

**1. Формула условной вероятности события.**

Для произвольной пары событий  $A, B \subseteq X$  условной вероятностью события  $A$  при условии  $B$  называется величина:

$$P(A|B) = \frac{P(AB)}{P(B)};$$

при  $P(B) \neq 0$  и величина  $P(A|B) = 0$  при  $P(B) = 0$ .

**2. Определение несовместных событий. Определение совместно независимых событий.**

События несовместны, если их пересечение - пустое множество. Для несовместных событий вероятность объединения равна сумме вероятностей событий. Если условие несовместности не выполнено, вероятность объединения становится меньше суммы вероятностей.

События  $A, B \subseteq X$  называют независимыми, если:

$$P(AB) = P(A)P(B);$$

Совместно независимыми называют события  $A_1, \dots, A_n \subseteq X$  такие, что:

$$P(A_1 \dots A_n) = P(A_1)P(A_2) \dots P(A_n);$$

**3. Формула вероятности совместных события (произведения событий).**

$$P(AB) = P(A|B)P(B);$$

**4. Формула полной вероятности (апостериорной вероятности).**

Формула полной вероятности и формула апостериорной вероятности, или формула Байеса:

$$P(H_j|A) = \frac{P(A|H_j)P(H_j)}{\sum_{m=1}^M P(A|H_m)P(H_m)};$$

**5. Произведение множеств.**

Произведением множеств  $X$  и  $Y$  называется множество:

$$Z = XY = \{(x, y) : x \in X, y \in Y\}$$

множество упорядоченных пар, первый элемент которых принадлежит множеству  $X$ , а второй - множеству  $Y$ .

**6. Формула условного распределения вероятностей.**

Условным распределением вероятностей на элементах множества  $X$  при фиксированном элементе  $y \in Y$  называется распределение:

$$p(x|y) = \begin{cases} \frac{p(x,y)}{p(y)} & \text{если } p(y) \neq 0, \\ 0 & \text{в противном случае,} \end{cases}$$

При этом  $x \in X$ .

**7. Условие независимости двух ансамблей.**

Ансамбли и независимы, если имеют место тождества:

$$x \in X, y \in Y$$
$$p(x, y) = p(x)p(y)$$

## 8. Матожидание, дисперсия и корреляционный момент случайных величин.

Математическим ожиданием случайной величины  $x \in X$  называется:

$$\mathbf{M}_X[x] = \sum_{x \in X} xp(x)$$

Другими важными характеристиками случайных величин являются дисперсия:

$$\mathbf{D}_X[x] = \mathbf{M}_X[(x - \mathbf{M}_X[x])^2]$$

и корреляционный момент:

$$K_{XY}(x, y) = \mathbf{M}_{XY}[(x - \mathbf{M}[x])(y - \mathbf{M}[y])]$$

Если для двух случайных величин корреляционный момент равен нулю, случайные величины называют некоррелированными.

## 9. Собственная информация сообщения и ее свойства.

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

$$I(x) = -\log p(x)$$

единица измерения - бит.

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  $x_1, \dots, x_n$  имеет место равенство:

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

## 10. Энтропия дискретного ансамбля.

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

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

## 11. Свойство энтропии дискретного ансамбля (свойства 1.3.1-1.3.7).

- Свойство 1.3.1:  $H(X) \geq 0$ ;
- Свойство 1.3.2:  $H(X) \leq \log X$ . Равенство имеет место в том и только в том случае, когда элементы ансамбля  $X$  равновероятны;

- Свойство 1.3.3: Если для двух ансамблей  $X$  и  $Y$  распределения вероятностей представляют собой одинаковые наборы чисел (различающиеся только порядком следования элементов), то  $H(X) = H(Y)$ ;
- Свойство 1.3.4: Если ансамбли  $X$  и  $Y$  независимы, то  $H(XY) = H(X) + H(Y)$ ;
- Свойство 1.3.5: Энтропия — выпуклая  $\cap$  функция распределения вероятностей на элементах ансамбля ;
- Свойство 1.3.6: Пусть  $X = \{x, p(x)\}$  и  $A \subseteq X$ . Введем новый 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 1.3.7: Пусть задан ансамбль  $X$  и на множестве его элементов определена функция  $g(x)$ . Введем ансамбль  $Y = \{y = g(x)\}$ . Тогда  $H(X) \leq H(Y)$ . Равенство имеет место тогда и только тогда, когда функция  $g(x)$  обратима.

## 12. Энтропия двоичного ансамбля.

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

$$H(X) = -p \log p - q \log q = \eta(p)$$

Мы ввели специальное обозначение  $\eta(p)$  для энтропии двоичного ансамбля, в котором один из элементов имеет вероятность 2. 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  $p \rightarrow 0$  и  $p \rightarrow 1$ . Получим  $\eta(0) = \eta(1) = 0$ . Далее, заметим, что  $\eta(p) = \eta(1 - p)$ , то есть функция симметрична относительно оси  $p = \frac{1}{2}$ .

## 13. Условная собственная информация.

Рассмотрим ансамбли  $X = \{x\}$  и  $Y = \{y\}$  и их произведение  $XY = \{(x, y), (x, y)\}$ .

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  $y$ .

## 14. Формула условной энтропии при фиксированном событии. Формула условной энтропии ансамбля при фиксированном другом ансамбле.

Энтропия ансамбля  $X$  — средняя информация сообщений  $x \in X$ . Усреднив условную информацию  $I(x|y)$  по  $x \in X$ , получим величину:

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

называемую условной энтропией  $X$  при фиксированном  $y \in Y$ . Заметим, что в определении имеется неопределенность при  $p(x|y) = 0$ .

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 формуле условной энтропии 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  $Y$ .

## 15. Свойства условной энтропии (свойства 1.4.1-1.4.6).

- 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:  $H(X_1 \dots X_n) = H(X_1) + H(X_2|X_1) + H(X_3|X_2X_1) + \dots + H(X_n|X_1, \dots, X_{n-1})$ ;
- 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:  $H(X_1 \dots X_n) \leq \sum_{i=1}^n H(X_i)$ , причем равенство возможно только в случае совместной независимости ансамблей  $X_1, \dots, X_n$ .

## 16. Определение стационарного процесса.

Вместо отдельных ансамблей и произведений конечного числа ансамблей мы будем рассматривать теперь — случайные последовательности из произвольного числа событий. Если элементы случайной последовательности — вещественные числа, то такие последовательности называются случайными процессами.

Процесс называется стационарным, если для любых  $n$  и  $y$  имеет место равенство

$$p(x_1, \dots, x_n) = p(x_{1+t}, \dots, x_{n+t})$$

в котором подразумевается, что  $x_i = x_{i+t}, i = 1, \dots, n$ .

Случайный процесс стационарен, если вероятность любой последовательности не изменяется при ее сдвиге во времени (не зависит от положения последовательности на оси времени).

## 17. Цепь Маркова связности s. Однородная цепь Маркова. Простая цепь Маркова. Матрица переходных вероятностей цепи Маркова.

Случайный процесс  $x_1, x_2, \dots$  называют цепью Маркова связности s, если для любых n и для любых  $\mathbf{x} = (x_1, \dots, x_n) \in X^n$  справедливы соотношения

$$p(x) = p(x_1, \dots, x_s) p(x_{s+1}|x_1, \dots, x_s) p(x_{s+2}|x_2, \dots, x_{s+1}) \times \dots \times p(x_n|x_{n-s}, \dots, x_{n-1})$$

Марковский процесс связности S называется такой процесс, для которого при  $n > s$

$$p(x_n|x_1, \dots, x_{n-1}) = p(x_n|x_{n-s}, \dots, x_{n-1})$$

то есть условная вероятность текущего значения при известных предшествующих s не зависит от всех других предшествующих значений.

Описание марковского процесса задается начальным распределением вероятностей на последовательностях из первых S значений (состояний) и условными вероятностями вида  $p(x_n|x_{n-s}, \dots, x_{n-1})$  для всевозможных последовательностей  $(x_{n-s}, \dots, x_n)$ . Если указанные условные вероятности не изменяются при сдвиге последовательностей  $(x_{n-s}, \dots, x_n)$  во времени, марковская цепь называется однородной.

Однородная марковская цепь связности  $s = 1$  называется простой цепью Маркова. Для описания простой цепи Маркова с множеством состояний  $X = \{0, 1, \dots, M-1\}$  достаточно указать начальное распределение вероятностей  $\{p(x_1), x_1 \in X\}$  и условные вероятности

$$\pi_{ij} = P(x_t = j | x_{t-1} = i), i, j = 0, \dots, M-1$$

называемые переходными вероятностями цепи Маркова.

Переходные вероятности удобно записывать в виде квадратной матрицы размерности  $M \times M$ :

$$\Pi = \begin{pmatrix} \pi_{0,0} & \pi_{0,1} & \cdots & \pi_{0,M-1} \\ \pi_{1,0} & \pi_{1,1} & \cdots & \pi_{1,M-1} \\ \vdots & \vdots & \ddots & \vdots \\ \pi_{M-1,0} & \pi_{M-1,1} & \cdots & \pi_{M-1,M-1} \end{pmatrix}$$

называемой матрицей переходных вероятностей.

## 18. Стационарное распределение для цепи Маркова. Эргодические цепи. Понятие состояния цепи. Неприводимые цепи.

Пусть существует стохастический вектор  $\mathbf{p}$

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

Стохастический вектор  $\mathbf{p}$ , удовлетворяющий уравнению выше, называется стационарным распределением для цепи Маркова, задаваемой матрицей переходных вероятностей  $\Pi$ .

Финальным распределением вероятностей называют вектор

$$p_\infty = \lim_{t \rightarrow \infty} p_t = \lim_{t \rightarrow \infty} p_1 \Pi^t$$

(если указанный предел существует).

Для широкого класса простых цепей Маркова предел финального распределения не зависит от начального распределения  $p_1$  и равен единственному решению уравнения  $\mathbf{p} = \mathbf{p}\Pi$ , то есть  $p_\infty = p$ . Такие цепи называют эргодическими.

Множество состояний  $S$  называется замкнутым, если никакое состояние вне  $S$  не может быть достигнуто из состояния, входящего в  $S$ . Цепь называется неприводимой, если в ней содержится не больше одного замкнутого множества. Цепь Маркова неприводима, в частности, тогда, когда все ее состояния достижимы друг из друга.

## 19. Энтропия на букву последовательности.

Для стационарного процесса энтропия распределения вероятностей на таких блоках  $H(X_1 \dots X_n) = H(X^n)$  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  $n$ .

## 20. Свойства двух информационных мер (теорема 1.5: положения A-D).

**ТЕОРЕМА 1.5.** 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.  $\lim_{n \rightarrow \infty} H_n(X) = \lim_{n \rightarrow \infty} H(X|X^n)$ ;

## 21. Энтропия на сообщение дискретного постоянного источника.

В качестве дискретного постоянного источника рассматривается дискретный стационарный источник без памяти. По свойствам энтропии для источника без памяти имеем:

$$\begin{aligned} H(X_1 \dots X_n) &= H(X_1) + \dots + H(X_n) \\ H(X^n) &= nH(X) \end{aligned}$$

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

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

И, следовательно,

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

Энтропия на сообщение дискретного постоянного источника в точности равна его одномерной энтропии.

## 22. Энтропия на сообщение Марковского источника.

Энтропия на сообщение марковского источника. Для стационарного источника, описываемого моделью цепи Маркова связности  $S$ , можем записать:

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

Правая часть не зависит от  $n$ , значит,

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

Используя стационарность и свойства условной энтропии, запишем:

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

Из

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

учитывая марковость и стационарность, получаем:

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

Теперь можно поделить обе части на  $n$  и устремить  $n$  к бесконечности:

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

Перепишем выражение с  $(n - s)H(X|X^s)$  в виде:

$$H_n(X) = H(X|X^s) + \frac{s}{n}(H_s(X) - H(X|X^s)) = H(X|X^n) + \frac{s}{n}(H_s(X) - H(X|X^s))$$

Отсюда видна разница между  $H_n(X)$  и  $H(X|X^n)$ .

### 23. Определение равномерного кода. Мощность кода. Скоростью равномерного кода.

Равномерным кодом длины  $n$  называется любое подмножество  $C$  множества  $A^n$  то есть любое подмножество множества последовательностей длины  $n$ . Элементы кода  $C$  называют кодовыми словами. Мощность кода  $|C|$  - это количество кодовых слов в коде  $C$ .

Скоростью равномерного кода называется величина:

$$R = \frac{[\log |C|]}{N}$$

(бит / букву источника), где  $[a]$  обозначает округление числа  $a$  вверх до ближайшего целого.

### 24. Условие взаимно однозначного кодирования.

Работа кодера (алгоритм кодирования) описывается отображением множества  $X^N$  на множество слов кода  $C$ . Декодирование задается отображением  $C$  на  $X^N$ .

Если оба отображения взаимно однозначные, то на выходе декодера можно будет получить точную копию передаваемой последовательности.

Взаимно однозначное кодирование возможно только тогда, когда:

$$|X|^N \leq |C|$$

или

$$R \geq \log |X| \geq H(X)$$

Если буквы источника не равновероятны ( $H(X) < \log |X|$ ), то скорость кода окажется заведомо больше энтропии источника.

### 25. Неравенство Чебышева для суммы независимых случайных величин.

Рассмотрим дискретную случайную величину  $X = \{x, p(x)\}$ . Предположим, что все ее значения  $x \in X$  неотрицательны, и при этом предположении оценим вероятность события  $P(x \geq a)$  для некоторого числа  $A > 0$ .

$$P(x \geq a) = \sum_{x \geq A} p(x) \leq \sum_{x \geq A} \frac{x}{A} p(x) \leq \frac{1}{A} \sum_{x \geq A} xp(x) = \frac{M[x]}{A}$$

Введем обозначение  $m_x = \mathbf{M}[x]$ . Тогда

$$P(x \geq a) \leq \frac{m_x}{A}$$

Для оценки искомой вероятности достаточно знать только ее математическое ожидание.

