Постановка задачи кодирования

Мы приступаем к рассмотрению одного из наиболее актуальных сегодня разделов теории информации – к кодированию источников при заданных ограничениях на точность восстановления информации декодером. Такие методы сжатия сегодня востребованы во многих областях обработки, хранения и передачи мультимедиа данных.

Для потребителя системы передачи или хранения информации критерием качества работы кодера и декодера является *субъективное* восприятие искажений. С точки зрения потребителя искажения могут быть характеризованы как незаметные, допустимые, грубые и т.п. Разработчики систем кодирования также стараются использовать субъективные критерии в своей работе, для чего к тестированию аппаратуры привлекаются группы экспертов. Каждое такое измерение качества требует значительных затрат времени и финансовых средств, поэтому на этапе разработки кодеров необходимо использовать *объективные* критерии качества.

Конечно, следует выбирать критерии так, чтобы уровень объективного качества был монотонно связан с уровнем субъективного качества. Однако, в большинстве случаев “повергнуть алгебре язык гармоний” не

удается. Приходится, как это часто бывает в теоретических исследованиях, накладывать на множестве мер искажений такие ограничения, которые позволили бы с разумной сложностью найти аналитические решения интересующих нас задач.

Итак, рассмотрим некоторое множество $X = \{x\}$, элементы которого – сообщения источника и второе множество, $Y = \{y\}$, которое будем называть *аппроксимирующим. Мера искажения* $d_n(\boldsymbol{x}, \boldsymbol{y})$ определяется как функция на множестве $X^n \times Y^n = \{(\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{x} \in X^n, \boldsymbol{y} \in Y^n\}$, удовлетворяющая следующим ограничениям:

- Для любых $\boldsymbol{x} = (x_1, ..., x_n)$, $\boldsymbol{y} = (y_1, ..., y_n)$

$$d_n(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n} \sum_{i=1}^{n} d(x_i, y_i), \tag{4.17}$$

  где $d(x, y)$ – так называемая *побуквенная мера искажения.*

- $$d(x, y) \geq 0 \text{ для всех } x \in X, y \in Y.$$

Согласно этим условиям, при кодировании последовательностей сообщений источника, мы рассматриваем только такие меры искажения которые вычисляются как среднее значение побуквенной меры по всем сообщениям последовательностей, причем побуквенная мера обязательно неотрицательна. Оба ограничения вполне естественны. Первое нормирует суммарную ошибку в последовательности сообщений по отношению к длине последовательности и позволяет объективно сравнивать кодеры, обрабатывающие последовательности разной длины. Второе ограничение тоже понятно, оно означает, что аппроксимация приводит к потере точности и $d(x, y)$ указывает величину потери, связанной с заменой точного значения x его аппроксимирующим значением y.

Приведем несколько примеров.

Пример 4.4.1 *Вероятностная (Хэммингова) мера искажения.* Пусть $X = \{0, ..., M-1\}$ – дискретное множество, и $Y = X$. Положим

$$d(x, y) = \begin{cases} 0, & x = y; \\ 1, & x \neq y. \end{cases}$$

Соответствующая мера (4.17) представляет собой долю "ошибок" при выдаче получателю $\boldsymbol{y}$ вместо истинной последовательности $\boldsymbol{x}$. При большой длине последовательности $d_n(\boldsymbol{x}, \boldsymbol{y})$ можно рассматривать как эмпирическую оценку вероятности ошибочного символа, отсюда – название "вероятностная мера"

Пример 4.4.2 *Абсолютная мера искажения.* Пусть X – произвольное множество вещественных чисел и $Y = X$. Положим

$$d(x, y) = |x - y| .$$

Соответствующая мера (4.17) представляет собой нормированную сумму модулей отклонения компонент $\boldsymbol{y}$ от компонент последовательности $\boldsymbol{x}$.

Пример 4.4.3 *Квадратическая мера искажения.* Пусть X – произвольное множество вещественных чисел и $Y = X$. Положим

$$d(x, y) = (x - y)^2 .$$

Соответствующая мера (4.17) представляет собой среднеквадратическую ошибку аппроксимации $\boldsymbol{x}$ последовательностью $\boldsymbol{y}$. Последняя мера наиболее часто используется на практике. Ее можно рассматривать как мощность шума, наложенного на исходную последовательность источника в результате кодирования. Отношение мощности кодируемого сигнала к мощности шума, порожденного кодером, называют *отношением сигнал/шум.*

Отметим, что во всех примерах предполагалось, что сообщения и аппроксимирующие значения выбираются из одного и того же множества.

Рассмотрим теперь приведенную на рис.4.2 модель системы, осуществляющей кодирование при заданном критерии качества.

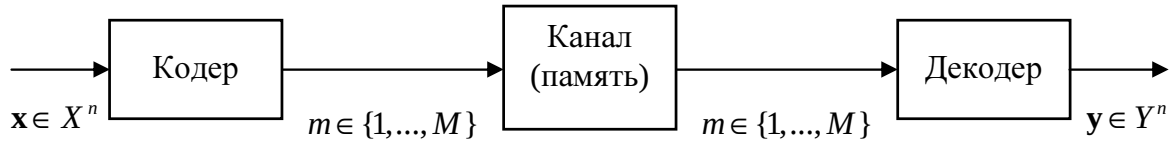


Рис. 4.2: Схема кодирования с заданным критерием качества

Последовательность сообщений источника разбивается на блоки длины n. Каждый из блоков должен быть аппроксимирован, по возможности с наименьшим искажением, некоторой последовательностью $\boldsymbol{y} \in Y^n$. Будем считать, что аппроксимирующие последовательности (*кодовые слова*) выбираются из дискретного подмножества $B = \{\boldsymbol{b}_1, ..., \boldsymbol{b}_M\} \subseteq Y^n$, которое называют *кодом* или (в технической литературе) *кодовой книгой.* Объем книги – некоторое конечное число $M = |B|$. Операция кодирования – это выбор наилучшего слова из кода. Ее можно выполнить,

поочередно перебирая кодовые слова и подсчитывая каждый раз величину ошибки аппроксимации. Передаче по каналу или записи в память подлежит номер m того слова, которое обеспечивает наименьшую ошибку.

Декодер хранит точную копию кода B и по полученному индексу m воспроизводит аппроксимирующую последовательность $\boldsymbol{y} = \boldsymbol{b}_m$.

Поскольку для передачи индекса m достаточно $\log M$ бит (для сокращения записи мы считаем M степенью двойки), *скоростью кода* источника называется величина

$$R = \frac{\log M}{n} \quad \text{бит / сообщение источника.}$$

При фиксированном коде B и заданной вероятностной модели источника можно усреднением по всем $\boldsymbol{x} \in X^n$ подсчитать *среднюю ошибку*

$$D = \mathbf{M}[d_n(\boldsymbol{x}, \boldsymbol{y}(\boldsymbol{x}))],$$

где через $\boldsymbol{y}(\boldsymbol{x})$ обозначена последовательность $\boldsymbol{y}$, которую кодер выбирает в качестве аппроксимирующей последовательности для $\boldsymbol{x}$.

