Обзор методов неискажающего кодирования дискретных источников *

В. Н. Потапов

Аннотация

В обзоре рассмотрены основные задачи и конструкции теории неискажающего кодирования дискретных источников: побуквенное, адаптивное и универсальное кодирование, принцип кратчайшего описания, построение дерева контекстов и преобразование Барроуза–Уилера. Описаны наиболее известные методы сжатия данных: блочное, равномерное по выходу и арифметическое кодирование, схема кодирования Лемпел–Зива и методы интервального кодирования. Приведены оценки избыточности и трудоемкости перечисленных методов. Даны схемы доказательства для некоторых наиболее важных утверждений. Кроме того, рассмотрены задачи рандомизации сообщений и кодирования с синхронизацией, а также способы кодирования текстов на естественных языках и источников с низкой энтропией.

Введение

Теория кодирования дискретных источников исследует задачу сжатия сообщений без потери информации. Под сообщением подразумевается конечное или бесконечное слово в некотором алфавите, порожденное стационарным источником. Источник называется последовательностью однakoво распределенных случайных величин, множество значений которых — алфавит источника. Кодированием называется инъективное отображение множества слов алфавита источника в множество двоичных слов, а стоимостью кодирования — отношение средней длины кода сообщения к длине сообщения. В статье [100], положившей начало современной теории информации в целом и теории кодирования источников в частности, К. Шеннон показал, что нижней гранью стоимости кодирования является энтропия источника. Избыточность — разность между стоимостью и энтропией — является основным критерием качества кодирования. Другими важными характеристиками кодирования являются объем памяти, который требуется при программной реализации метода, и среднее время кодирования и декодирования, измеряемое в операциях над битами. Основной областью применения алгоритмов кодирования являются компьютерные программы сжатия данных (архиваторы). Методы кодирования могут быть использованы также в криптографии (см. [23, 30, 29, 62, 65]), задачах прогнозирования (см. [20, 79]) и поиска информации (см. [11]).

*) Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (код проекта 99-01-00531) и Федеральной целевой программы "Интеграция" (проект 1997 года № 473)
В обзоре сделана попытка проследить основные направления развития теории кодирования дискретных источников от задачи оптимального кодирования известного источника к задаче оптимального универсального кодирования семейства источников и задаче построения модели источника по сообщению; а также от побуквенного кодирования к арифметическому кодированию и схеме кодирования Лемпела–Зива. Кроме того, рассмотрены самосинхронизирующиеся и омофонные коды, а также способы кодирования двух часто встречающихся на практике видов источников: текстов на естественных языках и источников с низкой энтропией.

Основная часть, приведенная в обзоре сведений содержится в монографии Р. Е. Кривчевского [11], обзорных статьях А. Д. Винера с соавторами [117], Н. Мерхеу и М. Федера [79], А. Барона с соавторами [40], а также кандидатских диссертациях А. В. Кадача [5], Л. Н. Фоинова [30], М. П. Шаровой [35].

1. Источники сообщений

Пусть $A = \{a_1, \ldots, a_k\}$ — конечный алфавит и $x \in A^\infty$. Будем обозначать через $x_i^j$ подслово последовательности $x$, начиная с $i$-ой и заканчивая $j$-ой буквой, а через $x^n$ начало последовательности $x$ длины $n$, т.е. $x_i^n = x_i x_{i+1} \ldots x_j$ и $x^n = x_1 x_2 \ldots x_n$. Дискретным источником $X$ называется дискретный случайный процесс со значениями в $A$. Источник полностью задается вероятностями $Pr(X^n = x^n)$, которые определены для всех $x^n \in A^n$ и целых $n \geq 0$ и удовлетворяют равенствам $\sum_{i=1}^n Pr(X^{n+1} = x^n a_i) = Pr(X^n = x^n) \geq 0$ и $\sum_{x^n \in A^n} Pr(X^n = x^n) = 1$. Тогда $Pr(X_i = x_i) = \sum_{y_1 = x_i} Pr(X_i = y_i)$. Источник $X$ называется стационарным, если для всех целых $t > 0$ и $x_i^j \in A^{j-i+1}$ справедливы равенства $Pr(X_i^j = x_i^j) = Pr(X_{i+t}^{j+t} = x_i^j)$.

Введем обозначение $P(x^n) = Pr(X^n = x^n)$. Для каждого стационарного источника $X$ равенство

$$H(X) = \lim_{n \to \infty} \frac{1}{n} E(-\log P(x^n))$$

определяет неотрицательную величину $H(X)$ (см., например, [3, 11]), которая называется энтропией источника.