Пусть  $X = \{x, p(x)\}$  - произвольная (необязательно неотрицательная) случайная величина. Для произвольного  $\epsilon > 0$  оценим вероятность  $P|x - m_x| \geq \epsilon$ .

Положим  $y = |x - m_x|$ . Для этой неотрицательной случайной величины получаем

$$P(y \geq \epsilon) = P(y^2 \geq \epsilon^2) \leq \frac{\mathbf{M}[y^2]}{\epsilon^2} = \frac{\mathbf{M}[(x - m_x)^2]}{\epsilon^2} = \frac{\mathbf{D}[x]}{\epsilon^2}$$

Результат запишем в виде:

$$P(|x - m_x| \geq \epsilon) \leq \frac{\sigma_x^2}{\epsilon^2}$$

где  $\sigma_x^2 = \mathbf{D}[x]$ . Данное неравенство называется неравенством Чебышева.

Рассмотрим теперь последовательность из  $n$  независимых случайных величин  $X_i, i = 1, \dots, n$ , имеющих одинаковое математическое ожидание  $m_x$  и одинаковую дисперсию  $\sigma_x^2$ . Найдем

$$P\left(\left|\frac{1}{n} \sum_{i=1}^n x_i - m_x\right| \geq \epsilon\right)$$

Положим

$$y = \frac{1}{n} \sum_{i=1}^n x_i$$

Учитывая независимость случайных величин  $x_1, \dots, x_n$ , получим

$$\mathbf{M}[y] = m_x$$

$$\mathbf{D}[y] = \frac{1}{n} \sigma_x^2$$

Применив к случайным величинам  $y$  неравенство Чебышева

$$P\left(\left|\frac{1}{n} \sum_{i=1}^n x_i - m_x\right| \geq \epsilon\right) \leq \frac{\sigma_x^2}{n\epsilon^2}$$

неравенство Чебышева для суммы независимых случайных величин.

## 26. Прямая теорема кодирования (теорема 1.6 с доказательством).

**ТЕОРЕМА 1.6.** Пусть  $H$  - энтропия дискретного постоянного источника. Для любых  $\epsilon, \delta > 0$  существует  $n_0$  такое, что для любого  $n > n_0$  найдется равномерный код, кодирующий источник блоками длиной  $n$  со скоростью  $R \leq H + \delta$  и вероятностью ошибки  $P_e \leq \epsilon$ .

**Доказательство.** Правило кодирования полностью описывается выбором множества  $T \subseteq X^n$  однозначно кодируемых последовательностей сообщений источника.

Выберем  $T \subseteq X^n$  следующим образом:

$$T = \left\{x : \left|\frac{1}{n} I(x) - H\right| \leq \delta_0\right\} \quad (1.20)$$

где через  $I(x) = -\log p(x)$  обозначена собственная информация последовательности  $\mathbf{x} \in X^n$ , а через  $\delta_0$  обозначена положительная константа, значение которой будет выбрано позже.

Множество  $T$  состоит из таких последовательностей, средняя собственная информация которых весьма близка к энтропии, если величина  $\delta_0$  достаточно мала.

Множество  $T$  называется множеством типичных последовательностей. Это множество играет важную роль в теории информации.

Из (1.20) следует, что вероятности последовательностей множества  $T$  удовлетворяют неравенствам

$$2^{-n(H+\delta_0)} \leq p(x) \leq 2^{-n(H-\delta_0)}$$

Левая и правая части могут рассматриваться как оценки снизу и сверху для вероятностей последовательностей из  $T$ . Для оценки числа элементов в этом множестве заметим, что

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

Следовательно

$$|T| \leq 2^{n(H+\delta_0)}$$

По определению скорость кода:

$$R = \frac{\lceil \log |T| \rceil}{n} \leq H + \delta_0 + \frac{1}{n} \quad (1.23)$$

Ошибка кодирования имеет место тогда, когда источник порождает последовательность, не входящую в множество  $T$ . Поэтому вероятность ошибки  $P_e$  удовлетворяет соотношениям:

$$P_e = P(\mathbf{x} \notin T) = P\left(\left|\frac{1}{n}I(x) - H\right| > \delta_0\right) = P\left(\left|\frac{1}{n}\sum_{i=1}^n I(x_i) - H\right| > \delta_0\right) \quad (1.24)$$

Последний переход учитывает, что буквы источника независимы и, в силу аддитивности собственной информации, информация последовательности равна сумме информации отдельных букв

$$\mathbf{M}[I(x)] = H$$

К выражению (1.24) можно применить неравенство Чебышева:

$$P_e \leq \frac{\mathbf{D}[I(x)]}{n\delta_0^2} \quad (1.25)$$

Значит, для любой наперед заданной величины  $\delta_0$  можно теперь подобрать длину кодируемых блоков  $n$  так, чтобы вероятность ошибки была меньше любого  $\epsilon$ .

Положим теперь  $\delta_0 = \frac{\delta}{2}$ . При  $n \geq n_{01} = \frac{\mathbf{D}[I(x)]}{\delta^2 \epsilon}$  из (1.25) имеем  $P_e \leq \epsilon$ .

Из (1.23) следует, что при  $n \geq n_{02} = \frac{2}{\delta}$  выполняется условие  $R < H + \delta$ . Следовательно, при  $n \leq n_0 = \max(n_{01}, n_{02})$  скорость кода  $R$  и вероятность ошибки  $P_e$  удовлетворяют требованиям теоремы.

## 27. Обратная теорема кодирования (теорема 1.7 с доказательством).

**ТЕОРЕМА 1.7.** Для дискретного постоянного источника с энтропией  $H$  существует  $\epsilon > 0$  такое, что для любого  $\delta > 0$  для любого равномерного кода со скоростью  $R \leq H - \delta$  вероятность ошибки удовлетворяет неравенству  $P_e \geq \epsilon$ .

**Доказательство** [от обратного] Если представить, что сколь угодно точное кодирование возможно при скорости меньшей, чем энтропия источника. То необходимо для заданного источника указать некоторую строго положительную независимую от длины кодируемых последовательностей величину  $\epsilon$ , которая была бы оценкой снизу вероятности ошибки любого кода со скоростью  $R \leq H - \delta$ .

Код полностью описывается выбором множества  $T_1$  однозначно кодируемых последовательностей. По определению, скорость кода равна  $R = \lceil \log |T_1| \rceil / n$ , следовательно, код содержит

$$|T_1| \leq 2^{nR} \leq 2^{n(H-\delta)}$$

кодовых слов. Вместо вычисления нижней оценки для вероятности ошибки  $P_e$  построим оценку сверху для вероятности правильного кодирования

$$P_c = 1 - P_e = \sum_{x \in T_1} p(x) \quad (1.27)$$

Введем вспомогательное множество:

$$T = \left\{x : \left|\frac{1}{n}I(x) - H\right| \leq \delta_0\right\}$$

где через  $I(x) = -\log p(x)$  обозначена собственная информация последовательности  $\mathbf{x} \in X^n$ , а через  $\delta_0$  обозначена положительная константа.

Разобьем сумму в правой части (1.27) на две части:

$$P_c = \sum_{x \in T_1 \cap T} p(x) + \sum_{x \in T_1 \cap T^c} p(x)$$



где через  $T^c$  обозначено дополнение к множеству  $T$ .

Оценим вторую сумму следующим образом:

$$\sum_{x \in T_1 \cap T^c} p(x) \leq \sum_{x \in T^c} p(x) = P(T^c) = P(x \notin T)$$

С помощью неравенства Чебышева получаем оценку:

$$\sum_{x \in T_1 \cap T^c} p(x) \leq \frac{\mathbf{D}[I(x)]}{n\delta_0^2}$$

Для первой из двух сумм:

$$\sum_{x \in T_1 \cap T} p(x) \leq |T_1 \cap T| \max_{x \in T_1 \cap T} p(x) \leq |T_1| \max_{x \in T_1 \cap T} p(x) \leq |T_1| \max_{x \in T} p(x) \quad (1.31)$$

Здесь мы сначала оценили сумму сверху, умножив число слагаемых на максимальное слагаемое. Затем использовали неравенство  $|T_1 \cap T| \leq |T_1|$ . Последний переход основан на том, что при расширении области поиска максимума максимальное значение не уменьшается.

Подставим в (1.31) оценку мощности кода и оценку максимальной вероятности последовательностей из  $T$ :

$$\sum_{x \in T_1 \cap T} p(x) \leq 2^{n(H-\delta)} 2^{-n(H-\delta_0)} = 2^{-n(\delta-\delta_0)}$$

Подстановка приводит к следующей оценке сверху для вероятности правильного кодирования:

$$P_c \leq 2^{-n(\delta-\delta_0)} + \frac{\mathbf{D}[I(x)]}{n\delta_0^2} \quad (1.33)$$

Пусть  $\delta$  - произвольное положительное число. Выберем теперь значение параметра  $\delta_0$  равным  $\delta_0 = \delta/2$ . При таком выборе правая часть неравенства (1.33) убывает с увеличением  $n$ .

Поскольку мы убедились, что с увеличением длины кодируемых блоков вероятность ошибки не только не убывает, но и стремится к 1, то для того чтобы окончательно доказать теорему, нужно установить, что и при конечных длинах блоков вероятность ошибки остается большой.

Вероятность ошибки положительна при любых длинах блоков  $n$ . Поскольку мы договорились о том, что все буквы источника имеют положительные вероятности, для достижения нулевой вероятности ошибки нужно назначить кодовое слово каждой последовательности сообщений, то есть объем кода должен удовлетворять неравенству

$$|T_1| \geq |X|^n$$

Для этого скорость кода должна быть

$$R \geq \log |T_1|/n \geq \log |X| \quad (1.34)$$

Из свойства 1.3.2 следует, что  $H(X) \leq \log |X|$  и, значит, (1.34) входит в противоречие с условиями теоремы, поскольку скорость должна быть меньше энтропии.

Вероятность ошибки положительна при любых  $n$ .

Обозначим через  $n_0$  такое значение длины последовательностей  $n$ , что при  $n \geq n_0$  справедливо неравенство  $P_c \leq 1/2$ . При этом, конечно, вероятность ошибки  $P_e \geq 1/2$ . Теперь обозначим через  $\epsilon_0$  минимальную вероятность ошибочного кодирования по всем  $n = 1, 2, \dots, n_0 - 1$ .

Поскольку  $\epsilon_0 > 0$ , то примем  $\epsilon = \min\{\epsilon_0, 1/2\}$ . Тогда получаем, что неравенство  $P_e \geq \epsilon$ , справедливое при всех  $n = 1, 2, \dots$

Таким образом, мы определили независящую от длины кода  $n$  нижнюю границу вероятности ошибки  $\epsilon$  для всех кодов со скоростью  $R \leq H - \delta$ .

## 28. Источник с памятью (теорема 1.8 с доказательством).

**ТЕОРЕМА 1.8.** Для любого  $\delta > 0$  имеют место следующие утверждения:

- $\lim_{n \rightarrow \infty} P(T_n(\delta)) = 1$
- Для любого натурального  $n$  справедливо неравенство:

$$|T_n(\delta)| \leq 2^{n(H(X)+\delta)}$$

- Для любого  $\epsilon > 0$  существует  $n_0$  такое, что при всех  $n \geq n_0$ :

$$|T_n(\delta)| \geq (1 - \epsilon)2^{n(H(X) - \delta)}$$

- Для  $x \in T_n(\delta)$ :

$$2^{-n(H(X) + \delta)} \leq p(x) \leq 2^{-n(H(X) - \delta)}$$

**Доказательство.** Первое утверждение следует из (1.29) и (1.30). Второе совпадает с (1.27), четвертое - с (1.26). Остановимся на третьем утверждении.

Из первого утверждения следует, что для любого  $\epsilon > 0$  найдется  $n_0$  такое, что при  $n > n_0$  имеет место неравенство

$$P(T_n(\delta)) \geq 1 - \epsilon \quad (1.36)$$

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

$$P(T_n(\delta)) \leq |T_n(\delta)| \max_{x \in T_n(\delta)} p(x) \leq |T_n(\delta)| 2^{-n(H(X) - \delta)} \quad (1.37)$$

Из (1.36) и (1.37) следует утверждение 3

## 29. Неравномерное кодирование. Понятие префиксного кода. Описание двоичного кодового дерева. Средняя длина кодовых слов.

Для некоторого дискретного источника  $X$  с известным распределением вероятностей  $\{p(x), x \in X\}$  требуется построить эффективный неравномерный двоичный код над алфавитом  $A = \{a\}$ . Рассмотрим кодирование в двоичных символах  $A = \{0, 1\}$ .

Неравномерный побуквенный код  $C = \{c\}$  объемом  $|C| = M$  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.

Если ни одно кодовое слово не является началом другого, код называется префиксным. Префиксные коды являются однозначно декодируемыми. Префиксность - достаточное, но не необходимое условие для однозначной декодируемости.