Таким образом, основными характеристиками описанной системы кодирования являются два числа: скорость R и средняя ошибка D. Наша задача – закодировать последовательность сообщений с наименьшей скоростью и с наименьшей ошибкой.

Понятно, что уменьшение ошибки потребует увеличения числа кодовых слов. Если же мы захотим уменьшить скорость кода, придется уменьшить число кодовых слов, что приведет к увеличению средней ошибки. Таким образом, два требования противоречат друг другу. Придется зафиксировать один из параметров, например, допустимое искажение D, и добиваться наименьшей скорости при заданном искажении. Для некоторого алгоритма или класса алгоритмов построения кодов и заданных процедур кодирования, изменяя требования к ошибке D, можно получить функцию $R(D)$, называемую *функцией скорость-искажение.*

Более формально, функцией скорость-искажение $R(D)$ для заданной модели источника называется функция вида

$$R(D) = \inf_n \min_{B: D(B) \le D} R_n(B), \tag{4.18}$$

где нижняя рань берется по длинам кодируемых последовательностей, а минимум – по таким кодам последовательностей длины n, которые удовлетворяют ограничению на допустимую ошибку D.

Аналогично можно определить функцию искажение-скорость $D(R)$.

Очевидно, определение (4.18) невозможно использовать для подсчета функции скорость-искажение. Желательно (как это было сделано для кодирования источников без потерь и для каналов связи) научиться вычислять $R(D)$ по модели канала без перебора по всевозможным кодам источника, аналогично тому, как для кодирования без потерь минимальная скорость кода определяется энтропией источника, а максимальная достижимая скорость передачи информации по каналу связи равна его пропускной способности.

Вернемся к схеме на рис. 4.2. Модель источника задана и тем самым задано распределение вероятностей на последовательностях из X^n. Для определенности, будем считать источник непрерывным и описывать модель источника плотностью распределения вероятностей $f(\boldsymbol{x}), \boldsymbol{x} \in X^n$. Заметим, что среднее количество информации об X^n, доставляемой получателю декодером может быть подсчитано как взаимная информация $I(X^n; Y^n)$. Соответственно, затраты в битах в расчете на одну букву источника составляют $I(X^n; Y^n)/n$ бит. Все промежуточные блоки от входа до выхода схемы на рис. 4.2 можно описать в виде условного распределения вероятностей $\varphi_n(\boldsymbol{y}|\boldsymbol{x})$. Многообразие таких плотностей полностью включает в себя все мыслимые схемы кодирования, но нас интересуют только такие плотности, при которых выполняется требование к допустимой ошибке, которая в данном случае вычисляется как

$$D(\varphi_n) = \int_{X^n} \int_{Y^n} f(\boldsymbol{x}) \varphi_n(\boldsymbol{y}|\boldsymbol{x}) d_n(\boldsymbol{y}, \boldsymbol{x}) d\boldsymbol{x} d\boldsymbol{y}.$$

Теоретической функцией скорость-искажение называется величина

$$H(D) = \inf_n \min_{\varphi_n : D(\varphi_n) \le D} \frac{1}{n} I(X^n; Y^n). \tag{4.19}$$

После ознакомления с предыдущими главами, не станет неожиданным то, что для большинства моделей источников функции $R(D)$ (4.18) и $H(D)$ (4.19) совпадают. Это означает, что при заданном D скорости сколь угодно близкие к $H(D)$ (но большие $H(D)$) достижимы, а при скорости меньшей $H(D)$ средняя ошибка неизбежно будет больше D.

Эти утверждения мы сформулируем, как всегда в виде двух теорем кодирования – прямой и обратной. Но прежде, чем мы приступим к формулировке и доказательству теорем кодирования, мы изучим свойства функции $H(D)$ и рассмотрим несколько важных моделей источников и мер искажения, для которых $H(D)$ может быть вычислена в явной форме как функция параметров модели.

4.5 Свойства функции скорость-искажение

Прежде, чем с помощью формулы (4.19) будут получены какие-нибудь полезные результаты, ее нужно существенно упростить. В данном параграфе мы изучим некоторые свойства $H(D)$, которые позволят понять ее поведение и получить простые выражения для этой функции для широкого класса моделей.

Первое свойство очень простое.

Свойство 4.5.1 $H(D) \geq 0$

Доказательство предоставляем читателю в качестве упражнения.

Свойство 4.5.2 $H(D)$ – *невозрастающая функция аргумента* D.

Доказательство. С увеличением D расширяется область поиска минимума в (4.19), при этом результат поиска минимума, очевидно, не увеличивается. □

Свойство 4.5.3 *Для стационарного источника без памяти*

$$H(D) = \min_{\varphi: D(\varphi)) \leq D} I(X;Y), \tag{4.20}$$

где $\varphi = \varphi(y|x)$ *– одномерное условное распределение на* Y *при известном* $x \in X$.

Доказательство. Имеем

$$\begin{aligned} I(X^n;Y^n) &= h(X^n) - h(X^n|Y^n) = \\ &\overset{\text{(a)}}{=} \sum_{i=1}^{n} h(X_i) - \sum_{i=1}^{n} h(X_i|X_1...X_{i-1}Y_1...Y_n) \geq \\ &\overset{\text{(b)}}{\geq} \sum_{i=1}^{n} h(X_i) - \sum_{i=1}^{n} h(X_i|Y_i) = \\ &= \sum_{i=1}^{n} I(X_i;Y_i), \end{aligned} \tag{4.21}$$

где (a) использует независимость букв источника, а (b) – неубывание условной энтропии с увеличением числа условий. Равенство в (4.21) достигается при

$$\varphi_n(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{n} \varphi^{(i)}(y_i|x_i), \tag{4.22}$$

где в обозначении $\varphi^{(i)}$ учтено, что условная плотность может, вообще говоря, зависеть от индекса i.

Поскольку нас интересует зависимость взаимной информации $I(X^n;Y^n)$ от условного распределения вероятностей φ_n, изменим на время обозначения: $I(X^n;Y^n)=I(\varphi_n)$. Так, (4.21) примет вид

$$\frac{1}{n}I(\varphi_n)\geq\frac{1}{n}\sum_{i=1}^{n}I(\varphi^{(i)}).$$

В силу выпуклости $\cup$ средней взаимной информации относительно условных распределений

$$\frac{1}{n}I(\varphi_n)\geq I\left(\frac{1}{n}\sum_{i=1}^{n}\varphi^{(i)}\right)=I(\varphi_0),\ \text{где}\ \varphi_0=\frac{1}{n}\sum_{i=1}^{n}\varphi^{(i)},\tag{4.23}$$

причем равенство имеет место, если $\varphi_0=\varphi^{(i)}$, при всех $i=1,...,n$.

Из (4.21) – (4.23) следует, что наименьшее значение $I(X^n;Y^n)$ достигается в том случае, когда условное распределение φ_n представляет собой произведение n одинаковых одномерных плотностей, вычисленных как среднее одномерных плотностей, полученных из φ_n. Чтобы завершить доказательство теоремы, осталось доказать, что такое распределение удовлетворяет ограничению на ошибку D, если исходное распределение ему удовлетворяло.