Основным видом стационарных дискретных источников, изучаемых в теории кодирования, являются марковские источники с конечным числом состояний. Пусть $S$ — множество состояний источника $X$ с алфавитом $A$. Тогда для каждой пары $\sigma \in S$ и $a \in A$ определена вероятность $P(a|\sigma) \geq 0$ порождения источником буквы $a$ в состоянии $\sigma$ и справедливы равенства $\sum_{a \in A} P(a|\sigma) = 1$. Кроме того, задана функция $\mu : S \times A \to S$, определяющая состояние $\sigma' = \mu(\sigma, a)$, в которое переходит источник после порождения буквы $a$ в состоянии $\sigma$. Ясно, что последовательность состояний $\sigma_0 \sigma_1 \sigma_2 \ldots$ марковского источника представляет собой марковскую цепь. В дальнейшем будем полагать, что эта цепь является неразложимой и непереходной. Для полного определения источника $X$ нужно задать начальное состояние $\sigma_0 \in S$. Тогда

$$Pr(X^n = x^n) = \prod_{i=0}^{n-1} P(x_{i+1} | \sigma_i),$$

где $\sigma_{i+1} = \mu(\sigma_i, x_{i+1})$. Таким образом, марковский источник $X$ задается алфавитом $A$, множеством состояний $S$, функцией $\mu$ и распределениями вероятностей $P(a_i|\sigma)$ в каждом состоянии, что будем кратко выражать равенством $X = \langle A, S, \mu, P \rangle$.

Обозначим через $q_{\sigma\tau}$ вероятности перехода источника $X = \langle A, S, \mu, P \rangle$ из состояния $\sigma$ в состояние $\tau$, т.е. $q_{\sigma\tau} = \sum_{i=\sigma(a_i)\sigma} P(a_i|\sigma)$. Тогда стационарные вероятности

$$q_{\sigma\tau} = \sum_{i=\sigma(a_i)\sigma} P(a_i|\sigma).$$

Тогда стационарные вероятности
$q_{\sigma}$ марковской цепи $\sigma_0\sigma_1\sigma_2\ldots$ можно получить из системы уравнений $q = qQ$ и $\sum_{\sigma \in \mathcal{S}} q_{\sigma} = 1$, где $Q$ — матрица переходных вероятностей $\{q_{\sigma\tau}\}$. Известно (см., например, [3]), что для марковского источника $X = \langle A, S, \mu, P \rangle$ справедливо равенство

$$H(X) = \sum_{\sigma \in \mathcal{S}} q_{\sigma}H_{\sigma},$$

где

$$H_{\sigma} = -\sum_{i=1}^{k} P(a_{i}|\sigma)\log P(a_{i}|\sigma).$$

Марковский источник $X = \langle A, S, \mu, P \rangle$ с единственным состоянием называется источником без памяти или источником Бернулли. Для него справедлива предложенная К. Шенноном [100] формула

$$H(X) = -\sum_{i=1}^{k} P(a_{i})\log P(a_{i}).$$

Стационарный источник называется марковским $r$-го порядка, если вероятность появления очередной буквы зависит только от $r$ предыдущих букв, т.е. множество состояний $S'$ источника можно отождествить с $A^r$.

Рис. 1.

Множество состояний марковского источника $r$-го порядка представим в виде листьев дерева. Пример такого дерева для двоичного алфавита и $r = 3$ изображен на рис. 1 (слева). Состояния $\sigma$ и $\sigma'$ будем называть эквивалентными, если $P(a_{i}|\sigma) = P(a_{i}|\sigma')$ для всех $a_{i} \in A$ и листья, соответствующие состояниям $\sigma$ и $\sigma'$, являются братьями. Объединяя каждую пару эквивалентных состояний в новое состояние, получим минимальное дерево состояний $S'$, не содержащее эквивалентных состояний. Пример преобразования приведен на рис. 1. Полученный источник-дерево $\langle A, S', \mu', P \rangle$ может оказаться не марковским как, например, на рис.1 (после порождения буквы $a_1$ в состоянии $\sigma = a_2$ следующее состояние неопределено). Однако каждый источник-дерево очевидно может быть дополнен до марковского обратной процедурой. Этот вид источников был введен Й. Риссаненом [90] и будет рассмотрен в разделе 10.

2. Основные определения теории кодирования источников

Пусть $E = \{0,1\}$, обозначим через $E^* = \bigcup_{n=1}^{\infty} E^n$ множество всех двоичных слов. Инъективное отображение $f : A \rightarrow E^*$ называется **побуквенным дешифруемым кодированием**, если произвольный конечный набор кодовых слов $f(a_1) \ldots f(a_n)$, записанных слитно, однозначно разделяется на кодовые слова. Известно несколько алгоритмов, позволяющих определить является ли отображение дешифруемым кодированием (см., например, [12, 48, 93]).