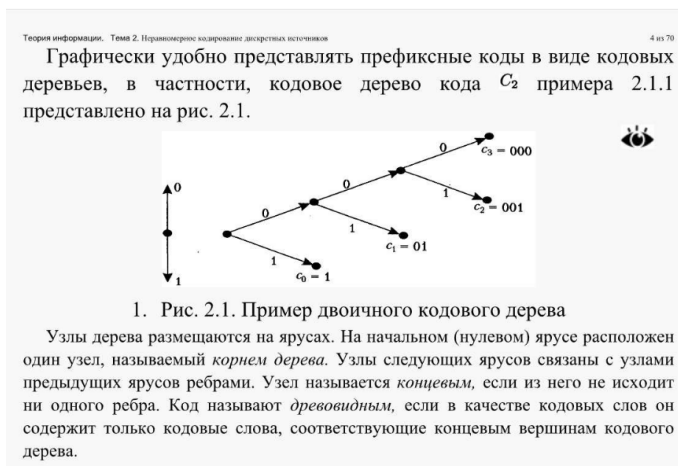


Рис. 1: Описание двоичного кодового дерева

Рассмотрим источник  $X = \{1, \dots, M\}$ , который порождает буквы с вероятностями  $\{p_1, \dots, p_M\}$ . Пусть для кодирования букв источника выбран код  $C = \{c_1, \dots, c_M\}$  с длинами кодовых слов  $l_1, \dots, l_M$  соответственно.

Средней длиной кодовых слов называется величина:

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

## 30. Неравенство Крафта.

**ТЕОРЕМА 2.1.** Необходимым и достаточным условием существования префиксного кода объемом  $M$  с длинами кодовых слов  $l_1, \dots, l_M$  является выполнение неравенства Крафта:

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

### 31. Условие существования префиксного кода (теорема 2.1 с доказательством).

**ТЕОРЕМА 2.1.** Необходимым и достаточным условием существования префиксного кода объемом  $M$  с длинами кодовых слов  $l_1, \dots, l_M$  является выполнение неравенства Крафта:

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

**Доказательство.** Начнем с необходимости.

Рассмотрим двоичное кодовое дерево произвольного префиксного кода объемом  $M$  с длинами кодовых слов  $l_1, \dots, l_M$ . Выберем целое число  $L$  такое, что  $L \geq \max_i l_i$ . Проложим все пути в дереве до яруса с номером  $L$ . На последнем ярусе мы получим  $2^L$  вершин.

Концевая вершина искомого кодового дерева, расположенная на глубине  $l_i$ , имеет два 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.

Множество потомков различных кольцевых 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.

Чтобы убедиться в достаточности, нужно показать, что из справедливости (2.1) вытекает существование кода с заданным набором длин кодовых слов. Будем считать  $l_i$  упорядоченными по возрастанию.

Из общего числа  $2^{l_1}$  вершин на ярусе  $l_1$  выберем одну любую вершину, сделаем ее концевой и закрепим ее за первым кодовым словом. Продолжим оставшиеся вершины до радиуса  $l_2$ . 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. Поделим обе его части на  $2^{l_2}$ . Сделаем одну из этих вершин концевой и закрепим ее за вторым словом. Аналогично, для третьего слова мы получим множество из

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

В силу (2.1) всегда найдется одна для третьего слова. Продолжая построение, на последнем ярусе с номером  $M$  получаем

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

вершин. Это число не меньше 1, если неравенство (2.1) верно.

### 32. Условие существования однозначно декодируемого двоичного кода (теорема 2.2 с доказательством).

**ТЕОРЕМА 2.2.** Для любого однозначно декодируемого двоичного кода объемом  $M$  с длинами кодовых слов  $l_1, \dots, l_M$  справедливо неравенство

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

**Доказательство.** Без потери общности можно считать  $l_M$  наибольшей из длин кодовых слов. Выберем некоторое натуральное число  $N$  и запишем  $N$ -ю степень левой части неравенства в виде

$$\left( \sum_{i=1}^M 2^{-l_i} \right)^N = \underbrace{\left( \sum_{i_1=1}^M 2^{-l_{i_1}} \right) \dots \left( \sum_{i_N=1}^M 2^{-l_{i_N}} \right)}_N = \sum_{i_1=1}^M \dots \sum_{i_N=1}^M 2^{-(l_{i_1} + \dots + l_{i_N})}$$

В правой части имеем  $M^N$  слагаемых. Их мы перегруппируем, упорядочив по величине суммы длин кодовых слов  $l_{i_1} + \dots + l_{i_N}$ . Обозначим через  $A_L$  количество таких последовательностей кодовых слов, сумма длин которых равна  $l_{i_1} + \dots + l_{i_N} = L$ . Тогда

$$\left( \sum_{i=1}^M 2^{-l_i} \right)^N = \sum_{L=1}^{Nl_M} A_L 2^{-L}$$

$A_L$ , то есть число таких последовательностей из  $N$  кодовых слов, суммарная длина которых в точности равна  $L$ , не может быть больше общего числа двоичных последовательностей длиной  $L$ . Таким образом,  $A_L \leq 2^L$ . Поэтому

$$\left( \sum_{i=1}^M 2^{-l_i} \right)^N \leq \sum_{L=1}^{Nl_M} 2^L 2^{-L} = Nl_M$$

Извлечем из обеих частей корень  $N$ -й степени. Получим

$$\sum_{i=1}^M 2^{-l_i} \leq (Nl_M)^{1/N} = 2^{\frac{\log(Nl_M)}{N}}$$

Неравенство справедливо при любых  $N$ . Перейдя к пределу при  $N \rightarrow \infty$ , в правой части получим 1, что и доказывает теорему.

### 33. Прямая теорема побуквенного неравномерного кодирования (теорема 2.3 с доказательством).

**ТЕОРЕМА 2.3.** 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  $\Sigma = \{1, \dots, M\}$  с вероятностями букв  $p_1, \dots, p_M$ . Сопоставим букве  $x_m$  кодовое слово длиной  $l_m = \lceil -\log p_m \rceil$  при  $m = \{1, \dots, M\}$ .

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

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

Средняя длина слов хорошего кода отличается от энтропии не больше, чем на 1. Если энтропия велика, то проигрыш по сравнению с минимально достижимой скоростью можно считать небольшим.

Но предположим, что  $H < 1$ . Например,  $H = 0,1$ . Теорема гарантирует, что существует код со средней длиной кодовых слов не больше 1,1 бита.

Но нам нужно затрачивать на передачу одного сообщения примерно в 10 раз меньше бит! Этот пример показывает, что побуквенное кодирование в этом случае не эффективно.

Предположим, что дан двоичный источник  $X = \{0, 1\}$  с вероятностями букв  $\{\epsilon, 1 - \epsilon\}$ . Минимально достижимая длина кодовых слов наилучшего кода равна 1. Теорема говорит, что средняя длина кодовых слов не больше  $1 + \eta(\epsilon)$ , то есть стремится к 1 при  $\epsilon \rightarrow 0$ . (теорема точна).

### 34. Обратная теорема побуквенного неравномерного кодирования (теорема 2.4 с доказательством).

**ТЕОРЕМА 2.4.** Для любого однозначно декодируемого кода дискретного источника  $X = \{x, p(x)\}$  с энтропией  $H$  средняя длина кодовых слов  $\bar{l}$  удовлетворяет неравенству

$$\bar{l} \geq H$$

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

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

Учитывая, что  $\log x \leq (x - 1) \log e$  получаем

$$H - \bar{l} = \log e \sum_{x \in X} p(x) \left( \frac{2^{-l(x)}}{p(x)} - 1 \right) = \log e \left( \sum_{x \in X} 2^{-l(x)} - \sum_{x \in X} p(x) \right) \leq \log e \left( 1 - \sum_{x \in X} p(x) \right) = 0$$

При каких условиях возможно равенство в обратной теореме?

Для этого равенство должно иметь место в первом и втором неравенствах. В этом случае 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  $\bar{l} = H$ .

### 35. Условие существования неравномерного кода (следствие 2.5).

**СЛЕДСТВИЕ 2.5.** 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.


### 36. Оптимальный побуквенный код Хаффмена.

Рассмотрим ансамбль сообщений  $= \{1, \dots, M\}$  с вероятностями сообщений  $\{p_1, \dots, p_M\}$ . Считаем сообщения упорядоченными по убыванию вероятностей, то есть  $p_1 \geq p_2 \geq \dots \geq p_M$ .

Необходимо построить оптимальный код, то есть код с наименьшей возможной средней длиной кодовых слов.

Пусть двоичный код  $C = \{c_1, \dots, c_M\}$  с длинами кодовых слов  $\{l_1, \dots, l_M\}$  оптимален для рассматриваемого ансамбля сообщений.

Теория информации. Тема 2. Неравномерное кодирование дискретных источников
25 из 70



**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

Рис. 2: Алгоритм кодирования по Хаффману

### 37. Свойства кодовых последовательностей (свойства 2.4.1- 2.4.4 с доказательством).

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

**Доказательство.** Свойство доказывается методом «от противного». Предположим, что  $l_i < l_j$ . Рассмотрим другой код  $C'$ , в котором сообщению  $x_i$  соответствует слово  $c_j$ , а сообщению  $x_j$  - слово  $c_i$ . Тогда 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. Слово максимальной длины можно будет сделать короче по меньшей

**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 (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.

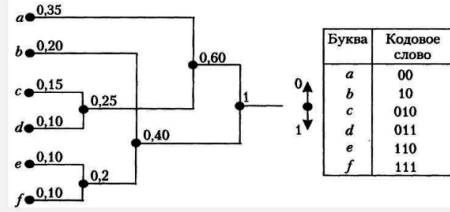


Рис. 2.5. Пример построения кода Хаффмена

Рис. 3: Пример кодирования по Хаффману

мере на 1 символ. При этом уменьшится средняя длина кодовых слов, что противоречит предположению об оптимальности кода.

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

**Доказательство.** Согласно предыдущему свойству, два слова длиной  $l_M$  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 различаются в одном последнем символе.

Для рассматриваемого ансамбля  $X = \{1, \dots, M\}$  и некоторого кода  $C$ , удовлетворяющего свойствам 2.4.1 - 2.4.3, введем вспомогательный ансамбль  $X' = \{1, \dots, M-1\}$  сообщениям которого сопоставим вероятности  $\{p'_1, \dots, p'_{M-1}\}$  следующим образом:  $p'_1 = p_1, \dots, p'_{M-2} = p_{M-2}, p'_{M-1} = p_{M-1} + p_M$ .

Из кода  $C$  построим код  $C'$  для ансамбля  $X'$ , приписав сообщениям  $x'_1, \dots, x'_{M-2}$  те же кодовые слова, что и в коде  $C$ , то есть  $c'_i = c_i, i = 1, \dots, M-2$ , а сообщению  $x'_{M-1}$  - слово  $c'_{M-1}$ , представляющее собой общую часть слов  $c_{M-1}$  и  $c_M$  (согласно свойству 2.4.3, эти два кодовых слова различаются 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 оптимального неравномерного кода. После выполнения алгоритма будет получено кодовое дерево кода, который имеет наименьшую возможную среднюю длину кодовых слов.

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

Отсюда:

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

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  $\bar{l}$ .

Рассмотренные свойства оптимальных префиксных кодов сводят задачу построения кода объемом  $M$  к задаче построения кодов объемом  $M' = M-1$ . Таким образом, получается рекуррентное правило построения кодового дерева.

### 38. Избыточность кода Хаффмена (теорема 2.6).

Разность  $r = \bar{l} - H$  называется избыточностью неравномерного кода.

Она показывает степень «несовершенства» кода в том смысле, что при кодировании с избыточностью  $r$  на каждое сообщение тратится на  $r$  бит больше, чем в принципе можно было бы потратить, если использовать теоретически наилучший (возможно, нереализуемый) способ кодирования.

**ТЕОРЕМА 2.6.** (Р. Галлагер). Пусть  $p_1$  - наибольшая из вероятностей сообщений конечного дискретного ансамбля. Тогда избыточность кода Хаффмена для этого ансамбля удовлетворяет неравенствам

$$r \leq \begin{cases} p_1 + \sigma & \text{при } p_1 < 1/2, \\ 2 - \eta(p_1) - p_1 & \text{при } p_1 \geq 1/2, \end{cases}$$

где  $\eta(x) = -x \log x - (1-x) \log(1-x)$  - энтропия двоичного ансамбля, и

$$\sigma = 1 - \log e - \log \log e \approx 0.08607$$

### 39. Оценка избыточности Галлагера и Манстеттена для кода Хаффмена.

Оценка Галлагера довольно точна в широком диапазоне значений  $p_1$ . Для ансамбля с вероятностями сообщений  $\{1/3, 1/3, 1/3, 0\}$  имеем оценку 0,419. Истинная избыточность кода Хаффмена для этого источника 0,417. Для троичного источника с распределением вероятностей  $\{p_1, 1-p_1, 0\}$  оценка теоремы дает точный результат.

НО существенно улучшить оценку, располагая только значением  $p_1$ , довольно трудно. В то же время для примера 2.4.1 теорема 2.6 дает оценку  $r \leq 0,35 + 0,0861 = 0,4361$  - при том, что истинное значение избыточности  $r = 2,45 - 2,4016 = 0,0484$ .

Оценка Галлагера недостаточно точна при  $p_1 < 0,5$ .

Провел оценку точной границы для всех значений  $p_1$  Манстеттен (рис. 2.6).

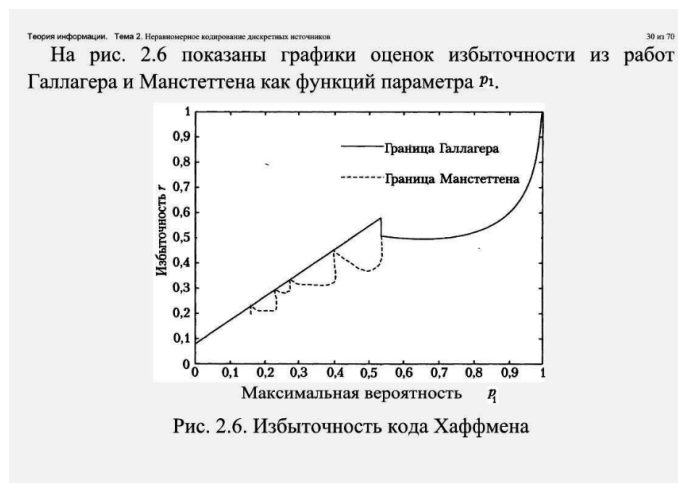


Рис. 4: Оценка избыточности по Галлагеру и Манстеттену

### 40. Код Шеннона (арифметический алгоритм построения, пример).

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

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

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  $q_m$ .