Простые выкладки

$$\begin{aligned}
D(\varphi_n) &= \mathbf{M}\left[d_n(\boldsymbol{x},\boldsymbol{y})\right]= \\
&= \mathbf{M}\left[\frac{1}{n}\sum_{i=1}^{n}d(x_i,y_i)\right]= \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbf{M}\left[d(x_i,y_i)\right]= \\
&= \frac{1}{n}\sum_{i=1}^{n}\int_X\int_Y f(x)\varphi^{(i)}(y|x)d(x,y)dxdy= \\
&= \int_X\int_Y f(x)\left(\frac{1}{n}\sum_{i=1}^{n}\varphi^{(i)}(y|x)\right)d(x,y)dxdy= \\
&= \int_X\int_Y f(x)\varphi_0(y|x)d(x,y)dxdy= \\
&= D(\varphi_0)
\end{aligned}$$

подтверждают, что при подстановке

$$\varphi_n(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i=1}^{n} \varphi_0(y_i|x_i),$$

ошибка не изменится. При этом

$$\frac{1}{n} I(X^n; Y^n) = I(\varphi_0) = I(X; Y),$$

где $I(X;Y)$ вычислено при $\varphi(x|y) = \varphi_0(x|y)$. Теперь из определения (4.19) следует требуемый результат. □

Свойство 4.5.4 *$H(D)$ выпуклая $\cup$ функция аргумента D.*

Доказательство. Для сокращения записи доказательства рассмотрим случай источника без памяти, т.е. считаем, что $H(D)$ вычисляется по формуле (4.20). Нужно доказать, что для любых D_1, D_2 и $\alpha \in [0,1]$

$$H(D_\alpha) \le \alpha H(D_1) + (1-\alpha) H(D_2),$$

где использовано обозначение $D_\alpha = \alpha D_1 + (1-\alpha) D_2$. Обозначим через φ_1 и φ_2 те условные распределения, на которых достигаются соответственно $H(D_1)$ и $H(D_2)$ и положим $\varphi_\alpha = \alpha\varphi_1 + (1-\alpha)\varphi_2$. Доказываемое утверждение вытекает из следующей цепочки преобразований:

$$\begin{aligned} H(D_\alpha) &\overset{\text{(a)}}{\le} I(\varphi_\alpha) = \\ &= I(\alpha\varphi_1 + (1-\alpha)\varphi_2) \le \\ &\overset{\text{(b)}}{\le} \alpha I(\varphi_1) + (1-\alpha) I(\varphi_2) = \\ &= \alpha H(D_1) + (1-\alpha) H(D_2). \end{aligned}$$

Здесь (a) имеет место потому, что на распределении φ_α не обязательно достигается минимум взаимной информации $I(X;Y)$ равный $H(D_\alpha)$; (b) следует из выпуклости $\cup$ средней взаимной информации. □

Свойство 4.5.5 *Для произвольного стационарного источника $H(D) = 0$ при*

$$D \ge D_0 = \min_y \int_X f(x) d(y,x) dx \tag{4.24}$$

Доказательство. Пусть y_0 обозначает тот элемент y, на котором достигается минимум в (4.24). В качестве условной плотности $\varphi_(\boldsymbol{y}|\boldsymbol{x})$ выберем плотность, с вероятностью 1 сопоставляющую последовательность $\boldsymbol{y}_0 = (y_0, ..., y_0)$ любой последовательности $\boldsymbol{x} \in X^n$. Поскольку последовательности $\boldsymbol{x}$ и $\boldsymbol{y}$ независимы, $I(X^n, Y^n) = 0$ и, следовательно, $H(D) = 0$. Средняя ошибка при этом равна величине D_0, определенной формулой (4.24). □

Итак, в данном параграфе мы узнали, что $H(D)$ – выпуклая $\cup$ монотонно убывающая функция неотрицательная функция, которая становится равной нулю при некоторой ошибке. Конкретные примеры функций скорость-искажение будут приведены в следующим параграфе на рис. 4.4 и 4.5.

Для источников без памяти мы получили формулу (4.20), которая очень похожа на формулу для пропускной способности канала без памяти. Вместо максимума – минимум и вместо поиска оптимального распределения на входе – поиск среди условных распределений. В действительности, поиск функции скорость-искажение чуть сложнее, поскольку экстремум – условный, и ответ – функция, а не число.

Заметим, что для источников с памятью свойство 4.5.5 дает не самую точную оценку той минимальной ошибки, начиная с которой возможна аппроксимация источника с нулевой скоростью (без передачи какой-либо информации о сообщениях источника).

4.6 Простые примеры вычисления функции скорость-искажение

Все определения и утверждения в предыдущем параграфе приводились для непрерывного источника, но в равной степени могли быть записаны для дискретного ансамбля и соответственно подобранного критерия качества.

Пример 4.6.1 *Двоичный источник.*

Начнем с примера двоичного ансамбля $X = \{0, 1\}$ и вероятностного критерия качества. Вероятности символов положим равными $p(0) = 1 - p, p(1) = p$. Источник порождает последовательности $\boldsymbol{x} \in X^n$ независимых одинаково распределенных двоичных символов. Эти последовательности аппроксимируются двоичными последовательностями $\boldsymbol{y} \in \{0, 1\}^n$ так, чтобы при заданной скорости аппроксимирующего кода

минимизировать среднюю ошибку

$$D = \mathbf{M}[d_n(\boldsymbol{x}, \boldsymbol{y})] = \frac{1}{n}\mathbf{M}[d_H(\boldsymbol{x}, \boldsymbol{y})],$$

где $d_H(\boldsymbol{x}, \boldsymbol{y})$ – расстояние Хэмминга (число несовпадающих позиций в $\boldsymbol{x}$ и $\boldsymbol{y}$).

В соответствии со свойством 4.5.3 функцию скорость-искажение следует искать по формуле

$$H(D) = \min_{\varphi: D(\varphi)) \le D} I(X;Y),$$

где условные вероятности $\varphi(y|x)$ задаются всего двумя числами, например, $\varphi(y=0|x=0)$ и $\varphi(y=0|x=1)$. Хотя поиск условного экстремума по этим двум параметрам не кажется сверхсложной задачей, мы не пойдем по этому прямолинейному пути. Вместо этого мы сначала найдем нижнюю оценку взаимной информации, а затем покажем, что эта оценка достижима при некотором φ.

Введем обозначение $\oplus$ для операции сложения по модулю два и заметим, что

$$x \oplus y = \begin{cases} 0, & x = y; \\ 1, & x \neq y. \end{cases}$$

Это означает, что ансамбль $X \oplus Y = \{x \oplus y\}$ – ансамбль ошибок при аппроксимации источника X символами из Y. При средней ошибке равной D взаимную информацию $I(X;Y)$ можно оценить следующим образом:

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) = \\ &\overset{\text{(a)}}{=} H(X) - H(X \oplus Y|Y) \geq \\ &\overset{\text{(b)}}{\geq} H(X) - H(X \oplus Y) = \\ &= \eta(p) - \eta(D). \end{aligned} \tag{4.25}$$

Здесь в (a) замена X на $X \oplus Y$ при условии Y представляет собой детерминированное преобразование и поэтому не меняет энтропии. Неравенство (b) имеет место потому, что условная энтропия не может быть больше безусловной, и $\eta(a) = -a\log a - (1-a)\log(1-a)$ – двоичная энтропия.