Множество $M \subset E^*$ двоичных слов называется **префиксным**, если никакое слово $v \in M$ не является префиксом (началом) другого слова $u \in M$. Инъективное отображение $f : A \rightarrow E^*$ называется **побуквенным префиксным кодированием**, если множество $f(A)$ кодовых слов — префиксное. Нетрудно заметить, что каждому префиксному множеству соответствует множество листовых вершин некоторого двоичного дерева. Непосредственно из этого можно заключить, что для произвольного побуквенного префиксного кодирования $f$ справедливо неравенство Крафта

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

где $l_i = |f(a_i)|$ — длина двоичного слова $f(a_i)$. Верно и обратное: для любого набора целых чисел $l_i > 0$, удовлетворяющего неравенству (4), найдется префиксное кодирование $f$ такое, что $l_i = |f(a_i)|$. Префиксный код называют **полным**, если в формуле (4) имеет место равенство. Ясно, что произвольное префиксное кодирование является дешифруемым, а обратное, вообще говоря, неверно. Однако, Б. Макмилланом [78] доказано неравенство (4) для произвольного побуквенного дешифруемого кодирования.

Блочное кодирование $f : A^n \rightarrow E^*$ можно свести к побуквенному, определив новый алфавит $B = A^n$. Вообще, **кодированием** называется инъективное отображение $f : A^n \rightarrow E^*$, где $A^* = \bigcup_{n=1}^{\infty} A^n$ — множеству слов алфавита $A$. Побуквенное дешифруемое кодирование $f : A \rightarrow E^*$ является частным случаем кодирования, поскольку допускает доопределение $f(x^n) = f(x_1) \ldots f(x_n)$. Кодирование $f : A^n \rightarrow E^*$ будем называть **префиксным**, если $f(A^n)$ — префиксное множество для всех целых $n > 0$.

**Стойкостью** $C(f,X)$ кодирования $f$ источника $X$ называется среднее число битов, которое нужно затратить для кодирования одной буквы исходного сообщения, т.е.

$$C(f,X) = \limsup_{n \to \infty} \frac{1}{n} L_n(f,X), \quad \text{где} \quad L_n(f,X) = E|f(X^n)|.$$

В частности, если $X$ — источник без памяти, а $f$ — побуквенное кодирование, то

$$C(f,X) = L_1(f,X) = \sum_{i=1}^{k} P(a_i)|f(a_i)|.$$
Избыточностью кодирования $f$ слова $x^n \in A^n$ на букву исходного сообщения называется величина $\frac{1}{n} \left( |f(x^n)| - \log \frac{1}{P(X^n)} \right)$. Средней избыточностью кодирования (на букву исходного сообщения) называется разность между стоимостью кодирования и энтропией источника (см. (1)), т.е.

$$R(f, X) = C(f, X) - H(X) = \lim_{n \to \infty} \sup R_n(f, X),$$

где $R_n(f, X) = \frac{1}{n} E\left(|f(X^n)| - \log \frac{1}{P(X^n)}\right)$.

Аргументы функций $C, R, L_n, R_n$ будем в дальнейшем опускать в тех случаях, когда они явны из контекста.

Если $f$ — префиксное кодирование, то из неравенства (4) и выпуклости вверх функции $\log t$ следует неравенство $R_n(f, X) \geq 0$. Для произвольного кодирования $f$ Леун-Ян-Ченгом и Т. Ковером [72] доказано неравенство

$$R_n(f, X) \geq -\frac{\log n + \log \log k + c}{n},$$

где $c > 0$ — константа, независящая от $k$ и $n$. Таким образом средняя избыточность $R$ произвольного кодирования неотрицательна.

3. Побуквенное и блочное кодирование

Сначала рассмотрим побуквенное префиксное кодирование источников без памяти. Задача построения для произвольного источника без памяти оптимального, т.е. имеющего наименьшую стоимость, побуквенного префиксного кодирования была решена Д. Хаффменом [64]. Он предложил следующий индуктивный по размеру алфавита алгоритм построения оптимального кодирования. Сначала упорядочиваем буквы алфавита по убыванию их вероятностей. Затем объединяем две наименьшие вероятные буквы $a_{k-1}$ и $a_k$ в одну новую букву $a'$, вероятность которой определяем как сумму вероятностей букв $a_{k-1}$ и $a_k$. Если код Хаффмена для нового алфавита, содержащего на одну букву меньше, известен, то полагаем коды неизмененных букв прежними, а коды букв $a_{k-1}$ и $a_k$ определяем как коды буквы $a'$ с добавлением в конце 0 или 1 соответственно.