Смотреть рисунки 5 и 6.

### 41. Графическая интерпретация кода Шеннона.

Смотреть рисунок 7.

**Input:** Объем алфавита  $M$ , вероятности букв  $p_i$ ,  $i = 1, \dots, M$

**Output:** Список кодовых слов кода Шеннона

Сортировка:

```
for  $i = 1$  to  $M$  do
   $j(i) \leftarrow$  индекс  $i$ -й по убыванию вероятностей буквы алфавита
end
```

Кумулятивные вероятности:

```
 $q_1 = 0$ ;
for  $i = 2$  to  $M$  do
   $q_i = q_{i-1} + p_{j(i-1)}$ 
end
```

Кодовые слова:

```
for  $i = 1$  to  $M$  do
   $c_i \leftarrow$  первые  $\lceil -\log p_i \rceil$  разрядов после запятой в двоичной записи числа  $q_i$ .
end
```

end

Рис. 2.7. Алгоритм построения кода Шеннона

Рис. 5: Алгоритм кодирования по Шеннону

**Пример 2.6.1.** Рассмотрим тот же ансамбль, что и в примере 2.4.1.

Таблица 2.1.

Построение кода Шеннона для примера 2.6.1

$x$	$p_m$	$q_m$	$l_m$	Двоичная запись $q_m$	Кодовое слово $c_m$
$a$	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

Средняя длина кодовых слов  $\bar{l} = 2,95$ . В данном случае избыточность кода Шеннона оказалась на 0,5 бита больше, чем избыточность кода Хаффмена.

Рис. 6: Пример кодирования по Шеннону

Рассмотрим числовой отрезок  $[0,1)$ , на котором расположим один за другим отрезки длиной  $p_1, \dots, p_M$ .  $M = 3$ ,  $p_1 = 0,6$ ,  $p_2 = 0,3$ ,  $p_3 = 0,1$ , кумулятивные вероятности  $q_1 = 0$ ,  $q_2 = 0,6$ , и  $q_3 = 0,9$  соответствуют началам отрезков. Эти точки идентифицируют сообщения источника.



Рис. 2.9. Графическая интерпретация кода Шеннона

Рис. 7: Графическая интерпретация кода Шеннона



Если требуется закодировать сообщение с номером  $m = 2(q_2 = 0.6)$ :

1. На первом шаге передачей кодового символа 0 либо 1 кодер указывает, в какой половине отрезка  $[0, 1)$  (левой или правой) находится начало соответствующего отрезка. Передается 1, поскольку значение правее середины исходного отрезка ( $q = 0.5$ ). В результате область возможных положений передаваемой точки уменьшается вдвое.

2. На следующем шаге передается символ 0, так как точка находится в левой половине интервала (относительно  $q = 0.75$ ) и длина интервала неопределенности уменьшается до  $1/4$ .

3. После второго шага в интервале неопределенности осталась только одна точка, поэтому передача может быть завершена (длина интервала равна  $1/4$ , что меньше длины кратчайшего прилегающего отрезка 0.3) - именно поэтому гарантируется единственность точки, а значит, и однозначность декодирования.

## 42. Код Гилберта-Мура (алгоритм построения кода, пример).

При построении кода Шеннона требуется упорядоченность сообщений по убыванию вероятностей. В алгоритме построения кода Шеннона наиболее трудоемкой частью является сортировка букв входного алфавита. Сильно упростить построение кода, можно, если модифицировать кодирование таким образом, чтобы упорядоченность не требовалась.

Предположим, что вероятности не упорядочены и 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 (тогда длина слова будет большой, даже если вероятность  $p_m$  велика).

Для того, чтобы избавиться от влияния  $p_{m-1}$  нужно соответствующую сообщению точку переместить 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 интервал неопределенности был 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  $X = \{1, \dots, M\}$  с вероятностями  $\{p_1, \dots, p_M\}$ . Сопоставим каждой букве  $m = 1, \dots, M$  кумулятивную вероятность  $q_m = \sum_{i=1}^{m-1} p_i$  и вычислим для каждой буквы величину  $\sigma_m$  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  $\sigma_m$ .

Теория информации. Тема 2. Неравномерное кодирование дискретных источников

41 из 70

Алгоритм построения кода приведен на рис. 2.10.

**Input:** Объем алфавита  $M$ , вероятности букв  $p_i, i = 1, \dots, M$

**Output:** Список кодовых слов кода Гилберта-Мура

Вспомогательные вероятности:

```
q1 = 0;
for i = 2 to M do
  qi = qi-1 + pi-1
  si = qi + pi/2
end
```

Кодовые слова:

```
for i = 1 to M do
  ci ← первые  $\lceil -\log p_i \rceil + 1$  разрядов после запятой в двоичной записи числа  $\sigma_i$ 
end
```

Рис. 2.10. Алгоритм построения кода Гилберта-Мура

Рис. 8: Алгоритм построения кода Гилберта-Мура

**Пример 2.7.1.** 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 табл. 2.2.



Таблица 2.2.

Пример кода Гилберта-Мура

$x_m$	$p_m$	$q_m$	$\sigma_m$	$l_m$	$c_m^a$	$\tilde{c}_m^b$
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

<sup>a</sup> Кодовые слова кода Гилберта-Мура,

<sup>b</sup> Кодовые слова кода Шеннона без упорядочения вероятностей букв.

Запись  $[a]$  обозначает представление числа  $a$  в двоичной форме. В последнем столбце таблицы показано, как выглядели бы кодовые слова  $\tilde{c}_m$  соответствующего кода Шеннона, если бы мы забыли упорядочить буквы по вероятностям. Код получился бы непrefixным (в отличие от кода Гилберта-Мура).

Рис. 9: Пример построения кода Гилберта-Мура

#### 43. Однозначность декодируемости кода Гилберта-Мура.

Докажем однозначную декодируемость кода Гилберта-Мура в общем случае. Выберем сообщения с номерами  $i$  и  $j$ ,  $i < j$  ( $\sigma_j > \sigma_i$ ). Нужно доказать, что соответствующие слова  $c_i$  и  $c_j$  различаются хотя бы в одном из первых  $\min\{l_i, l_j\}$  кодовых символов.

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

Длина слова и его вероятность связаны соотношением

$$\begin{aligned}l_m &= \lceil -\log(p_m/2) \rceil \geq -\log \frac{p_m}{2} \\ \sigma_j - \sigma_i &\geq \frac{\max\{p_i, p_j\}}{2} \geq 2^{-\min\{l_i, l_j\}}\end{aligned}$$

слова  $c_i$  и  $c_j$  различаются в одном из первых  $\min\{l_i, l_j\}$  двоичных символов, поэтому ни одно из двух слов не может быть началом другого.

Поскольку все слова кода Гилберта-Мура ровно на 1 длиннее слов кода Шеннона, получаем оценку средней длины кодовых слов  $\bar{l} < H + 2$ .

#### 44. Неравномерное кодирование для стационарного источника.

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

Пусть указан некоторый способ кодирования, строящий (для любых  $n$  для каждой последовательности  $x \in X^n$ ) на выходе источника кодовое слово  $c(x)$  длиной  $l(x)$ ). Тогда средняя скорость кодирования для блоков длиной  $n$  определяется как

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

бит/сообщение источника.

Подбирая длину блоков, при которой средняя скорость будет наименьшей, получаем следующее определение для средней скорости кодирования:

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

Здесь используется нижняя грань ( $\inf$ ) вместо минимума, поскольку наименьшее значение скорости может достигаться в пределе при  $n \rightarrow \infty$ .

Рассматриваемое кодирование - FV-кодирование (fixed-to-variable), поскольку блоки из фиксированного числа сообщений  $n$  кодируются кодовыми словами переменной длины.

**45. Средняя скорость кодирования для дискретного стационарного источника (теорема 2.7 с доказательством).**

**ТЕОРЕМА 2.7.** 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 его элементам применим теорему побуквенного кодирования

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

Второе неравенство здесь выполняется так как  $H_n(X)$  не возрастает с увеличением  $n$ . Тогда

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

при любых  $n$ .

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

**46. Прямая теорема кодирования для дискретного стационарного источника (теорема 2.8 с доказательством).**

**ТЕОРЕМА 2.8.** 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

$$\mathbf{M}[l(x)] \leq H(X^n) + 1 = nH_n(X) + 1$$

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

$$|H_n(X) - H| \leq \frac{\delta}{2}, (2.8)$$

из которого следует, что

$$H_n(X) \leq H + \frac{\delta}{2}, n > n_1, (2.9)$$

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

$$\frac{1}{n} \leq \frac{\delta}{2}, (2.10)$$

С учетом (2.9) и (2.10) из (2.8) получаем при  $n \geq \max\{n_1, n_2\}$

$$\bar{R} = \inf_m \bar{R}_m \geq \bar{R}_n = \frac{\mathbf{M}[l(x)]}{n} \leq H_n(X) + \frac{1}{n} \leq H + \frac{\delta}{2} + \frac{\delta}{2} = H + \delta$$

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  $o(n) \rightarrow 0$  при  $n \rightarrow \infty$ .

Но наблюдается экспоненциальный рост сложности при росте длины блоков  $n$ .

**47. Арифметическое кодирование (алгоритм построения кода, пример).**

Сам алгоритм является почти тривиальным обобщением кода Шеннона на последовательности. Рассмотрим дискретный постоянный источник, выбирающий сообщения из множества  $X = \{1, \dots, M\}$  с вероятностями  $\{p_1, \dots, p_M\}$ .

Обозначим через  $\{q_1, \dots, q_M\}$  кумулятивные вероятности сообщений. Необходимо закодировать последовательности множества  $X^n = \{x\}$ . Для записи подпоследовательности  $(x_i, \dots, x_j)$  в последовательности  $x = (x_1, \dots, x_n)$  будем использовать обозначение  $x_i^j$ .

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

Можно использовать код Шеннона или код Гилберта-Мура. Однако использование кода Шеннона предполагает упорядоченность сообщений по убыванию вероятностей. При больших  $n$  сложность упорядочения окажется очень большой, поэтому можно использовать только код Гилберта-Мура.

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

Чтобы вычислить  $q(x)$ , необходимо определить нумерацию последовательностей из  $X^n$ . Будем использовать лексикографический (алфавитный) порядок для последовательностей. Запись  $y \prec x$  будет означать, что  $y$  лексикографически предшествует  $x$ .

Понятие лексикографического порядка определяется следующим образом.

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

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

Лексикографический порядок - это порядок, который обычно используется при составлении словарей.

Основная задача состоит в вычислении кумулятивной вероятности

$$q(x) = \sum_{y \prec x} p(y)$$

Для источника без памяти вероятности последовательностей  $p(x)$ :

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

Выведем рекуррентную формулу для  $q(x)$ . Для этого выразим вероятность  $q(x_1^n)$  через  $q(x_1^{n-1})$ :

$$q(x_1^n) = q(x_1^{n-1}) + p(x_1^{n-1})q(x_n)$$

$$p(x_1^n) = p(x_1^{n-1})p(x_n)$$

Здесь каждая пара значений  $(q(x_1^i), p(x_1^i))$  используется ровно на одном шаге при вычислении следующей пары  $(q(x_1^{i+1}), p(x_1^{i+1}))$ .

При реализации арифметического кодирования вновь вычисленные значения записываются в те же ячейки памяти, в которых находились предыдущие значения.

#### 48. Сложность реализации кодирования арифметического кода

Сложность кодирования. Из описания алгоритма следует, что на каждом шаге кодирования выполняется одно сложение и два умножения. Кажется, что сложность кодирования последовательности из  $n$  сообщений пропорциональна  $n$ . Однако, это неверно, поскольку на каждом шаге линейно растет сложность выполнения самих операций сложения и умножения, так как нарастает число двоичных разрядов, необходимых для записи операндов.

Для представления вероятностей  $\{p_1, \dots, p_M\}$  используются 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

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

Сложность арифметического кодирования пропорциональна  $n^2$

На самом деле все же возможна практическая реализация арифметического кодирования со сложностью  $n$  (за счет потери в точности вычислений).

Теория информации. Тема 2. Неравномерное кодирование дискретных источников
56 из 70

В приведенном на рис. 2.12 алгоритме кумулятивные вероятности  $q(x_i^i), i = 1, 2, \dots$ , хранятся в переменной  $F$ , а вероятности последовательностей  $p(x_i^i), i = 1, 2, \dots$ , - в переменной  $G$ .

**Input:** Объем алфавита  $M$  вероятности букв  $p_i, i = 1, \dots, M$  длина последовательности  $n$  последовательность на выходе источника  $(x_1, \dots, x_n)$ ,

**Output:** Кодовое слово арифметического кода

Кумулятивные вероятности:

```

q1 = 0;
for i = 2 to M do
    qi = qi-1 + pi-1
end
Кодирование:
for i = 1 to n do
    F ← F + q(xi)G
    G ← p(xi)G
end
Формирование кодового слова:
с ← первые ⌈-log G⌉ + 1 разрядов после запятой в двоичной записи числа F + G/2

```

**Рис. 2.12.** Алгоритм арифметического кодирования

Рис. 10: Алгоритм построения арифметического кода

Теория информации. Тема 2. Неравномерное кодирование дискретных источников
57 из 70

**Пример 2.8.1.** Рассмотрим источник из примера 2.7.1:  $X = \{a, b, c\}$ , распределение вероятностей  $p_a = 0,1, p_b = 0,6, p_c = 0,3$ . Вычисления, выполняемые арифметическим кодером при кодировании последовательности  $x = (bcab)$  длиной  $n = 5$ , приведены в табл. 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.