Предполагая, что в (4.25) получено правильное выражение для функции скорость-искажение, подберем к заданному ансамблю источника X такой ансамбль Y, для которого неравенства в (4.25) можно заменить на точные равенства. Такая пара ансамблей показана на Рис. 4.3.

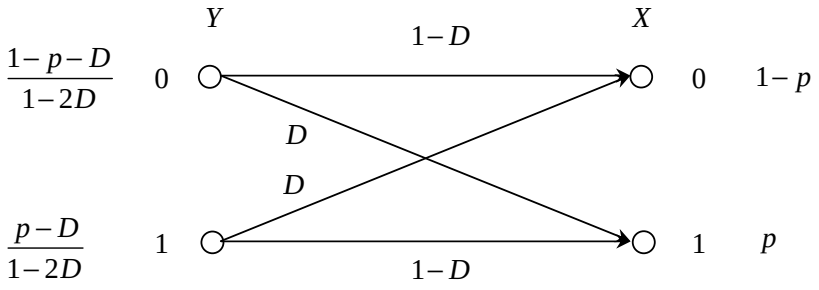


Рис. 4.3: Совместное распределение X и Y для двоичного источника.

Положим вероятности аппроксимирующих символов равными $p(y = 0) = 1 - \lambda$, $p(y = 1) = \lambda$. Вероятности символов $x \in X$ известны: $p(x = 0) = 1 - p$, $p(x = 1) = p$. Условные вероятности выбраны исходя из требований к ошибке: $p(y = 0|x = 1) = p(y = 1|x = 0) = D$. Поэтому неизвестный параметр λ находим из уравнения

$$(1 - \lambda)(1 - D) + \lambda D = 1 - p.$$

Решением является

$$\lambda = \frac{p - D}{1 - 2D},$$

что и показано на Рис. 4.3.

Теперь заметим, что при $D = \min\{p, 1 - p\}$ правая часть (4.25) обращается в нуль. Это легко объяснить. Среднюю ошибку, равную этой величине можно получить, всегда используя в качестве аппроксимирующего символа один и тот же символ 0 либо 1, в зависимости от того, вероятность какого из них больше.

Итак, ансамбль, на котором достигается нижняя граница (4.25) найден и тем самым найдена функция скорость-искажения. При этом мы так и не указали в явном виде функцию φ, на которой достигается минимум взаимной информации. При желании эту функцию можно легко получить, воспользовавшись рисунком 4.3.

Сформулируем полученный в этом примере результат в виде теоремы.

Теорема 4.6 *Функция скорость-искажение двоичного источника независимых символов с вероятностью единицы равной p при вероятностной мере искажения равна*

$$H(D) = \begin{cases} \eta(p) - \eta(D), & 0 \le D \le \min\{p, 1 - p\} \\ 0, & D > \min\{p, 1 - p\} \end{cases} \qquad (4.26)$$

Типичный график $H(D)$ для двоичного источника показан на Рис. 4.4.

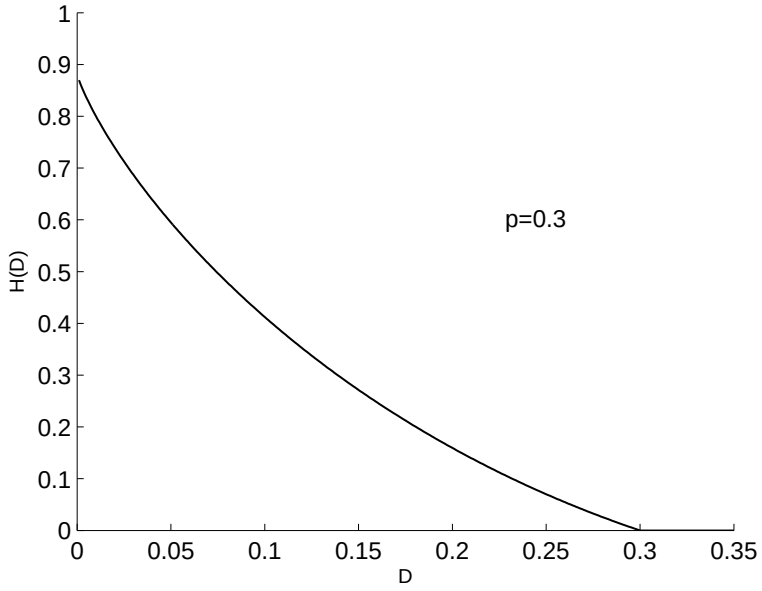


Рис. 4.4: Функция скорость-искажение для двоичного источника.

Пример 4.6.2 *Гауссовский источник*

Рассмотрим последовательность независимых одинаково распределенных гауссовских с.в. с нулевым математическим ожиданием и дисперсией σ^2, т.е. с плотностью

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}},$$

и вычислим функцию скорость-искажение при квадратическом критерии качества. Как и в предыдущем примере, начнем с вывода нижней границы на взаимную информацию, а потом убедимся, что граница достижима. Итак, по аналогии с двоичным источником

$$\begin{aligned} I(X;Y) &= h(X) - h(X|Y) = \\ &= h(X) - h(X-Y|Y) \geq \\ &\geq h(X) - h(X-Y) = \\ &= \log\sqrt{2\pi e\sigma^2} - \log\sqrt{2\pi eD} \\ &= \frac{1}{2}\log\frac{\sigma^2}{D}. \end{aligned} \tag{4.27}$$

Здесь использована формула дифференциальной энтропии гауссовского ансамбля из Таблицы 4.1 и то, что гауссовское распределение имеет наибольшую энтропию среди всех распределений с заданной дисперсией (см. свойство 4.1.7).

Чтобы доказать достижимость границы (4.27), введем с.в. $Z = X - Y$ и будем считать ее гауссовской с нулевым математическим ожиданием и дисперсией D. В качестве распределения для Y выберем гауссовское с нулевым математическим ожиданием и дисперсией $\sigma_y^2 = \sigma^2 - D$. Поскольку сумма независимых гауссовских с.в. есть гауссовская с.в. с суммарной дисперсией, дисперсия $X = Y + Z$ равна $\sigma_y^2 + D = \sigma^2$. Нетрудно видеть, что пара случайных величин X и Y удовлетворяет (4.27) с равенством. (Мы опять обошлись без указания в явном виде условного распределения $\varphi(y|x)$, но его легко найти с помощью формулы Байеса).

Приведенные рассуждения имеют смысл только при $D \le \sigma^2$. Поскольку значение $y = 0$ аппроксимирует значения с.в. X со средней ошибкой равной σ^2, то при $D > \sigma^2$ функция скорость-искажение тождественно равна нулю. Мы получили следующий результат.

Теорема 4.7 *Функция скорость-искажение источника независимых гауссовских с.в. с дисперсией σ^2 при квадратическом критерии качества равна*

$$H(D) = \begin{cases} \frac{1}{2}\log\frac{\sigma^2}{D}, & 0 \le D \le \sigma^2 \\ 0, & D > \sigma^2 \end{cases} \tag{4.28}$$

Формулу (4.28) иногда удобнее записывать в виде

$$H(D) = \max\left\{\frac{1}{2}\log\frac{\sigma^2}{D}, 0\right\}. \tag{4.29}$$

График функции скорость-искажение для гауссовского источника с единичной дисперсией показан на Рис. 4.5.