Широко известны побуквенные префиксные коды Шеннона-Фано и Шеннона [100]. Первый из них строится по индукции. Сначала упорядочиваем буквы алфавита по убыванию их вероятностей. Затем разделяем алфавит на две части, имеющие наиболее близкие вероятности, не меняя порядка букв. Запишем 0 в качестве первого символа кода для всех букв первой половины алфавита и 1 в качестве первого символа кода для всех букв второй половины. Каждую из двух половин алфавита делим на две части с возможно близкими вероятностями и приписываем к коду первых частей 0, к коду вторых частей — 1. Процесс деления продолжаем пока в каждой из частей не останется лишь по одной букве.

Алгоритм построения кода Шеннона позволяет получить код произвольной буквы независимо от других с помощью только арифметических операций. Пусть буквы упорядочены по убыванию их вероятностей. Определим числа $\delta_1 = 0$ и $\delta_{i+1} = \delta_i + P(a_i)$. В качестве кодового слова $f(a_i)$ возьмем первые после запятой $[-\log P(a_i)]$ двоичных знаков числа $\delta_i$. Очевидно, что избыточность кодирования $R$ каждой буквы не превышает 1 и из формул (3) и (6) получается оценка средней избыточности кодирования Шеннона источника без памяти

$$R = C - H = \sum_{i=1}^{k} P(a_i)([-\log P(a_i)] + \log P(a_i)) \leq 1.$$
Можно заметить, что избыточность кодирования Шеннона не меньше избыточности кодирования Шеннона–Фано, которая в свою очередь не меньше избыточности оптимального кодирования Хаффмена (в среднем). Достаточно рассмотреть источники с двухбуквенным алфавитом и вероятностями букв 0, 1 и 1/2, 1/2, чтобы убедиться, что верхняя и нижняя оценка избыточности 1 и 0 соответственно достигаются для всех трех рассмотренных кодов.

Однако можно получить более точные оценки избыточности кодирования, если известны вероятности букв. Нетривиальные нижние оценки избыточности побуквенного префиксного кодирования были получены Р. Е. Кричевским и Г. Л. Ходаком [7, 11]. Различные верхние оценки избыточности кодирования Хаффмена были получены в работах Р. Галлагера [58], Р. Капоселли и А. Ди Санти [46], Д. Манстетина [77]. Последний, в частности, доказал неравенство

\[ R(f_0, X) \leq \log \left( \frac{2 \log e}{e} \right) + \frac{2 \log e}{e}, \]

где \( f_0 \) — кодирование Хаффмена, \( p \) — максимальная вероятность букв, порожденной источником без памяти \( X \) и \( e \) — постоянная Эйлера.

Объем памяти \( V(f) \), требуемый для кодирования и декодирования побуквенного префиксного кода, почти линейно зависит от длины алфавита, а время кодирования \( T(f) \) на букву исходного сообщения прямо пропорционально стоимости кодирования.

Побуквенное префиксное кодирование можно использовать для кодирования блоков из \( n \) букв исходного алфавита. Пользуясь формулами (7) и (8), можно доказать теорему кодирования Шеннона [100] (см. также [3, 11]) для нетеряющего информационного кодирования:
1) избыточность кодирования неотрицательна,
2) для произвольного стационарного источника и \( \varepsilon > 0 \) найдется блочное префиксное кодирование, избыточность которого не превышает \( \varepsilon \).

Практическое применение блочного кодирования ограничивает быстрый рост объема используемой памяти при увеличении длины блока. Для кодирования \( f_n \) блоками длины \( n \) объем памяти \( V(f_n) = O(k^n) \) растет экспоненциально, в то время как избыточность убывает не быстрее чем линейно \( R_n(f_n, X) \geq \frac{1}{n} \) при \( n \to \infty \), где \( c > 0 \) — некоторая константа (см. [7]).

Марковский источник с множеством состояний \( S \) как и источник Бернулли можно кодировать побуквенным кодом. Для этого нужно разделить сообщение на \( |S| \) подпоследовательностей, состоящих из букв, порожденных в каком-то одном состоянии, и затем кодировать каждую подпоследовательность собственным префиксным кодом, учитывая вероятности букв. В этом случае средняя избыточность кодирования с использованием любого из трех описанных кодов не превышает 1 (это следует из формул (2) и (8)), а объем требуемой памяти линейно зависит от \( k|S| \).