**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 $F + G/2 = 0,5423 \dots \rightarrow \hat{F} = 0,541 \rightarrow 100010101$				

Рис. 11: Пример построения арифметического кода

Если в качестве модели источника рассматривается простая цепь Маркова, то алгоритм кодирования остается прежним - за тем исключением, что вместо одномерных вероятностей  $p(x_i)$  и  $q(x_i)$  нужно использовать условные вероятности  $p(x_i|x_{i-1})$  и

$$q(x_i|x_{i-1}) = \sum_{y \prec x_i} p(y|x_{i-1})$$

#### 49. Декодирование арифметического кода (алгоритм декодирования, пример).

При арифметическом кодировании и использовании последовательностей большой длины  $n$  возникают следующие проблемы:

- арифметическое кодирование требует большой (в пределе - бесконечной) точности вычислений, что ведет к недопустимо высокой сложности реализации;
- для формирования кодового слова формально необходима вся последовательность сообщений, что приводит к недопустимо большой задержке кодирования, равной длине кодируемой последовательности.

Обе эти проблемы преодолимы. Решение состоит в том, что та часть данных, которая не участвует в дальнейших вычислениях и уже не влияет на окончательный результат, может быть исключена из вычислений и сразу выдана на выход кодера. При этом уменьшается сложность вычислений и задержка кодирования.

Теория информации. Тема 2. Нормальное кодирование дискретных источников
66 из 70

Эту задачу решает показанный на рис. 2.15 алгоритм.

**Input:** Объем алфавита  $M$ ,  
вероятности букв  $\{p_1, \dots, p_M\}$ ,  
кумулятивные вероятности букв  $q_i, i = 1, \dots, M$   
длина декодируемой последовательности  $n$  кодовое слово в виде числа  $\hat{F}$ .

**Output:** Декодированная последовательность букв  $(x_1, \dots, x_n)$

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**  
 $S \leftarrow S + q_j G$   
 $G \leftarrow p_j G$   
 $x_i = j$   
**end**  
Результат: последовательность  $(x_1, \dots, x_n)$ ;  
Рис. 2.15. Алгоритм декодирования арифметического кода последовательность

Рис. 12: Алгоритм декодирования арифметического кода

Теория информации. Тема 2. Нормальное кодирование дискретных источников
68 из 70

Таблица 2.4.

Декодирование последовательности из примера 2.8.2

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	$0,5456 > \hat{F}$		

Рис. 13: Пример декодирования арифметического кода

## 50. Формула, определяющая число сочетаний.

Число различных подмножеств:

$$C_M^n = \binom{M}{n} = \frac{A_M^n}{P_n} = \frac{M(M-1) \times \dots \times (M-n+1)}{n!} = \frac{M!}{n!(M-n)!}$$

Это количество называют числом сочетаний из  $M$  элементов по  $n$ .

## 51. Композиция последовательности.

Композицией последовательности  $x$  называется вектор  $\tau(x) = (\tau_0(x), \dots, \tau_{M-1}(x))$  в котором  $\tau_i(x)$  обозначает число элементов  $x_i = i$  в последовательности  $x = (x_1, \dots, x_n)$ .

Необходимо определить число последовательностей  $x$  с заданной композицией  $\tau = (\tau_0, \dots, \tau_{M-1})$ .

Положим  $M = 3$  и найдем число последовательностей длины  $n$  с композицией  $\tau = (\tau_0, \tau_1, \tau_2)$ ,  $n = \tau_0 + \tau_1 + \tau_2$ .

Зафиксируем некоторое расположение нулей (это можно сделать  $\binom{n}{\tau_0}$  способами) и подсчитаем количество различных расположений  $\tau_1$  единиц на оставшихся  $n - \tau_0$  позициях. Это число равно  $\binom{n - \tau_0}{\tau_1}$ .

Поскольку каждому расположению нулей соответствует именно такое количество расположений единиц, общее число последовательностей

$$N(\tau) = \binom{n}{\tau_0} \binom{n - \tau_0}{\tau_1} = \frac{n!}{\tau_0!(n - \tau_0)!} \frac{(n - \tau_0)!}{\tau_1!(n - \tau_0 - \tau_1)!} = \frac{n!}{\tau_0! \tau_1! \tau_2!}$$

Аналогично для алфавита произвольного объема  $M$  справедлива формула

$$N(\tau) = \frac{n!}{\tau_0! \dots \tau_{M-1}!}$$

## 52. Представления натурального числа (лемма 3.1 с доказательством).

**ЛЕММА 3.1.** Натуральное число  $n$  может быть представлено в виде суммы  $M$  неотрицательных целых слагаемых  $\binom{n+M-1}{M-1}$  способами.

**Доказательство.** Будем записывать последовательность из  $j$  нулей в виде  $0^j$ . Каждому разбиению числа  $n$  на целые слагаемые  $a_1 + \dots + a_M = n$  соответствует двоичная последовательность вида  $(0^{a_1}1, 0^{a_2}1, \dots, 0^{a_M})$ . Длина последовательности равна  $n + M - 1$ , а вес равен  $M - 1$ .

Тогда число разбиений равно числу таких последовательностей, которое, в свою очередь, равно числу двоичных последовательностей длиной  $n + M - 1$ , содержащих  $M - 1$  единиц, которое равно  $\binom{n+M-1}{M-1}$ .

## 53. Рекуррентная формула вычисления числа двоичных последовательностей длиной $n$ весом $w$ .

Число  $\binom{n}{w}$  можно интерпретировать как число двоичных последовательностей длиной  $n$  весом  $w$ . Справедлива рекуррентная формула

$$\binom{n+1}{w} = \binom{n}{w} + \binom{n}{w-1}$$

Тогда любая последовательность длиной  $n + 1$  весом  $w$  может быть получена либо из последовательности длиной  $n$  весом  $w$  дописыванием нуля, либо из последовательности длиной  $n$  весом  $w - 1$  дописыванием единицы.

Применим эту формулу ко второму слагаемому. Выполнив это рекурсивно получим

$$\binom{n+1}{w} = \binom{n}{w} + \binom{n-1}{w-1} + \dots + \binom{n-w+1}{1}$$

## 54. Двухпроходное побуквенное кодирование.

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

На практике длина последовательности либо передается отдельно в заголовке файла некоторым стандартным способом, либо в алфавите источника имеется специальный символ, указывающий на завершение файла. В любом случае для всех методов универсального кодирования проблема конца файла решается одинаковым способом, и от способа ее решения сравнительные характеристики алгоритмов не зависят.

Рассмотрим универсальное кодирование для класса источников без памяти с неизвестным распределением вероятностей на буквах источника.

Тогда для решения задачи кодер просматривает последовательность и определяет число появлений  $\tau_n(a)$  каждой буквы  $a \in X$  в последовательности  $x$  длины  $n$ .

Затем, используя полученную информацию, кодер вычисляет оценки вероятностей сообщений и строит по ним некоторый код для множества  $X$ .

На втором проходе этот код используется для кодирования последовательности сообщений источника. Кодовое слово состоит из двух частей  $c(x) = (c_1(x), c_2(x))$ . Первая часть  $c_1(x)$  содержит информацию об использованном коде, вторая  $c_2(x)$  — собственно закодированную последовательность букв.

Декодер сначала по  $c_1(x)$  строит код, затем по  $c_2(x)$  восстанавливает одно за другим закодированные сообщения.

## 55. Подсчет информационного объема для описания двоичного дерева (лемма 3.2).

**ЛЕММА 3.2.** Полное кодовое дерево, имеющее  $M$  концевых вершин, имеет  $M - 1$  промежуточных вершин. Для полного описания дерева достаточно  $2M - 1$  бит.

## 56. Средняя скорость кодирования при двухпроходном кодировании (теорема 3.3 с доказательством).

**ТЕОРЕМА 3.3.** При двухпроходном кодировании с использованием кода Хаффмена дискретного постоянного источника с объемом алфавита  $M$  и энтропией  $H$  средняя скорость кодирования удовлетворяет неравенству

$$\bar{R} \leq H + 1 + \frac{1}{n}(M \log M + 3M - 1)$$

**Доказательство.** Для заданной последовательности  $x$  через  $l_1(x)$  и  $l_2(x)$  обозначаем длину первой и второй частей кодового слова соответственно.

Информация о коде может быть разбита на две части:

1) информация о кодовом дереве и информация о том, какая из букв соответствует каждой вершине кодового дерева. Длина описания кодового дерева может быть оценена с использованием леммы 3.2. Указание буквы для одной вершины потребует не более

$$\lceil \log M \rceil \leq \log M + 1$$

бит.

В результате имеем оценку:

$$l_1(x) \leq 2M - 1 + M \lceil \log M \rceil \leq M \log M + 3M - 1, (3.20)$$

2) Длина второй половины кодового слова подсчитывается как:

$$l_2(x) = \sum_{i=1}^n l(x_i) = \sum_{x \in X} \tau_n(x) l(x) = n \sum_{x \in X} \frac{\tau_n(x)}{n} l(x) = n \sum_{x \in X} \hat{p}_n(x) l(x) = n \mathbf{M}_{\hat{p}_n}[l(x)] \leq n(H(\hat{p}_n) + 1), (3.21)$$

С учетом (3.20) и (3.21) средняя скорость кодирования для заданной последовательности  $x$  удовлетворяет неравенству

$$\bar{R}(x) = \frac{l(x)}{n} = \frac{l_1(x) + l_2(x)}{n} \leq H(\hat{p}_n) + 1 + \frac{1}{n}(M \log M + 3M - 1)$$

Следующий шаг доказательства — усреднение по всем последовательностям  $x$ . В правой части от  $x$  зависит только энтропия:

$$\mathbf{M}[H(\hat{p}_n)] \leq H(\mathbf{M}[\hat{p}_n]) = H(p) = H$$

Здесь неравенство следует из выпуклости  $\cap$  энтропии как функции распределения вероятностей. Для обоснования перехода определим, что

$$\mathbf{M}[\hat{p}_n] = p$$

где через  $p$  обозначен вектор, компонентами которого являются вероятности букв источника.

Иными словами, нужно доказать, что в последнем тождестве равенство имеет место для каждой компоненты

$$\mathbf{M} \left[ \frac{\tau_n(a)}{n} \right] = p(a), a \in X$$

Введем индикаторную функцию

$$\chi_a(x) \leq \begin{cases} 1 & \text{при } x = a, \\ 0 & \text{при } x \neq a, \end{cases}$$

Кроме того,

$$\mathbf{M}[\chi_a(x)] = 1 \times p(a) + 0 \times (1 - p(a)) = p(a)$$

Теперь можно записать:

$$\mathbf{M} \left[ \frac{\tau_n(a)}{n} \right] = \frac{1}{n} \mathbf{M} \left[ \sum_{i=1}^n \chi_a(x_i) \right] = \frac{1}{n} \sum_{i=1}^n \mathbf{M}[\chi_a(x_i)] = p(a), a \in X$$

Отсюда следует (3.25) и из него (3.24). В свою очередь, используя (3.24), в результате усреднения в левой и правой частях (3.22) приходим к доказываемому результату (3.19).

**57. Оценка избыточности нумерационного кодирования. Средняя скорость нумерационного кодирования (теорема 3.4).**

**ТЕОРЕМА 3.4.** При нумерационном кодировании дискретного постоянного источника с объемом алфавита  $M$  и энтропией  $H$  средняя скорость кодирования удовлетворяет неравенству

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

где величина  $K$  не зависит от длины последовательности  $n$ .

**58. Адаптивное кодирование.**



Универсальное кодирование 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. У декодера не возникнет проблем с восстановлением данных, поскольку он располагает всей информацией, использованной кодером для построения кода.

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

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

Рассмотрим эти возможности.

Рассмотрим выбор длины «окна», то есть длины последовательности, по которой оцениваются вероятности сообщений.

Увеличение длины окна вообще приводит к росту точности оценок, то есть к повышению эффективности кодирования.

Это 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, длина которого в несколько раз превышает объем алфавита источника. Дальнейшее увеличение длины окна не приведет заметному уменьшению избыточности. Исходя из этого целесообразным всегда выбирать окна конечной длины.

Однако при этом возрастает сложность кодирования. Так как 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 вероятности обычно оцениваются по всей предшествующей последовательности.

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  $x_1, \dots, x_n$  по формуле

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

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.

Оценку вероятностей букв можно провести по формуле

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

В данной формуле в числителе добавлена единица к счетчику числа появлений букв — тем самым исключаются нулевые вероятности.

К знаменателю нужно добавить объем алфавита, иначе нарушается условие нормировки вероятностей.

### 59. Средняя скорость кодирования при адаптивном арифметическом кодировании (теорема 3.5).

В лекциях и в вопросах РАЗНЫЕ номера теорем и тем!

**ТЕОРЕМА 3.8.** При адаптивном арифметическом кодировании дискретного постоянного источника с объемом алфавита  $M$  и энтропией  $H$  средняя скорость кодирования удовлетворяет неравенству

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

где величина  $K$  не зависит от длины последовательности  $n$ .

## 60. Кодирование А-алгоритмом. Кодирование D-алгоритмом.

Алгоритм А (формула (3.49)) — простой для понимания и анализа, но не самый лучший. Действительно, после кодирования первой буквы источника на втором шаге мы имеем равные вероятности esc-символа и единственного известного кодеру и декодеру первого символа.

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

Это не совсем справедливо, поскольку с esc-символом ассоциируются все остальные буквы алфавита.

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

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

Кодирование с использованием этой оценки назовем D-алгоритмом.

Видно, что для D-алгоритма условие нормировки выполняется и по сравнению с А-алгоритмом заметно увеличена вероятность, приписываемая новым символам, по крайней мере, на первых шагах кодирования.