Попутно с доказательством Теоремы 4.7 нами установлена оценка снизу функции скорость-искажение произвольного стационарного источника, порождающего независимые сообщения (см. (4.27)). Эту оценку часто называют границей Шеннона.

Следствие 4.8 *(Граница Шеннона). Функция скорость-искажение источника независимых с.в. с дисперсией σ^2 и дифференциальной энтропией $h(X)$ при квадратическом критерии качества удовлетворяет неравенству*

$$H(D) \ge h(X) - \log\sqrt{2\pi e D}. \tag{4.30}$$

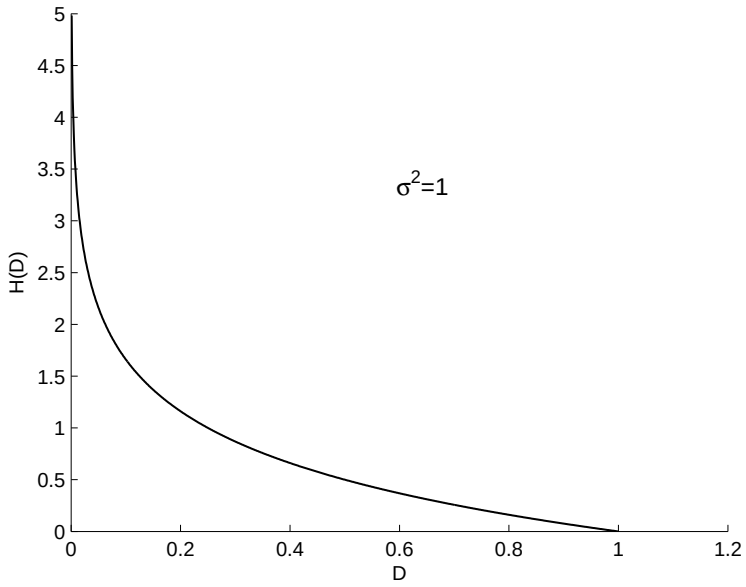


Рис. 4.5: Функция скорость-искажение для гауссовского источника.

4.7 Функция скорость-искажение для гауссовских последовательностей

В предыдущем параграфе по сути дела вычислялась функция скорость-искажение для случайных величин, а теперь мы должны сначала рассмотреть случайные векторы, а затем и случайные процессы. Здесь уместно вспомнить параграф 4.2. При вычислении дифференциальной энтропии произвольного вектора мы воспользовались тем, что: а) при линейном преобразовании гауссовского вектора изменение его дифференциальной энтропии зависит от определителя матрицы преобразования; б) из случайного вектора с заданной корреляционной матрицей с помощью линейного преобразования можно получить вектор с независимыми компонентами; в) гауссовский вектор имеет наибольшую дифференциальную энтропию среди векторов с заданной корреляционной матрицей. Эти знания будут использованы ниже при вычислении функции скорость-искажение.

В данном параграфе рассматривается только квадратическая мера качества, и в качестве первого шага подсчитаем функцию скорость-искажение для последовательности независимых (но не обязательно одинаково распределенных) гауссовских с.в. с совместной плотностью

$$f(\boldsymbol{x}) = \prod_{i=1}^{n} f_i(x_i),$$

где

$$f_i(x) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{x^2}{2\sigma_i^2}}.$$

Нам предстоит минимизировать взаимную информацию $I(X^n;Y^n)$ при условии, что среднеквадратическая ошибка на один символ не превышает заданной величины D. В точности как в (4.21), получаем

$$\begin{aligned} I(X^n;Y^n) &= h(X^n) - h(X^n|Y^n) = \\ &= \sum_{i=1}^n h(X_i) - \sum_{i=1}^n h(X_i|X_1...X_{i-1}Y_1...Y_n) \geq \\ &\geq \sum_{i=1}^n h(X_i) - \sum_{i=1}^n h(X_i|Y_i) = \\ &= \sum_{i=1}^n I(X_i;Y_i). \end{aligned} \tag{4.31}$$

Введем обозначение $D_i = \mathbf{M}[(x_i - y_i)^2]$ для ошибки в i-й компоненте вектора $\boldsymbol{x}$. Из (4.31) с учетом (4.29) следует

$$\begin{aligned} I(X^n;Y^n) &\geq \sum_{i=1}^n H(D_i) \geq \\ &\geq \sum_{i=1}^n \max\left\{\frac{1}{2}\log\frac{\sigma_i^2}{D_i}, 0\right\}. \end{aligned} \tag{4.32}$$

Понятно, что неравенства в (4.31) и (4.32) можно обратить в равенства соответствующим выбором условного распределения $\varphi(\boldsymbol{y}|\boldsymbol{x})$.

Нужно теперь отыскать минимум правой части (4.32) по всем неотрицательным $D_1, ..., D_n$ таким, что

$$\frac{1}{n}\sum_{i=1}^n D_i = D. \tag{4.33}$$

Заметим теперь, что введение дополнительного ограничения

$$D_i \leq \sigma_i^2, \quad i = 1, ..., n, \tag{4.34}$$

не приводит к потере минимума. Действительно, предположим, что минимум достигнут при таком наборе $D_1, ..., D_n$, что при некотором i имеем $D_i > \sigma_i^2$ и при некотором j имеет место $D_j < \sigma_j^2$ (такое j обязательно найдется, если правая часть (4.32) положительна). Очевидно, замена D_i на

$D_i' = \sigma_i^2$ и D_j на $D_j' = D_j + D_i - \sigma_i^2$ сохраняет ошибку (4.33) неизменной, уменьшая слагаемое с номером j в правой части (4.32). Следовательно, выбор $D_i > \sigma_i^2$ неоптимален.

Заметим теперь, что правая часть (4.32) выпукла $\cup$ как функция аргументов $D_1, ..., D_n$, поскольку она является суммой выпуклых $\cup$ функций (см. параграф 1.3). Поэтому любое выравнивание значений $D_1, ..., D_n$ (не нарушающее ограничений (4.34)) приводит к уменьшению суммы. Отсюда следует, что минимум суммы (4.32) будет достигаться в том случае, когда все D_i, удовлетворяющие (4.34) со строгим неравенством, одинаковы. При этом они должны быть равны некоторому θ такому, чтобы выполнялось (4.33). Окончательный результат сформулируем в виде следующей теоремы.

Теорема 4.9 *Для гауссовского вектора с независимыми компонентами, имеющими дисперсии $\sigma_i^2, i = 1, ..., n$, при квадратической мере искажения функция скорость-искажение вычисляется по формуле*

$$H(D) = \frac{1}{2n} \sum_{i=1}^{n} \max \left\{ \log \frac{\sigma_i^2}{\theta}, 0 \right\}, \tag{4.35}$$

где θ выбирается таким, что

$$\frac{1}{n} \sum_{i=1}^{n} \min \left\{ \theta, \sigma_i^2 \right\} = D. \tag{4.36}$$

Полученные соотношения заслуживают дополнительного обсуждения. Прежде всего, заметим, что как средняя ошибка для вектора, так и средняя ошибка для отдельных компонент будет не больше θ. Это означает, что при заданных требованиях к средней ошибке D все компоненты, дисперсия которых меньше D, игнорируются, и при кодировании нужно принимать во внимание только компоненты с большой дисперсией. Это правило подсчета затрат на кодирование получил название “обратного принципа заполнения водой”. Объяснение такого названия будет дано ниже после того, как полученные результаты будут обобщены на гауссовские случайные процессы.