Обычно задача построения префиксного побуквенного кода ставится для источника с конечным алфавитом, но проблема кодирования сообщений источника с счетным алфавитом также возникнет весьма часто. Обычная двоичная запись натуральных чисел не является префиксным кодом. Простейшим способом ее преобразования в префиксный код является добавление перед двоичной записью числа блока из нулей равного длине двоичной записи. Естественно этот двоичный код увеличивает длину двоичной записи вдвое. Эффективный полный префиксный код натурального ряда предложил В. И. Левенштейн [13]. Длина кодового слова Левенштейна числа
\[ l(n) = \log n + \log \log n(1 + o(1)) \]

при \( n \to \infty \). Впоследствии подобные коды были предложены П. Эйлесом [54], К. Стоутом [103], а также С. Эвеном и М. Роде [56]. Из работы Р. Альсведе с соавторами [38] следует, что для длины произвольного префикского кода натурального ряда выполнено (при \( n \to \infty \)) неравенство

\[ l(n) \geq \log n + \log \log n - \varepsilon, \]

где \( \varepsilon > 0 \) — константа, независящая от кода.

4. Омофонное кодирование и кодирование с синхронизацией

Для решения некоторых специальных задач на метод кодирования помимо минимизации избыточности и сложности вычислений налагаются дополнительные требования. В частности, с целью защиты информации от незаконного доступа рассматривается задача рандомизации сообщений: необходимо закодировать сообщение так, чтобы символы кодовой последовательности 0 и 1 были независимы и равновероятны.

В первом приближении эта задача решается с помощью произвольного кодирования с низкой избыточностью, поскольку вероятности символов 0 и 1 в кодовой последовательности неизбежно стремятся к 1/2 при стремлении к нулю избыточности кодирования. Однако, существуют специальные методы решения задачи рандомизации сообщений. К. Гюнтер [62] предложил использовать для этой цели омофонное кодирование. В отличие от обычного при омофонном кодировании одному сообщению могут соответствовать различные кодовые слова. В этом случае дешифрируемость уже не является необходимым условием правильного декодирования омофонного кода, если в различных разбиениях сообщения на кодовые слова на одинаковых местах находятся омофоны — коды одной и той же буквы. Свойства такого кодирования рассмотрены, например, А. Вебером и Т. Хедом [108]. Для решения задачи рандомизации сообщений можно ограничиться префиксным омофонным кодированием. Например, для рандомизации источника Бернулли с вероятностями букв \( P(a_1) = 3/4, P(a_2) = 1/4 \) можно использовать омофонное кодирование \( f \): \( Pr(f(a_1) = 0) = 2/3, Pr(f(a_1) = 10) = 1/3, Pr(f(a_2) = 11) = 1 \). Кодовые слова 0 и 01 — омофоны.

Для выбора омофона требуется решить задачу генерации случайной величины с заданным распределением. Поэтому важной характеристикой омофонного кодирования является число случайных бит, используемых при кодировании одного символа. В работе Дж. Мэйси с соавторами [65] было предложено и исследовано оптимальное (т.е. имеющее минимальную избыточность) омофонное префиксное побудительное кодирование. Установлены следующие оценки его эффективности: \( R < 2, \eta < 4 \), где \( \eta \) — число случайных бит на символ сообщения. Задача построения омофонного кодирования источника без памяти с произвольно малыми избыточностью и числом случайных бит решена Б. Я. Рэйко и А. Н. Финовым [23, 29, 30] на основе арифметического кодирования (см. раздел 6).

Разработка методов помехоустойчивого кодирования является одной из центральных задач теории информации. Она обычно решается за счет введения дополнительной избыточности с помощью кодов, исправляющих ошибки. Однако задачи...
локализации ошибок можно решать и без введения проверочных символов, например, с помощью кодирования с синхронизацией.

Слово \( v \in E^* \) называется синхронизатором кода \( f(A) \), если для суффикса \( u \) произвольного кодового слова конкатенация \( uv \) является кодовой последовательностью, т.е. \( uv \in f(A^*) \). Если \( v \) — синхронизатор кода \( f(A) \), то после любой ошибки декодирование будет выполняться неправильно до тех пор, пока встретится синхронизатор. Например, для кода \( f(A) = \{1, 01, 00\} \) синхронизатором является слово 1. Если в сообщении 00,1,01,1,01 произошла ошибка в первом разряде, то сообщение 1,01,01,1,01 будет декодироваться правильно начиная с синхронизатора, т.е. после третьей буквы. Необходимым условиями существования синхронизатора префиксного кода являются, во-первых, выполнение равенства Крафта (4) для набора длин кодовых слов \( l_1, \ldots, l_k \), а во-вторых, равенство \( \text{НОД}(l_1, \ldots, l_k) = 1 \).