## 61. Сравнение алгоритмов кодирования.

Сравнение алгоритмов их характеристики сведены в табл. 3.5. В этой таблице в формулах для асимптотической избыточности через  $n$  обозначена длина последовательности,  $M$  - объем алфавита, а величины  $K_i, i = 1, \dots, 5$ , представляют собой константы, не зависящие от  $n$ .

Полученные результаты показывают, что если не принимать во внимание сложность алгоритмов, то предпочтительными являются нумерационное кодирование и адаптивное арифметическое кодирование.

Использование двух проходов при кодировании почти не сказывается на достижимом сжатии. Причина в том, что адаптивное кодирование практически оптимально (его избыточность близка к нижней границе), поэтому выигрыш в принципе большим быть не может.

Таблица 3.5.  
Сравнение алгоритмов универсального кодирования

Алгоритм	Число проходов	Асимптотическая избыточность	Длина слова для текста (3.16)
Двухпроходное кодирование, код Хаффмена	2	$1 + K_1/n$	302
Нумерационное кодирование	2	$\frac{M \log n + K_3}{2n}$	283
Адаптивное кодирование (алгоритм А)	1	$\frac{M \log n + K_4}{2n}$	291
Адаптивное кодирование (алгоритм D)	1	$\frac{M \log n + K_5}{2n}$	283

Рис. 14: Сравнение способов кодирования

**62. Монотонные коды: унарный код, код Голомба, код Галлагера-Ван Вухриса, код Левенштейна, код Элайеса.**

Унарный код. Запись вида  $0^m$  или  $1^m$  означает серию из  $m$  нулей или единиц соответственно. Унарный код сопоставляет числу  $i$  двоичную комбинацию вида  $1^{i-1}0$ . Например, унарными кодами чисел 1, 2 и 3 являются последовательности

$$\text{unar}(1) = 0, \text{unar}(2) = 10, \text{unar}(3) = 110$$

Длина кодового слова для числа  $i$  равна  $l_i = i$ .

Унарный код оптимален (минимизирует среднюю длину кодовых слов), если числа  $i$  распределены по геометрическому закону

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

с параметром  $\alpha = 1/2$ , то есть при  $p_i = 2^{-i}, i = 1, 2, \dots$

Для значений  $\alpha > 1/2$  более эффективен код Голомба. Код Голомба. 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 — двоичная запись остатка от деления  $i$  на  $T$  в виде последовательности длины  $m$ . 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

$$(110)(101) = 110101$$

т.е.

$$(\text{unar}(3))(\text{bin}(5))$$

В технической литературе код Голомба иногда называют кодом Райса.

Код Галлагера-Ван Вухриса — естественное обобщение кода Голомба на случай, когда параметр  $T$  не является степенью двойки. Вычисляется префикс — это закодированное унарным кодом число  $\lfloor i/T \rfloor$ . 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, а для остальных (больших) значений используется  $m + 1$  бит. 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.

Оптимальное значение параметра  $T$  для кода Галлагера-Ван Вухриса связано с параметром геометрического распределения  $\alpha$  соотношением

$$\alpha^T + \alpha^{T+1} \leq \alpha^T + \alpha^{T-1}$$

Благодаря чрезвычайно простой схеме кодирования и достаточно высокой эффективности коды Голомба и Галлагера-Ван Вухриса часто применяют при кодировании аудио- и видеосигналов.

Код Левенштейна. Предположим, что нужно передать число  $i = 21$ . Двоичное представление этого числа имеет вид 10101. Непосредственно использовать при кодировании двоичные представления натуральных чисел нельзя, ибо 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

$$[(\text{unar}(5))(\text{bin}(21))] \rightarrow (11110)(10101) = 1111010101$$

В общем случае длина двоичного представления будет равна  $2\lceil \log i \rceil$ .

Пошагово улучшим способ кодирования - первая значащая цифра двоичной записи числа - всегда 1. Ее можно не передавать. декодер сам допишет недостающую единицу, если будет знать длину двоичной записи. Обозначим через  $\text{bin}'(i)$  двоичную запись натурального числа  $i$  без первой единицы, прямыми скобками обозначается длина двоичной последовательности.

Итак, простой монотонный код числа имеет следующую структуру:

$$\text{mon}(i) = (\text{unar}(|\text{bin}'(i)| + 1), \text{bin}'(i))$$

Длины кодовых слов этого монотонного кода

$$l_i = 2\lceil \log i \rceil + 1$$

Чтобы сделать запись еще короче, с длиной двоичной записи можно поступить так же, как и с самим числом, то есть 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.

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.

Упрощенный код Левенштейна (код Элайеса). Более простой код приведен в работе Элайеса. Числу  $i = 1$  припишем кодовое слово  $elias(1) = 0$ . Для чисел  $i > 1$  кодовые слова вычисляются по следующему правилу:

$$elias(i) = (unar(|bin'(|bin'(i))| + 2), bin'(|bin'(i))|, bin'(i))$$

Кодовое слово состоит из 3 частей. Справа (в третьей части) записано двоичное представление числа без первой единицы. ей предшествует вторая часть, в которой пишется двоичное представление длины третьей части, тоже без первой единицы. Наконец, в первой части записан унарный код увеличенной на 2 длины второй части кодового слова.

Длина кодового слова кода Элайеса для произвольного числа  $i$

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

### 63. Теорема о средней длине кодовых слов (теорема 4.1 с доказательством).

**ТЕОРЕМА 4.1.** 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  $o(H) \rightarrow 0$  при  $H \rightarrow \infty$ .

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

$$p_i \leq \frac{1}{i}, i \leq \frac{1}{p_i}, i = 1, 2, \dots$$

$$\bar{l} = \sum_{i=1}^{\infty} l_i p_i \leq \sum_{i=1}^{\infty} p_i (\log i + 2 \log \log i + 2) \leq \sum_{i=1}^{\infty} p_i \log \frac{1}{p_i} + 2 \sum_{i=1}^{\infty} p_i \log \left( 1 + \log \frac{1}{p_i} \right) + 2 \leq H + 2 \log(1 + H) + 2$$

Здесь первое неравенство основано на (4.5). второе использует (4.6). Последнее неравенство основано на выпуклости  $\cap$  функции  $\log x$ . Из (4.1) следует утверждение теоремы.

### 64. Интервальное кодирование.

Интервальное кодирование. Пусть  $X = \{0, 1, \dots, M - 1\}$  - алфавит источника и на выходе источника наблюдается последовательность  $x_1, x_2, \dots$ . Алгоритм описывается рекуррентно.

Предположим, что начальная часть последовательности  $x_1^{n-1} = (x_1, \dots, x_{n-1})$  уже закодирована и передана и, следовательно, известна декодеру.

Вместо буквы  $x_n$ , передается длина интервала (количество букв) между предыдущим и текущим появлением данной буквы т. е. кодер вычисляет и передает минимальное число  $r_n = r$  такое, что  $x_{n-r} = x_n$ .

Исходная последовательность  $x_1, x_2, \dots$  преобразуется в последовательность чисел  $r_1, r_2, \dots$ . Для завершения описания алгоритма нужно определить правило вычисления интервалов для тех букв, которые не встречались в последовательности  $x_1^{n-1}$ . Проще всего условиться о том, что первой передаваемой букве предшествовали все буквы алфавита в заранее согласованном (для определенности, в алфавитном) порядке.

### 65. Метод «стопка книг».

Метод «стопка книг».

Этот метод известен под названиями «move-to-front coding» и «recency rank coding» («кодирование степени новизны»).

Как и при кодировании длин интервалов, предположим, что начальная часть последовательности  $x_1^{n-1} = (x_1, \dots, x_{n-1})$  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  $x_{n-r+1}^{n-1}$ . Тем самым исходная последовательность  $x_1, x_2, \dots$  преобразуется в последовательность чисел  $d_1, d_2, \dots$ . Для передачи букв, которые не встречались в  $x_1^{n-1}$ , можно использовать стратегию для интервального кодирования.

### 66. Теорема о средней скорости интервального кодирования (теорема 4.2 с доказательством).

**ТЕОРЕМА 4.2.** Пусть  $H(X)$  — энтропия одномерного распределения дискретного стационарного источника. Средняя скорость интервального кодирования в сочетании с использованием кода Элайеса удовлетворяет неравенству

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

где  $o(H) \rightarrow 0$  при  $H \rightarrow \infty$ .

**Доказательство.** Обозначим через  $r_i(a)$  длину интервала между  $i$ -м и  $i-1$ -м появлениями буквы  $x = a$  и через  $l(a)$  соответствующие затраты на передачу буквы. В соответствии с оценкой (4.5)

$$l_i(a) \leq \log r_i(a) + 2 \log(1 + \log r_i(a)) + 2, (4.8)$$

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 (4.8) после усреднения получаем

$$\bar{l}(a) \leq \log \bar{r}(a) + 2 \log(1 + \log \bar{r}(a)) + 2$$

В соответствии с леммой Каца (Кас) (ее доказательство приведено ниже) имеет место неравенство

$$\bar{r}(a) \leq \frac{1}{p(a)}$$

Подставим эту оценку в (4.8) и усредним по всем буквам алфавита. Далее снова используем выпуклость логарифма. В результате получим

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

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 простой реализации кодера и декодера.

#### 67. Лемма о случайной величине (лемма 4.3 с доказательством).

**ЛЕММА 4.3.** Пусть случайная величина  $i$  принимает значения из множества натуральных чисел  $1, 2, \dots$  в соответствии с распределением вероятностей  $\{p_i, i = 1, 2, \dots\}$  Тогда

$$M[i] = \sum_{j=1}^{\infty} P(i \geq j)$$

**Доказательство.**  $P(i \geq j) = \sum_{i=j}^{\infty} p_i$ . Сумма в правой части содержит каждое слагаемое  $p_i$  ровно  $i$  раз. т.е. эта сумма равна математическому ожиданию  $i$ .

В теории вероятностей для дискретного случайного процесса  $x_0, x_1, \dots$  длины интервалов между одинаковыми значениями процесса  $x_t = a$  называют временем возвращения процесса в состояние  $a$ . Более точно, время возвращения в состояние  $a$  для стационарного процесса определяется как

$$r = \min\{k \geq 1 : x_0 = x_k = a\}$$

Распределение вероятностей времени возвращения в  $a$  определяется формулой

$$q_a(r) = P(x_1, \dots, x_{r-1} \neq a, x_r = a | x_0 = a)$$

соответственно, среднее время возвращения в  $a$

$$\bar{r}(a) = \sum_{r=1}^{\infty} r q_a(r)$$

#### 68. Лемма Каца (лемма 4.4 с доказательством).

**ЛЕММА 4.4.** (Лемма Каца) Для дискретного стационарного источника такого, что  $P(a) > 0$  справедливо соотношение

$$\bar{r}_a p(a) = P(x_n = a \text{ хотя бы для одного } n, n = 0, 1, \dots) (4.15)$$

В частности, для эргодического источника

$$\bar{r}(a) \leq \frac{1}{p(a)}$$

**Доказательство.** Рассмотрим произведение в левой части (4.15). Воспользовавшись леммой 4.3, получим

$$\bar{r}_a p(a) = p(a) \sum_{t=1}^{\infty} r q_a(r) = p(a) \sum_{t=1}^{\infty} Q_a(t)$$

где использовано обозначение

$$Q_a(t) = \sum_{j=t}^{\infty} q_a(j) = P(r \geq t)$$

Событие  $r \geq t$  (время возвращения в  $a$  не меньше  $t$ ) имеет место в том случае, когда вслед за буквой  $a$  порождены  $t$  букв. ни одна из которых не совпадает с  $a$ , то есть

$$Q_a(t) = P(x_1, \dots, x_t \neq a | x_0 = a)$$

Подстановка этого выражения дает

$$\bar{r}_a p(a) = P(x_0 = a) \sum_{t=1}^{\infty} P(x_1, \dots, x_t \neq a | x_0 = a) = \sum_{t=1}^{\infty} P(x_0 = a, x_1, \dots, x_t \neq a)$$

Введем распределение вероятностей на множестве последовательностей источника. Оно соответствует инверсии во времени последовательностей исходного источника. обозначим его как  $\tilde{p}(x)$ ,  $x \in X^n$ ,  $n = 1, 2, \dots$

Для произвольного  $n$  и любой последовательности  $x = (x_1, \dots, x_n)$  обозначим через  $\overleftarrow{x}$  последовательность, полученную из  $x$  инвертированием порядка следования символов, то есть  $\overleftarrow{x} = (x_n, \dots, x_1)$ . По определению,  $\tilde{p}(\overleftarrow{x}) = p(x)$ . Вероятности, вычисленные относительно этого распределения, будем обозначать как  $\tilde{p}(\cdot)$ . Из последнего равенства с учетом стационарности инверсного процесса имеем

$$\bar{r}_a p_a(t) = \sum_{t=1}^{\infty} \tilde{P}(x_0, \dots, x_{t-1} \neq a, x_t = a) = \tilde{P}(x_t = a \text{ хотя бы для одного } t, t = 1, 2, \dots)$$

Вероятности появления хотя бы одной буквы  $a$  для исходного процесса и инверсного одинаковы. Для любого эргодического процесса и любой буквы  $a$  такой, что  $p(a) > 0$  эта вероятность равна 1.

## 69. Метод скользящего словаря (LZ-77): Алгоритм LZ-77, сложность алгоритма LZ-77, алгоритм LZFG.

Данный алгоритм после его опубликования Зивом и Лемпелом в был многократно модифицирован, некоторые модификации получили самостоятельные названия с аббревиатурами вида LZ $X$ , где  $X$  - первая буква имени автора модификации.

Рассмотрим формальное описание алгоритма.