Рассмотрим теперь гауссовский вектор $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ с произвольной корреляционной матрицей K_n. Напомним (см. параграф 4.3), что с помощью ортогонального преобразования (умножения на матрицу из собственных векторов) из $\boldsymbol{x}$ можно получить вектор с независимыми компонентами с дисперсиями, равными собственным числам матрицы K_n.

Напомним, что при взаимнооднозначном преобразовании ансамблей средняя взаимная информация не изменяется (см. свойство 4.1.11). Кроме того, ортонормированное преобразование векторов сохраняет расстояние между векторами. С помощью этих аргументов легко приходим к следующему утверждению.

Теорема 4.10 *Для n-мерного гауссовского вектора с корреляционной матрицей K_n, при квадратической мере искажения функция скорость-искажение вычисляется по формуле*

$$H(D) = \frac{1}{2n} \sum_{i=1}^{n} \max \left\{ \log \frac{\lambda_i}{\theta}, 0 \right\}, \tag{4.37}$$

где $\lambda_i, i = 1, ..., n$ – собственные числа K_n, а θ выбирается таким, что

$$\frac{1}{n} \sum_{i=1}^{n} \min \{\theta, \lambda_i\} = D. \tag{4.38}$$

Перейдем теперь к наиболее важному случаю, когда наблюдаемая на выходе источника последовательность $x_1, x_2, ...$ – стационарная гауссовская с корреляционной функцией $K(\tau)$. Напомним, что в параграфе 4.3 была введена спектральная плотность мощности

$$S(\omega) = \sum_{n=-\infty}^{\infty} K(n) e^{-in\omega}, \quad \omega \in (-\pi, \pi]$$

и сформулирована Теорема 4.5, связывающая дифференциальную энтропию процесса с его спектральной плотностью мощности.

Повторим, что если разбить область значений ω на достаточно малые интервалы, то интегралы от $S(\omega)$ по этим интервалам характеризуют энергию процесса в соответствующих спектральных диапазонах. При этом в силу свойств преобразования Фурье случайные значения сигналов в этих диапазонах асимптотически (с увеличением n) некоррелированы. Для гауссовских процессов некоррелированность эквивалентна независимости. Поэтому к сигналам в частотных поддиапазонах можно применить Теорему 4.9. Эти рассуждения могут служить эвристическим обоснованием следующей теоремы.

Теорема 4.11 *Для стационарного гауссовского процесса с ограниченной и интегрируемой спектральной плотностью мощности $S(\omega)$, при*

квадратической мере искажения функция скорость-искажение вычисляется по формуле

$$R(D) = \frac{1}{4\pi} \int_{-\pi}^{\pi} \max \left\{ \log \frac{S(\omega)}{\theta}, 0 \right\} d\omega. \tag{4.39}$$

где θ выбирается таким, что

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} \min \{\theta, S(\omega)\} \, d\omega = D. \tag{4.40}$$

Подробнее о доказательстве этого результата можно прочитать в [7], [3].

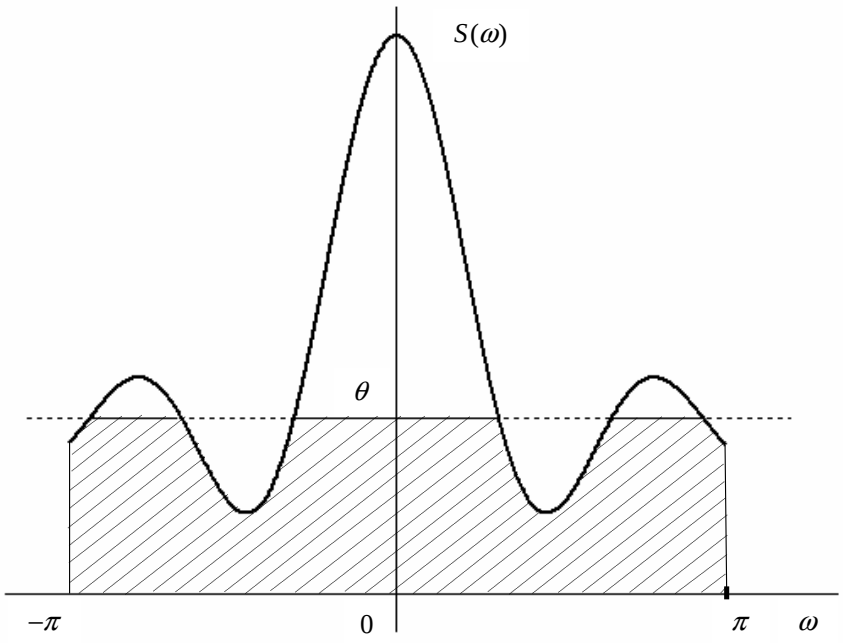


Рис. 4.6: Функция скорость-искажение для гауссовского источника.

Интерпретация вычислений по формуле (4.40) показана на Рис. 4.6. Представьте себе, что на гладкую поверхность воды опускается крышка неправильной формы. Благодарю принципу сообщающихся сосудов уровень воды поддерживается постоянным, в нашем случае этот уровень определяется значением параметра θ. Эту графическую интерпретацию называют “принципом заполнения водой”.

Согласно (4.39) при вычислении $R(D)$ интегрирование выполняется в тех интервалах частот ω, где спектральная плотность превышает уровень θ.

4.8 Обратная теорема кодирования для дискретного постоянного источника при заданном критерии качества

До сих пор в задачах кодирования с искажениями мы рассматривали в качестве источника информации непрерывные случайные величины или случайные последовательности. Напомним, что, с точки зрения применения методов теории информации, принципиальное различие между дискретными и непрерывными источниками состоит в том, что для непрерывных источников не определена собственная информация для отдельных сообщений источника и поэтому вместо обычной энтропии для них вычисляется дифференциальная энтропия.

В оставшейся части главы все доказательства приводятся для самого простого случая – дискретных постоянных источников. Мы рассматриваем дискретные источники, поскольку запись доказательств для них проще, чем для непрерывных. На самом деле, в доказательстве обратной теоремы кодирования для перехода к непрерывным источникам достаточно было бы поменять, там, где это надо, обычную энтропию на дифференциальную. Обобщение же прямой теоремы кодирования или алгоритма подсчета функции скорость-искажение на непрерывные источники потребовало бы значительно больших усилий.

Итак, имеется множество X^n последовательностей длины n, на выходе источника без памяти, выбираемых в соответствии с заданным на нем распределением вероятностей $p(\boldsymbol{x}) = p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i)$. Кодовая книга или код со скоростью R для этого источника определен как множество из $M = 2^{Rn}$ последовательностей из аппроксимирующего множества Y^n. При построении кода мы стремимся минимизировать величину средних искажений $D = \mathrm{M}[d_n(\boldsymbol{x}, \boldsymbol{y})]$, где мера искажения $d_n(\boldsymbol{x}, \boldsymbol{y})$ удовлетворяет ограничениям, сформулированным в начале главы, т.е. неотрицательна и $d_n(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n} \sum_{i=1}^{n} d(x_i, y_i)$.