М. Шмотценберг [99] (см. также [12, 14]) показал, что эти два условия являются достаточными для существования префиксного кода с синхронизатором, у которого длины кодовых слов равны \( l_1, \ldots, l_k \). Т. Фергюсон и Дж. Рабинович [57] предложили рассматривать коды содержащие синхронизатор как кодовое слово. Такие коды называют самосинхронизирующимися. В [57] предложены достаточные условия существования у источника Бернулли самосинхронизирующегося кода Хаффмена. Б. Монтгомери и Дж. Абрахам [82], а также Р. Капелли с соавторами [47] предложили алгоритмы построения близких к оптимальным самосинхронизирующихся кодов. Синхронизатор кода позволяет локализовать ошибку, но не всегда дает возможность определить количество неправильно декодированных букв, т.е. после декодирования последовательность букв может оказаться свинутой. Последнего недостатка лишенны сильно самосинхронизирующиеся коды, предложенные В. М. Ламом и С. Кулкари [76].

Не менее важным является исследование средней задержки синхронизации. Говорят, что синхронизация имеет задержку \( \tau \), если после того как в позиции \( t \) произошла ошибка, только начиная с позиции \( \tau + t \) последовательность декодируется правильно. М. Титченер [104] предложил метод построения кодов с небольшой средней задержкой синхронизации.

5. Равномерное по выходу кодирование

Равномерное по выходу кодирование также позволяет локализовать ошибки без введения дополнительной избыточности. Ошибка в одном кодовом слове не может повлиять на декодирование других кодовых слов, если все кодовые слова имеют одинаковую длину. Ясно, что побуквенное или блочное равномерное по выходу кодирование не может сжимать данные. Основная идея равномерного по выходу кодирования состоит в том, что разные по длине, но близкие по вероятности слова алфавита источника кодируются жесткими возможными блоками из нулей и единиц одинаковой длины. Чтобы любую последовательность букв входного алфавита можно было закодировать, слова в \( k \)-буквенном алфавите \( A \), которым сопоставляются кодовые слова, должны соответствовать листвам некоторого полного \( k \)-ичного дерева. Например, дерево на рис.1 (справа) порождает равномерный по выходу код: \( f(a_1, a_1) = 00, f(a_1, a_2, a_1) = 01, f(a_1, a_2, a_2) = 10, f(a_2) = 11 \).

Оптимальным равномерным по выходу кодированием называется кодирование, имеющее минимальную среднюю избыточность среди кодов с определенной длиной кодовых слов. Способ построения оптимального равномерного кодирования для источников Бернулли был независимо предложен Г. Л. Ходаком [32] и Б. И. Танталом.
Дерево, соответствующее оптимальному равномерному по выходу кодированию с длиной $m$ кодовых слов, можно построить по индукции, начиная с дерева, состоящего только из корня. Пусть имеется некоторое дерево $\Delta \subseteq \mathcal{A}^*$. Выберем лист $x \in \Delta$ с наибольшей вероятностью и добавим к дереву всех сыновей. Эту процедуру будем повторять пока число листьев дерева не превышает $2^m$. Листья $x \in \mathcal{A}^*$ получившегося дерева будем кодировать различными двоичными словами длины $m$.

Ч. Чокенс и Ф. Виллемс [105] показали, что существует равномерное по выходу кодирование произвольного марковского источника со сколь угодно малой избыточностью. С. Савари и Р. Галлагер [97] показали, что кодирование Танстэла—Ходака является асимптотически (при $m \to \infty$) близким к оптимальному для марковских источников, причем его средняя избыточность убывает как $c/m$, где константа $c > 0$ зависит только от источника и точно определена в работе [97].

6. Арифметическое кодирование

История разработки и исследования одного из наиболее популярных и эффективных методов сжатия данных — арифметического кодирования — восходит к классическому коду Шеннона, Э.Н. Гильберт и Э. Ф. Мур [60] предложили блочный код, подобный коду Шеннона, но не требующий предварительного упорядочивания блоков из $n$ букв по вероятностям. А именно, пусть $X$ — стационарный источник в алфавите $A$. Все наборы из $n$ букв алфавита $A$ упорядочим лексикографически, т. е. $a_{i_1}, a_{i_2}, \ldots, a_{i_n} < a_{j_1}, a_{j_2}, \ldots, a_{j_n}$, если $i_m = j_m$ при $m < l$ и $i_l < j_l$. Для каждого слова $x \in \mathcal{A}^n$ определим величину $Q(x) = \sum_{y<x,y \in \mathcal{A}^n} P(y)$. В качестве кода $f(x^n)$ рассмотрим $[-\log P(x^n)] + 1$ двоичных знаков после запятой числа $Q(x^n) + P(x^n)/2$. Полученное кодирование является префиксным и из формулы (7) следует, что $R_n \leq 2/n$.