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

Пусть  $X = \{0, 1, \dots, M-1\}$  — алфавит источника и на выходе источника наблюдается последовательность  $1, 2, \dots, n$ . Алгоритм работы кодера показан на рис. 4.1.

Кодер LZ-77 хранит в памяти скользящий словарь объемом  $W$ . 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 слова от текущей позиции и длины словарного слова.

Сложность алгоритма. Сложность кодирования и декодирования не равны. Вычислительная сложность декодирования очень низкая. Декодер непосредственно из битового потока получает информацию о том, начиная с какой буквы и какой длины последовательность букв он должен извлечь из памяти и выдать получателю.

Кодер для каждого из значений  $l = 1, 2, \dots$  должен повторить попытку поиска в окне длиной  $W$  (типичные размеры окна составляют от 2048 до 16384 символов) образцов, совпадающих с вновь поступившими буквами источника. Такая «прямолинейная» реализация алгоритма неприемлемо сложна. Поэтому применяемые в архиваторах кодеры для быстрого поиска образцов в словаре кодера используют хеширование.

---

```

Заданы:
алфавит источника  $X = \{0, 1, \dots, M-1\}$ ;
длина «окна»  $W$ 
Input: Длина последовательности  $n$ ;
Последовательность источника  $x = x_1^n$ ;
Output: Кодовое слово для  $x$ ;
Инициализация:
 $N = 0$ ,  $c$  - «пустое» слово;
while  $N < n$  do
Находим максимальное  $l$  такое, что  $x_{N+1}^{N+l} = x_{N-d+1}^{N-d+l}$ 
некоторого  $d \in \{1, \dots, W\}$ ;
if  $l > 0$  then (последовательность найдена)
    ► к кодовому слову  $c$  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  $c$  дописываются:
        флаг 0;
        новая буква  $x_{N+1}$  в виде двоичной последовательности
        длины  $\lceil \log M \rceil$ ;
    ►  $N \leftarrow N + 1$ 
end

```

Рис. 4.1. Алгоритм LZ-77

Рис. 15: Алгоритм кодирования LZ-77

---

```

Заданы:
алфавит источника  $X = \{0, 1, \dots, M-1\}$ ; длина «окна»  $W$ 
Input: Длина последовательности  $n$ ;
Последовательность источника  $x = x_1^n$ ;
Output: Кодовое слово  $c$  для  $x$ ;
Инициализация:
 $N = 0$ ,  $c$  - «пустое» слово;
while  $N < n$  do
Находим максимальное  $l$  из диапазона  $\{3, \dots, 17\}$  такое, что
 $x_{N+1}^{N+l} = x_{N-d+1}^{N-d+l}$  для некоторого  $d \in \{1, \dots, W\}$ ;
if  $l > 2$  then
    ► к кодовому слову  $c$  дописываются:
        число  $l$  в виде ненулевой двоичной последовательности длиной 4
        (числам  $l = 3, \dots, 17$  соответствуют комбинации 0001, ..., 1111);
        расстояние до образца  $d$  в виде двоичной последовательности длиной  $\lceil \log W \rceil$ ;
    ►  $N \leftarrow N + l$ 
else ( $l \leq 2$ )
    ► к кодовому слову  $c$  дописываются:
        последовательность 0000;
        число букв  $L \in \{1, \dots, 16\}$ , поступивших на вход кодера непосредственно с выхода
        источника (без кодирования), числами  $1, \dots, 16$  соответствуют комбинации 0000, ..., 1111
         $L$  букв источника, передаваемых без кодирования в виде двоичных последовательностей
        длиной  $\lceil \log M \rceil$ 
    ►  $N \leftarrow N + L$ 
end

```

Рис. 4.2. Алгоритм LZFG

Рис. 16: Алгоритм кодирования LZFG

Вариант алгоритма LZFG (авторы Фиала и Гриине), приведенный на рис. 4.2.

## 70. Алгоритм LZW (LZ-78).

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  $X = \{0, 1, \dots, M-1\}$  — алфавит источника и на выходе источника наблюдается последовательность  $x_1, x_2, \dots$ . Для простоты описания алгоритма будем считать, что в начале работы кодера каждая из букв алфавита является словом длиной 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 рис. 4.3. Исключение составляет первый шаг, когда словарь полностью известен кодеру и декодеру.

## 71. Предсказание по частичному совпадению: идея алгоритма PPM.

```

Задан алфавит источника  $X = \{0, 1, \dots, M-1\}$ ;
Input: Длина последовательности  $n$ ;
Последовательность источника  $x = x_1^n$ ;
Output: Кодовое слово  $c$  для  $x$ ;
Инициализация:
 $N = 0$ ;
 $c$  - «пустое» слово;
Словарь состоит из  $M$  слов длины 1, то есть из букв алфавита  $X$ 
Число слов в словаре  $c = M$ 
while  $N < n$  do
  ► Находим максимальное  $l$  такое, что  $x_{N+1}^{N+l}$  совпадает с  $j$ -м
  словом словаря
  при 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 длиной  $l+1$  вида
   $x_{N+1}^{N+l+1} = (x_{N+1}^{N+l}, x_{N+l+1})$ 
  ►  $N \leftarrow N+l$ 
  ►  $c \leftarrow c+1$ 
end

```

Рис. 4.3. Алгоритм LZW

Рис. 17: Алгоритм кодирования LZW

Рассмотрим наиболее «логичный» алгоритм кодирования при неизвестной статистике источника. На первый взгляд он является почти тривиальным развитием адаптивного арифметического кодирования.

Разница состоит в том, что в процессе кодирования вместо безусловных вероятностей букв оцениваются их условные вероятности, при известном «контексте», то есть при известных предшествующих буквах. Этот метод кодирования получил название PPM (prediction by partial matching).

Предположим, что последовательность  $x_1^t = (x_1, \dots, x_t)$  первых  $t$  букв источника уже передана и предстоит передать символ  $x_{t+1}$ .

При кодировании очередной буквы выполняются следующие шаги.

1. Находим контекст  $x_{t-d+1}^t$  наибольшей длины  $d$ , не превышающей заданной величины  $D$ . Под контекстом — понимается последовательность  $s = x_{t-d+1}^t$ , непосредственно предшествующая кодируемому символу и такая, что в точности такая же последовательность  $s$  уже встречалась в — переданной последовательности  $x_1^{t-1}$ .
2. Для всех возможных значений символа  $x_{t+1}$  вычисляются оценки условных вероятностей символа при известном контексте  $s$ .
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, то есть вычисления оценки условной вероятности символа.

```

Заданы алфавит источника  $X = \{0, 1, \dots, M-1\}$ , максимальная длина контекста  $D$ 
Input: Длина последовательности  $n$ ;
Последовательность источника  $x = x_1^n$ ;
Output: Кодовое слово  $c$  для  $x$ ;
Инициализация:
 $t = 0$ 
 $c$  - «пустое» слово;
while  $t < n$  do
  Находим наибольшее  $d$  такое, что  $\hat{p}(x_{t-d+1}^t) > 0, d \leq D$ 
  Выбираем контекст  $s = x_{t-d+1}^t$ 
  while  $\hat{p}(x_{t+1}|s) = 0$  do
    Кодруем  $esc$  в соответствии с  $\hat{p}(esc|s)$ 
    Уменьшаем длину контекста
     $d \leftarrow d-1, s = x_{t-d+1}^t$ 
  end
  if  $d > 0$  then
    Кодруем  $x_{t+1}$  в соответствии с  $\hat{p}(x_{t+1}|s)$ 
  else
    Кодирование без контекста
    if  $\hat{p}_0(x) > 0$  then
      Кодруем  $x_{t+1}$  в соответствии с  $\hat{p}_0(x)$ 
    else
      Вычисляем  $\hat{p}(esc)$  и передаем  $esc$ .
      Кодруем  $x_{t+1}$  в соответствии с равномерным распределением
      не встречавшихся в  $x_1^t$  букв
    end
  end
   $t \leftarrow t+1$ 
end

```

Рис. 4.4. Идея алгоритма PPM

Рис. 18: Алгоритм кодирования PPM

## 72. Сжатие с использованием преобразования Барроуза-Уилера.

Рассмотрим стационарный источник. Предположим, что модель источника может быть с достаточной точностью аппроксимирована цепью Маркова конечно связности  $m$ .

При передаче длинной последовательности  $x = (x_1, \dots, x_m)$  подпоследовательность  $s = (s_1, \dots, s_m)$  встретится в тексте длиной  $n$  приблизительно  $np(s)$  раз. После циклического упорядочивания мы получим  $np(s)$  строк,



которые будут начинаться с  $s = (s_1, \dots, s_m)$ , распределение вероятностей букв в последнем столбце в этой части таблицы будет иметь вид  $p(x|s)$ .

Использование одного из универсальных кодов, например, метода «стопки книг», позволит потратить на передачу этой части последовательности примерно  $np(s)H(X|s)$  бит. Суммируя по всем последовательностям  $s$ , приходим к скорости кодирования близкой к  $H(X|X^m)$ .

Выход преобразователя — это нестационарная последовательность, которая состоит из стационарных подпоследовательностей, соответствующих различным состояниям исходного источника.

Данный алгоритм хорошо подходит для источников, которые могут быть аппроксимированы моделью с относительно небольшим числом состояний, причем при известном состоянии условное распределение вероятностей должно быть близким к вырожденному (иметь небольшую энтропию). Этим условиям в большой степени удовлетворяют некоторые тексты, листинги программ и т. п.

### 73. Сравнение способов кодирования, использующихся для сжатия.

Сравним рассмотренные алгоритмы. Используем для сравнения текст из 50 букв, на котором мы иллюстрировали принципы работы алгоритмов. Результаты вычислений сведены в табл. 4.10. Хотя маленькая длина текста не позволяет делать надежных выводов о сравнительных характеристиках алгоритмов, тем не менее приведенные в таблице данные довольно — показательны.

Предложенные относительно недавно алгоритмы RPRM и кодирование с использованием преобразования Барроуза-Уилера существенно эффективнее других методов кодирования.

Однако при сравнении рассмотренных алгоритмов в реальных условиях применения можно увидеть, что однозначного решения эта задача не имеет. В частности, при кодировании различных файлов данных предпочтительными будут различные алгоритмы.

Алгоритм	Затраты (бит)
Передача без кодирования	400
Двухпроходное кодирование, код Хаффмена	302
Нумерационное кодирование	283
Адаптивное арифметическое кодирование (алгоритм A)	293
Адаптивное арифметическое кодирование (алгоритм D)	283
Алгоритм LZ77	280
Алгоритм LZFG	299
Алгоритм LZW	291
Алгоритм PPMa	250
Алгоритм RPRM	<b>232</b>

Рис. 19: Сравнение алгоритмов для сжатия данных

### 74. Скорость помехозащищенного кода. Пропускная способность канала.

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

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

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

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 длиной  $n$ .

Число  $C$  называется пропускной способностью канала, если при любой скорости кода  $R < C$  существуют коды, обеспечивающие сколь угодно малую вероятность ошибки и, напротив, при  $R > C$  существует константа  $\epsilon > 0$  такая, что вероятность ошибки любого кода ограничена снизу величиной  $\epsilon$  (то есть вероятность ошибки не может быть сделана сколь угодно малой).

### 75. Модель канала. Канал без памяти. Двоичный симметричный канал (ДСК). Двоичный симметричный канал со стираниями (ДСТК).

Модель канала задана, если для любых  $n$  и любых последовательностей  $x \in X^n$ ,  $x \in X^n$  указано правило вычисления условной вероятности  $p(y|x)$ .

Обозначение  $x_i^n = (x_i, \dots, x_n)$ : канал называется стационарным, если для любых  $j$  и  $n$  и любых  $x_{j+1}^{j+n} \in X^n$ ,  $y_{j+1}^{j+n} \in Y^n$  условные вероятности  $p(y_{j+1}^{j+n}|x_{j+1}^{j+n})$  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  $x_{j+1}^{j+n} \in X^n$ ,  $y_{j+1}^{j+n} \in Y^n$

$$p(y_{j+1}^{j+n}|x_{j+1}^{j+n}) = \prod_{i=j+1}^{j+n} p(y_i|x_i)$$

Согласно данному определению, в канале без памяти события, происходящие при передаче последовательных символов, независимы.

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

Классификация каналов аналогична классификации источников. Однако задача кодирования для каналов значительно сложнее.

Для описания ДПК достаточно указать одномерные условные вероятности  $\{p(y|x), x \in X, y \in Y\}$ .

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

$$\begin{pmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,L-1} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{K-1,0} & p_{K-1,1} & \cdots & p_{K-1,L-1} \end{pmatrix}$$

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

Пример 5.2.1. 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

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

Пример 5.2.2. Двоичный симметричный канал со стираниями (ДСтК). Матрица переходных вероятностей имеет вид

$$P = \begin{pmatrix} 1-p-\epsilon & \epsilon & p \\ p & \epsilon & 1-p-\epsilon \end{pmatrix}$$

Входной алфавит ДСтК содержит два символа, 0 и 1, а выходной алфавит помимо этих двух символов содержит дополнительный символ  $\epsilon$ , который называют символом стирания.

В ДСтК каждый из входных символов может с вероятностью  $p$  превратиться в противоположный символ и с вероятностью  $\epsilon$  может принять неопределенное значение, называемое стиранием.

Вероятности ошибки  $p$  стирания  $\epsilon$  постоянны, они не зависят от результатов передачи предыдущих и следующих символов и от значения передаваемого в данный момент символа.

## 76. Дискретный постоянный канал. Матрица переходных вероятностей канала.

Канал называется каналом без памяти, если для любых  $j$  и  $n$  и любых  $x_{j+1}^{j+n} \in X^n$ ,  $y_{j+1}^{j+n} \in Y^n$

$$p(y_{j+1}^{j+n}|x_{j+1}^{j+n}) = \prod_{i=j+1}^{j+n} p(y_i|x_i)$$

Согласно данному определению, в канале без памяти события, происходящие при передаче последовательных символов, независимы.