Приведенная ниже обратная теорема кодирования утверждает, что для достижения требуемого уровня средних искажений не выше D скорость R кода источника должна быть не меньше значения функции скорость-искажение $H(D)$.

Теорема 4.12 *Пусть заданы дискретный алфавит постоянного источника $X = \{x\}$, аппроксимирующий алфавит $X = \{x\}$ и мера искажения $\{d(x, y), x \in X, y \in Y\}$. Для любого кода со скоростью R и средней ошибкой D имеет место неравенство $R \geq H(D)$*

Доказательство. Следующая цепочка преобразований доказывает теорему

$$
\begin{aligned}
nR &\overset{(a)}{\geq} H(Y^n) \geq \\
&\overset{(b)}{\geq} H(Y^n) - H(Y^n|X^n) = \\
&\overset{(c)}{=} I(X^n;Y^n) = \\
&\overset{(d)}{=} H(X^n) - H(X^n|Y^n) = \\
&\overset{(e)}{=} \left(\sum_{i=1}^{n} H(X_i)\right) - H(X^n|Y^n) \geq \\
&\overset{(f)}{\geq} \sum_{i=1}^{n} (H(X_i) - H(X_i|Y_i)) \geq \\
&\overset{(g)}{\geq} \sum_{i=1}^{n} I(X_i;Y_i) \geq \\
&\overset{(h)}{\geq} \sum_{i=1}^{n} H(D_i)) \geq \\
&\overset{(i)}{\geq} nH\left(\frac{1}{n}\sum_{i=1}^{n} D_i\right) = \\
&\overset{(j)}{=} H(D).
\end{aligned}
$$

Переход (a) справедлив, поскольку энтропия кодовой книги не превышает логарифма числа кодовых слов, (b) имеет место, поскольку вычитаемое неотрицательно. Далее, (c) и (d) – хорошо известные свойства взаимной информации, равенство (e) использует отсутствие зависимости букв источника. В (f) мы неравенство, опустив некоторые условия в вычитаемом, отчего вычитаемое могло только увеличиться. В (g) и (h) использованы определения взаимной информации и затем функции скорость-искажение как минимума взаимной информации при ограниченной ошибке, причем через D_i обозначено средняя ошибка аппроксимации X_i значениями из Y_i. Неравенство (i) основано на выпуклости функции скорость-искажение, равенство (j) – на определении средней меры искажения для последовательностей длины n. $\square$

4.9 Прямая теорема кодирования с заданным критерием качества

Прямая теорема кодирования устанавливает существование кодов источника, для которых скорость R при заданном допустимом среднем искажении D может быть сделана сколь угодно близкой к соответствующему значению функции скорость-искажение $H(D)$. Один из способов доказательства этого утверждения могло быть указание конкретного способа кодирования и декодирования, как это было сделано в случае кодирования без потерь. Такое доказательство мы назвали бы конструктивным. К сожалению такое конструктивное решение задачи возможно только для некоторых вырожденных ситуаций (например, при $D = 0$). Чтобы доказать существование хороших кодов для произвольных источников без памяти, придется воспользоваться уже знакомой нам техникой случайного кодирования.

Итак, метод доказательства состоит в том, что среднее искажение оценивается для кода, слова которого получены случайным независимым выбором кодовых слов. При подсчете ошибки кодирования усреднение выполняется как по множеству последовательностей источника, так и по множеству кодов. Если среднее значение искажений окажется равным D, то мы сможем утверждать что найдется хотя бы один код, для которого среднее искажение не больше D.

Теорема 4.13 *Для дискретного постоянного источника* $X = \{x, p(x)\}$, *аппроксимирующего множества* $Y = \{y\}$ *и заданной ограниченной меры искажений* $d(x, y)$, *для любых положительных* ε, δ *найдется достаточно большое число* n_0, *такое, что при* $n \geq n_0$ *при любом заданном* D *найдется код со скоростью* $R \leq H(D) + \delta$, *и среднем искажении не больше* $D + \varepsilon$, *где*

$$H(D) = \min_{\{p(y|x)\}:\mathrm{M}[d(x,y)]\leq D} I(X;Y) \quad -$$

функция скорость-искажение дискретного постоянного источника.

Иными словами, при любом D и любой скорости кода большей функции скорость-искажение $H(D)$ возможно кодирование со средними искажениями сколь угодно близкими к D.

Доказательство. Доказательство, аналогично теореме кодирования для канала с шумом, разбивается на три шага: построение ансамбля случайных кодов, выбор процедуры кодирования, оценка величины средних

(по ансамблю кодов и по последовательностям источника) искажений. Полностью доказательство приведено в [12].

Сложность применения теоретических оценок к реальным задачам связана с тем, что мы умеем вычислять значения $H(D)$ для весьма ограниченного набора моделей, и эти модели связаны в основном со стационарными гауссовскими источниками.

Существенным подспорьем в практическом применении теории кодирования для источников с заданной мерой точности является численный алгоритм Блэйхута, позволяющий построить функцию $H(D)$ по дискретной модели или по ее оценке, полученной по реализации последовательности данных на выходе источника. Описание алгоритма приведено в [12].

Литература

[1] Бабкин В. Ф. Метод универсального кодирования независимых сообщений неэкспоненциальной трудоемкости. *Проблемы передачи информации*, 7(4):13–21, 1971.

[2] Блейхут Р. *Быстрые алгоритмы цифровой обработки сигналов.* М.: Мир, 1989.

[3] Галлагер Р. *Теория информации и надежная связь.* М.: Советское радио, 1974.

[4] Гоппа В. Д. *Введение в алгебраическую теорию информации.* М.: Наука, Физматлит, 1995.

[5] Ватолин Д., Ратушняк А., Смирнов М., Юкин В. *Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео.* М.: Диалог-МИФИ, 2002.

[6] Двайт Г. Б. *Таблицы интегралов и другие математические формулы.* М.: Наука, М., 1977.

[7] Колесник В. Д., Полтырев Г. Ш. *Курс теории информации.* М.: Наука, 1982.

[8] Колмогоров А.Н. Три подхода к определению понятия "количество информации". *Проблемы передачи информации*, 1:3–11, 1965.

[9] Конвей Дж., Слоэн Н. *Упаковки шаров, решетки и группы, Т. 1,2.* М.: Мир, 1990.

[10] Кошелев В. Н. Квантование с минимальной энтропией. *Проблемы передачи информации*, 14:151–156, 1963.

[11] Кричевский Р. Е. *Сжатие и поиск информации.* М.: Радио и связь, 1989.

[12] Кудряшов Б. Д. *Теория информации.* С.-Пб.: Питер, 2009.

[13] Кудряшов Б.Д., Юрков К.В. Границы случайного кодирования для второго момента многомерных числовых решеток. *Проблемы передачи информации*, 43(1):53–64, 2007.