Процесс кодирования можно представить как разделение отрезка $[0,1]$ на пересекающиеся полунепереды $[Q(x^n), Q(x^n) + P(x^n))$. Кодовое слово $f(x^n)$ оказывается числителем двоично рационального числа со знаменателем $2^{[-\log P(x^n)] + 1}$, попавшем в полунеперед $[Q(x^n), Q(x^n) + P(x^n))$. П. Элайосом (см. [36, 112]) была предложена процедура последовательного вычисления границ полунепереды $[Q(x^n), Q(x^n) + P(x^n))$ по мере поступления букв блока $x^n$. Она естественно вытекает из того, что соответствующий код $x \in \mathcal{A}^n$ полунеперед по определению разделяется на полунепереды, соответствующие словам $xa_1, xa_2, \ldots, xa_k$. Однако в таком виде арифметическое кодирование невозможно использовать на практике при больших $n$, поскольку требуемая точность вычислений быстро возрастает при увеличении длины блока. Й. Риссенен [86, 87] предложил алгоритм округленного вычисления границ полунепереды, использующий арифметические операции с числами ограниченной длины. Этот алгоритм позволяет кодировать слова $x^n \in \mathcal{A}^n$ с любой наперед заданной избыточностью при $n \to \infty$. Кроме того, в методе Й. Риссенен кодовое слово $f(x^n)$ можно вычислять и передавать поразрядно по мере поступления букв слова $x^n$. Это усовершенствование оказалось решающим на пути практического использования арифметического кодирования.

Существует несколько версий арифметического кодирования Й. Риссенена, наиболее известная из них разработана И. Уайтенем с соавторами [114]. Оценки эффективности этого алгоритма для источников без памяти содержатся в работе Б.Я. Рябко и А. Н. Финова [24]:

$R_n \leq \frac{k(t + \log e)}{2^n} + \frac{2}{n}, V = O(t), T = O(t \log t \log \log t)$,
где $k$ — объем алфавита, $\tau$ — количество битов в представлении вероятностей букв и $t > \tau + 2$ — точность арифметических операций в битах.

Отличные от метода Ы. Рассмотрена варианты арифметического кодирования предложены Ю. М. Штаркзом [36] и Б. Я. Рябко [21, 94].

7. Универсальное кодирование

Чтобы декодировать сообщение нужно знать не только кодовую последовательность, но и функцию, с помощью которой сообщение закодировано, или метод кодирования должен быть универсальным, т. е. пригодным для эффективного кодирования всех источников из некоторого множества.

Рассмотрим некоторое параметрическое семейство $M$ источников, где источник с параметром $\theta \in \Theta$ определяется своим распределением $P_\theta(x)$ вероятностей порождения сообщений $x \in A^\theta$. Глазное (в смысле зависимости $P_\theta$ от $\theta$) параметрическое семейство источников будем называть моделью. Целое число $d > 0$ называется размерностью модели, если область $\Theta \subset R^d$ изменения параметра имеет ненулевой объем. Примером модели может служить множество $M_k$ источников без памяти в $k$-буквенном алфавите $A$. В этом случае областью изменения параметра является $(k - 1)$-мерный симплекс со стороной 1 (параметр — набор вероятностей букв) и

$$P_\theta(x^n) = \theta_1^{r_1} \cdots \theta_{k-1}^{r_{k-1}} (1 - \theta_1 - \cdots - \theta_{r-1})^r,$$

где $r_i$ — число вхождений буквы $a_i$ в слово $x^n$.

Если распределение $P_\theta(x)$ источника известно, то задачу минимизации избыточности решает кодирование с длинами $|f(x^n)|$ кодовых слов наиболее близкими к $- \log P_\theta(x^n)$. С другой стороны, для произвольного префикского кодирования $f$, вследствии неравенства Крафта (4), найдется распределение $Q(x^n)$ такое, что $|f(x^n)| \geq - \log Q(x^n)$ для всех $x^n \in A^n$. Из этого следует, что задачу кодирования неизвестного источника удобно разделить на две части: во-первых, выбор распределения $Q(x^n)$ и, во-вторых, построение кодирования с длинами кодовых слов близкими к $- \log Q(x^n)$. Вторая задача была рассмотрена в предыдущих параграфах. Соответственно разделим избыточность кодирования на две части: избыточность модели

$$D_n(P_\theta||Q) = E_{P_\theta} \log \frac{P_\theta(X^n)}{Q(X^n)},$$

(11)