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

Классификация каналов аналогична классификации источников. Однако задача кодирования для каналов значительно сложнее.

Для описания ДПК достаточно указать одномерные условные вероятности  $\{p(y|x), x \in X, y \in Y\}$ .

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

$$\begin{pmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,L-1} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{K-1,0} & p_{K-1,1} & \cdots & p_{K-1,L-1} \end{pmatrix}$$

Эта стохастическая матрица называется матрицей переходных вероятностей канала. Она полностью описывает модель ДПК.

## 77. Взаимная информация канала. Средняя взаимная информация канала. Свойства взаимной информации.

В каналах связи важным является возможность получения информации о передаваемых сообщениях по сигналам, 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

$$I(x; y) = I(x) - I(x|y)$$

В выражении  $I(x; y)$  в качестве разделителя используется точка 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.

Свойства взаимной информации:

- Свойство 5.3.1. Симметричность:  $I(x; y) = I(y; x)$
- Свойство 5.3.2. Если  $x$  и  $y$  независимы, то  $I(x; y) = 0$

Средней взаимной информацией между ансамблями  $X$  и  $Y$  называется величина

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

Свойства средней взаимной информации:

- Свойство 5.3.3. Симметричность:  $I(X; Y) = I(Y; X)$
- Свойство 5.3.4. Неотрицательность:  $I(X; Y) \geq 0$
- Свойство 5.3.5. Тожество  $I(X; Y) = 0$  имеет место тогда и только тогда, когда ансамбли  $X$  и  $Y$  независимы.
- Свойство 5.3.6.  $I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(XY)$
- Свойство 5.3.7.  $I(X; Y) \leq \min\{H(X), H(Y)\}$
- Свойство 5.3.8.  $I(X; Y) \leq \min\{\log |X|, \log |Y|\}$
- Свойство 5.3.9. Взаимная информация  $I(X; Y)$  - выпуклая  $\cap$  функция распределения вероятностей  $p(x)$
- Свойство 5.3.10. Взаимная информация  $I(X; Y)$  - выпуклая  $\cup$  функция условных распределений  $p(y|x)$

## 78. Условная средняя взаимная информация. Теорема о переработке информации.

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

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

Разумеется, средняя условная информация обладает всеми свойствами средней взаимной информации.

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 5.1.** Пусть  $X$ ,  $Y$  и  $Z$  - вероятностные ансамбли, формируемые системой последовательной обработки информации. Тогда имеют место неравенства:

$$I(X; Y) \geq I(X; Z)$$

$$I(Y; Z) \geq I(X; Z)$$

## 79. Информационная емкость канала.

Величина

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

называется информационной емкостью канала.

Пусть, например,  $A = \{(n-1)/n, n = 1, 2, \dots\}$ . Максимального числа в этом множестве не существует, но  $\sup A = 1$ , причем эта верхняя грань совпадает с пределом последовательности при  $n \rightarrow \infty$ .

Аналогично в определении информационной емкости канала наибольшее значение средней взаимной информации на букву канала может достигаться при бесконечной длине кода, поэтому в формуле используется именно верхняя грань, а не максимальное значение.

Информационную емкость канала  $C_0$  определим как некоторую функцию его переходных вероятностей. Мы предполагаем, что именно эта величина характеризует потенциально достижимую скорость передачи по каналу.

## 80. Неравенство Фано (с доказательством)

**ТЕОРЕМА 5.2.** (Неравенство Фано)

$$H(U|V) \leq \eta(P_e) + P_e \log(M-1), \quad (5.15)$$

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

**Доказательство.** Используя (5.13) и (5.14), запишем выражения, участвующие в неравенстве Фано (5.15), в следующем виде:

$$H(U|V) = - \sum_u \sum_{v \neq u} p(u, v) \log p(u|v) - \sum_u \sum_{v=u} p(u, v) \log p(u|v), \quad (5.17)$$

$$\eta(P_e) = - \sum_u \sum_{v \neq u} p(u, v) \log P_e - \sum_u \sum_{v=u} p(u, v) \log P_c, \quad (5.18)$$

$$P_e \log(M-1) = \sum_u \sum_{v \neq u} p(u, v) \log(M-1), \quad (5.19)$$

Рассмотрим разность

$$\Delta = H(U|V) - \eta(P_e) - P_e \log(M-1)$$

Чтобы доказать (5.15), нужно доказать, что  $\Delta \leq 0$ . Для этого вычтем из правой и левой части (5.17) соответствующие части тождеств (5.18) и (5.19). 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

$$\Delta \leq (\log e) \left[ \sum_u \sum_{v \neq u} p(u, v) \log \frac{P_e}{p(u|v)(M-1)} - \sum_u \sum_{v \neq u} p(u, v) + \sum_u \sum_{v=u} p(u, v) \log \frac{P_c}{p(u|v)} - \sum_u \sum_{v=u} p(u, v) \right]$$

Воспользуемся тем, что  $p(u, v) = p(v)p(u|v)$ , и обозначениями (5.13) и (5.14).

Результат легко привести к виду

$$\Delta \leq (\log e) \times \left[ \frac{P_e}{M-1} \sum_u \sum_{v \neq u} p(v) - P_e + P_c \sum_u \sum_{v=u} p(v) - P_c \right], \quad (5.20)$$

Заметим теперь, что

$$\sum_u \sum_{v \neq u} p(v) = (M-1) \sum_v p(v) = M-1, \quad (5.21)$$

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

$$\sum_u \sum_{v=u} p(v) = \sum_v p(v) = 1, \quad (5.22)$$

Подстановка (5.21) и (5.22) в (5.20) приводит к неравенству  $\Delta \leq 0$ . Тем самым неравенство Фано доказано.

### 81. Теорема для последовательностей, составленных из элементов множества объема $M$ (теорема 5.3)

**ТЕОРЕМА 5.3.** Для последовательностей  $(u, v) \in U^N V^N$ , составленных из элементов множества объема  $M$ , имеет место неравенство

$$\frac{1}{N} H(U^N | V^N) \leq \eta(\bar{P}_e) + \bar{P}_e \log(M-1)$$

**Доказательство.** Напомним, что условная энтропия не возрастает с увеличением числа условий. Поэтому

$$H(U^N | V^N) = \sum_{i=1}^N H(U_i | U_1 \dots U_{i-1} V^N) \leq \sum_{i=1}^N H(U_i | V_i)$$

Поделим обе части на  $N$  и к каждому слагаемому применим неравенство Фано. Получим

$$\frac{1}{N} H(U^N | V^N) \leq \frac{1}{N} \sum_{i=1}^N \eta(P_{ei}) + \frac{1}{N} \sum_{i=1}^N P_{ei} \log(M-1)$$

Поскольку энтропия - выпуклая  $\cap$  функция, средняя по  $i$  энтропия не меньше энтропии средней вероятности ошибки. Учитывая это, из последнего неравенства получаем доказываемое неравенство.

### 82. Обратная теорема кодирования.

**ТЕОРЕМА 5.4.** Обратная теорема кодирования. Для дискретного стационарного канала с информационной емкостью  $C_0$  для любого  $\delta > 0$  существует число  $\epsilon > 0$ , такое, что для любого кода со скоростью  $R > C_0 + \delta$  средняя вероятность ошибки удовлетворяет неравенству  $P_e \geq \epsilon$ .

**Доказательство.** Рассмотрим произвольный код  $C$  длиной  $n$  со скоростью  $R = \log |C|/n$ . Обозначим через  $N$  длину последовательностей сообщений  $u \in U^N$ , сопоставляемых словам кода  $C$ , через  $v \in V^N$  обозначим последовательности решений, принимаемых декодером. Имеет место цепочка соотношений

$$\begin{aligned} nR = \log |C| &= H(X^n) \leq H(U^N) = H(U^N) - H(U^n | V^N) + H(U^n | V^N) = I(U^n; V^N) + H(U^n | V^N) \leq \\ &\leq I(X^n; Y^n) + H(U^n | V^N) \leq nC_0 + n\gamma(\bar{P}_e) \end{aligned}$$

Из последнего неравенства получаем

$$\gamma(\bar{P}_e) \geq R - C_0 > \delta$$

Из свойств функции  $\gamma(\cdot)$  следует, что из неравенства  $\gamma(\bar{P}_e) > \delta$  следует существование положительного  $\epsilon > 0$  такого, что  $P_e \geq \epsilon$ . Теорема доказана.

### 83. Вычисление информационной емкости каналов без памяти (теорема с доказательством).

**ТЕОРЕМА 5.5.** Информационная емкость дискретного постоянного канала вычисляется по формуле

$$C_0 = \max_{\{p(x)\}} I(X; Y), \quad (5.26)$$

**Доказательство.** Доказательство состоит из двух шагов. Сначала докажем, что правая часть (5.26) является границей сверху на информационную емкость канала. Затем докажем, что эта граница достигается при некотором распределении  $p(x)$ .

При произвольном распределении  $\{p(x), x \in X^n\}$  запишем взаимную информацию между входными и выходными последовательностями в виде

$$I(X^n; Y^n) = H(Y^n) - H(Y^n | X^n), \quad (5.27)$$

Воспользовавшись (5.24) и свойствами логарифма и математического ожидания, выполним простые преобразования.

$$\begin{aligned} H(Y^n|X^n) &= \mathbf{M}[-\log p(y|x)] = \mathbf{M}\left[-\log \prod_{i=1}^n p(y_i|x_i)\right] = \\ &= \sum_{i=1}^n \mathbf{M}[-\log p(y_i|x_i)] = \sum_{i=1}^n H(Y_i|X_i) \end{aligned}$$

Из свойств энтропии

$$H(Y^n) \leq \sum_{i=1}^n H(Y_i), \quad (5.28)$$

причем равенство имеет место только для независимых ансамблей. С учетом выполненных выкладок вместо (5.27) получаем неравенство

$$I(X^n; Y^n) \leq \sum_{i=1}^n [H(Y_i) - H(Y_i|X_i)] = \sum_{i=1}^n I(X_i; Y_i), \quad (5.29)$$

Завершили первый шаг доказательства. Предположим теперь, что буквы на входе канала независимы, и докажем, что при этом предположении в (5.29) имеет место равенство. Это будет означать, что максимум в (5.25) следует искать только среди таких распределений на  $X^n$ , которые порождают последовательности независимых букв на выходе канала.

Доказательство сводится к доказательству того, что при этом предположении имеет место равенство в (5.28). То есть, 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

$$p(y) = \sum_{x \in X^n} p(x)p(y|x)$$

В предположении о независимости входных символов с учетом (5.24) получаем

$$\begin{aligned} p(y) &= \sum_{x \in X^n} \prod_{i=1}^n p(x_i) \prod_{i=1}^n p(y_i|x_i) = \sum_{x \in X^n} \prod_{i=1}^n p(x_i)p(y_i|x_i) = \\ &= \sum_{x_1 \in X^n} \sum_{x_2 \in X^n} \dots \sum_{x_n \in X^n} p(x_1)p(y_1|x_1) \cdot p(x_2)p(y_2|x_2) \cdot \dots \cdot p(x_n)p(y_n|x_n) \end{aligned}$$

Поочередно вынося сомножители за знаки сумм, приходим к равенству

$$p(y) = \prod_{i=1}^n \sum_{x_i \in X^n} p(x_i)p(y_i|x_i) = \prod_{i=1}^n p(y_i)$$

из которого и вытекает доказываемое утверждение о независимости выходных символов канала.

#### 84. Симметричные каналы. Свойства симметричных каналов.

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.

Сформулируем те свойства симметричных каналов:

- Свойство 5.4.1. Для симметричного по входу канала без памяти  $x \in X$

$$C_0 = \max_{\{p(x)\}} \{H(Y)\} - H(Y|x)$$

- Свойство 5.4.2. Для симметричного по входу канала без памяти  $x \in X$

$$C_0 \leq \log L - H(Y|x)$$

**Пример 5.3.1.** На рис. 5.7 показаны три диаграммы переходных вероятностей каналов и соответствующие матрицы переходных вероятностей.



Рис. 20: Симметричные каналы

- Свойство 5.4.3. Для симметричного по выходу канала без памяти при равновероятных входных символах выходные символы также равновероятны.
- Свойство 5.4.4. Для симметричного по входу канала без памяти  $x \in X$

$$C_0 = \log |Y| - H(Y|x)$$

## 85. Прямая теорема кодирования для дискретных постоянных каналов (теорема).

**ТЕОРЕМА 5.6.** Для дискретного постоянного канала с информационной емкостью  $C_0$  для любых  $\epsilon, \delta > 0$  существует достаточно большое число  $n_0$  такое, что для любого натурального числа  $n \geq n_0$  существует код длиной  $n$  со скоростью  $R \geq C_0 - \delta$ , средняя вероятность ошибки которого  $P_e < \epsilon$ .

Иными словами, при скорости кода сколь угодно близкой к информационной емкости  $C_0$  (но, конечно, меньшей  $C_0$  хотя бы на малую величину  $\delta$ ) увеличением длины кодовых слов можно добиться сколь угодно малой вероятности ошибки (меньше любого  $\epsilon$ ).

Оптимистический результат, обещаемый случайным кодированием, неконструктивен. Случайное кодирование - мощный метод анализа, придуманный Шенноном, - не может быть использовано для построения хороших кодов.

Для надежной передачи информации нужны довольно длинные коды. С ростом длины кодов экспоненциально растет число кодовых слов, а значит, и сложность описания кодов, сложность процедур кодирования и декодирования.

Поэтому используемые на практике коды имеют регулярную структуру, что делает возможной реализацию кодирования и декодирования с разумной сложностью.