[14] Левенштейн В. И. Об избыточности и замедлении разделимого кодирования натуральных чисел. *Проблемы кибернетики*, 14:173–179, 1965.

[15] Дж. К. Омура, Витерби А. Д. *Принципы цифровой связи и кодирования.* М.: Радио и связь, 1982.

[16] Рябко Б.Я. Сжатие информации с помощью стопки книг. *Проблемы передачи информации*, 16(4):16–21, 1980.

[17] Рябко Б.Я. Быстрая нумерация комбинаторных объектов. *Дискретная математика*, 10(2):101–119, 1998.

[18] Фано Р. *Передача информации. Статистическая теория связи.* М.: Мир, 1965.

[19] Фитингоф Б. М. Оптимальное кодирование при меняющейся и неизвестной статистике сообщений. *Проблемы передачи информации*, 2(2):3–11, 1967.

[20] Чисар И., Кернер Я. *Теория информации. Теоремы кодирования для дискретных систем без памяти.* М.: Мир, 1985.

[21] Шеннон К. Математическая теория связи. In *Работы по теории информации и кибернетике*, pages 243–332. М.: ИЛ, 1963.

[22] Шеннон К. Теоремы кодирования для дискретного источника при заданном критерии качества. In *Работы по теории информации и кибернетике*, pages 587–621. ИЛ, 1963.

[23] Штарьков Ю. М. Универсальное последовательное кодирование отдельных со- общений. *Проблемы передачи информации*, 23(3):3–17, 1987.

[24] Штарьков Ю. М., Чокенс Ч. Дж., Виллемс Ф. М. Дж. Оптимальное универсальное кодирование по критерию максимальной индивидуальной относительной избыточности. *Проблемы передачи информации*, 33(1):21–34, 1997.

[25] Штарьков Ю. М., Чокенс Ч. Дж., Виллемс Ф. М. Дж. Мультиалфавитное взвешенное универсальное кодирование древовидно-контекстных источников. *Проблемы передачи информации*, 40(1):98–110, 2004.

[26] Abramson N. *Information theory and coding.* McGraw-Hill, New York, 1963.

[27] Bentley J. L., Sleator D., Tarjan R. E., Wei V. K. A locally adaptive data compression scheme. In *Proceedings of the 22nd Alterton Conference on Communication, Control and Computing*, pages 233–242, Oct. 1984.

[28] Berger T. *Rate distortion theory: A mathematical basis for data compression.* Prentice-Hall, Englewood Cliffs, NJ, 1971.

[29] Berger T. Optimum quantizers and permutation codes. *IEEE Transactions on Information Theory*, 18(6):759–765, Nov. 1972.

[30] Berger T., Gibson J. D. Lossy source coding. *IEEE Transactions on Information Theory*, IT-44(6):2693–2723, October 1998.

[31] Berrow C., Glavieux A. Near-optimum error-correcting coding and decoding: Turbo codes. *IEEE Transactions on Communications*, COM-44(10):1261–1271, October 1996.

[32] Blahut R. E. Computation of channel capacity and rate-distortion functions. *IEEE Transactions on Information Theory*, 18(4):460–473, Jul 1972.

[33] Bocharova I. *Compression for multimedia.* Lund University, 2007.

[34] Burrows M., Wheeler D. J. A block-sorting lossless data compression algorithm. Report 124, Digital Systems Research Center Research, 1994.

[35] Choi B. S., Cover T. M. An information-theoretic proof of burgs maximum entropy spectrum. *Proceedings IEEE*, 72:1094–1095, 1984.

[36] Cleary J. G., Witten I. H. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Comminication*, 32(4):396–402, April 1984.

[37] Costello D. J., Forney G. D., Jr. Channel coding: The road to channel capacity. *Proceedings of th IEEE*, 95(6):1150–1177, June 2007.

[38] Costello D. J., Hagenauer J. Applications of error-control coding. *IEEE Transactions on Information Theory*, IT-44(6):2531–2560, Oct 1998.

[39] Cover T. M., Thomas J. A. *Elements of Information Theory.* John Wiley & Sons, New York, 1991.

[40] Elias P. Interval and recency rank source encoding: two on-line adaptive variable-rate schemes. *IEEE Transactions on Information Theory*, IT-33(1):3–10, January 1987.

[41] Farvardin N., Modestino J. W. Optimum quantizer performance for a class of non-gaussian memoryless sources. *IEEE Transactions on Information Theory*, 30(3):485–497, May 1984.

[42] Fiala E. R., Greene D. H. Data compression with finite windows. *Communication of the ACM*, 32, 1989.

[43] Forney G. D., Jr. Convolutional codes, II: Maximum likelihood decoding. *Information and control*, 25(4):222–266, 1974.

[44] Forney G. D., Jr., Ungerboeck G. Modulation and coding for linear gaussian channels. *IEEE Transactions on Information Theory*, IT-44(6):2384–2415, Oct 1998.

[45] Gallager R. G. Variation on a theme by Huffman. *IEEE Transactions on Information Theory*, IT-24(6):668–674, November 1978.

[46] Gallager R. G., VanVoorhis D. C. Optimal source codes for geometrically distributed integer alphabets. *IEEE Transactions on Information Theory*, IT-21(2):228–230, March 1975.

[47] Gish H., Pierce J.N. Asymptotically efficient quantizing. *IEEE Transactions on Information Theory*, 14(5):676–683, Sept. 1968.

[48] Golomb S. W. Run-length encodings. *IEEE Transactions on Information Theory*, IT-12(3):399–401, May 1966.

[49] Gray R. M., Neuhoff D. L. Quantization. *IEEE Transactions on Information Theory*, IT-44(6):2325–2383, Oct 1998.

[50] Howard P. G. The design and analysis of efficient lossless data compression systems. Report CS-93-28, Dept. of Comp. Sci. Brown University, Providence, Rhode Island, 1993.

[51] Johannessson R., Zigangirov. K. Sh. *Fundamentals of Convolutional Coding*. IEEE Press, Piscataway, NJ, 1999.

[52] Kolmogorov A. N. On the shannon theory of information transmission in the case of continious signals. *IRE Transactions on Information Theory*, IT-2:pp. 102–108, 1956.

[53] Linde Y., Buzo A., Gray R. M. An algorithm for vector quantizer design. *IEEE Transactions on Communications Trechnology*, COM-28(1):84–95, Jan 1980.

[54] Lloyd S. P. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, Mar 1982.

[55] Manstetten D. Tight bounds on the redundancy of Huffman codes. *IEEE Transactions on Information Theory*, 38(1):144–151, Jan. 1992.

[56] Max J. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, Mar. 1960.

[57] Welch T. A. A technique for high-performance data compression. *IEEE Computer*, 17(6):8–19, 1984.

[58] Witten I. C., Neal R. M., Cleary J. G. Arithmetic coding for data compression. *Communication of the ACM*, 30(6):520–540, 1987.

[59] Zador P. Asymptotic quantization error of continuous signals and their quantization dimension. *IEEE Transactions on Information Theory*, 28(2):139–149, March 1982.

[60] Ziv J., Lempel A. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, IT-23(3):337–343, May 1977.

[61] Ziv J., Lempel A. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, IT-24(5):530–536, Nov. 1978.

Предметный указатель