заявившую от выбора целевого распределения $Q$, и избыточность метода кодирования

$$r_n(f, Q) = E_{P_\theta} (|f(X^n)| + \log Q(X^n)).$$

Тогда из (7) имеем

$$R_n = \frac{1}{n} (D_n(P_\theta||Q) + r_n(f, Q)).$$

Из выпуклости вверх функции $\log t$ следует, что избыточность модели неотрицательна, т. е.

$$D_n(P_\theta||Q) \geq 0,$$

(12)

Теорема Шеннона утверждает, что избыточность метода кодирования $\frac{r_n(f, Q)}{n}$ на букву исходного сообщения можно сделать сколь угодно малой. В частности, это достигается с помощью арифметического кодирования.
Задача универсального кодирования, впервые поставленная Б.П. Фитингофом [31], состоит в нахождении распределения \( Q^0(x^n) \), минимизирующего максимальную по всем \( \theta \in \Theta \) избыточность модели

\[
\inf_Q \sup_{\theta \in \Theta} D_n(P_\theta||Q) = \sup_{\theta \in \Theta} D_n(P_\theta||Q^0).
\]

(13)

Рассмотрим произвольную вероятностную меру \( \omega(\theta) \) на множестве \( \Theta \). Поскольку \( \omega(\theta) \geq 0 \) и \( \int_\Theta d\omega(\theta) = 1 \), справедливо равенство

\[
\sup_{\theta \in \Theta} D_n(P_\theta||Q) = \sup_\omega \int_\Theta D_n(P_\theta||Q) d\omega(\theta).
\]

(14)

Известно (см. [11]), что

\[
\inf_Q \sup_\omega \int_\Theta D_n(P_\theta||Q) d\omega(\theta) = \inf_Q \sup_\omega \int_\Theta D_n(P_\theta||Q) d\omega(\theta),
\]

принем \( \sup_\omega \inf_Q \) и \( \sup_\omega \sup_Q \) достигаются одновременно на некоторых распределениях \( Q^0 \) и мере \( \omega_0 \). Из (14) следует, что распределение \( Q^0 \), решающее задачу (15), минимизирует избыточность модели в смысле задачи (13).

Н. Мерхев и М. Федер [80] показали, что распределение \( Q^0(x^n) \) содержится в параметрическом семействе \( M \), если область \( \Theta \) изменения параметра — выпуклая.

Пусть \( \omega(\theta) \) — произвольная вероятностная мера и распределение \( Q^\omega(x^n) \) определено равенством

\[
Q^\omega(x^n) = \int_\Theta P_\theta(x^n) d\omega(\theta).
\]

Из определения (11) величины \( D_n(P_\theta||Q) \) следуют равенства

\[
D_n(P_\theta||Q) = D_n(P_\theta||Q^\omega) + E_{P_\theta} \log \frac{Q^\omega(X^n)}{Q(X^n)},
\]

\[
\int_\Theta E_{P_\theta} \log \frac{Q^\omega(X^n)}{Q(X^n)} d\omega(\theta) = D_n(Q^\omega||Q).
\]

Тогда

\[
\int_\Theta D_n(P_\theta||Q) d\omega(\theta) = \int_\Theta D_n(P_\theta||Q^\omega) d\omega(\theta) + D_n(Q^\omega||Q).
\]

(16)

Величину \( D_n(P_\theta||Q^\omega) \) можно рассматривать как среднюю (относительно меры \( \omega \)) избыточность модели. Асимптоматическое поведение \( D_n(P_\theta||Q^\omega) \) при \( n \to \infty \) было исследовано Б. Кларком и А. Барроном [49]. Поскольку интеграл \( \int_\Theta D_n(P_\theta||Q^\omega) d\omega(\theta) \) не зависит от распределения \( Q \), а величина \( D_n(Q^\omega||Q) \) неотрицательна (12) и равна 0 при \( Q = Q^\omega \), то из (16) получаем

\[
\inf_Q \int_\Theta D_n(P_\theta||Q) d\omega(\theta) = \int_\Theta D_n(P_\theta||Q^\omega) d\omega(\theta) = I(\Theta, A^n),
\]

(17)

где последнее равенство является определением взаимной информации \( I(\Theta, A^n) \) между областью \( \Theta \) изменения параметра с вероятностной мерой \( \omega(\theta) \) и множеством \( A^n \) слов с условными вероятностями, заданными равенством \( Pr(X^n = x^n|\theta) = P_\theta(x^n) \).

Величина \( \sup_\omega I(\Theta, A^n) \) называется пропускной способностью канала связи. Известно (см., например, [3]), что супремум величины \( I(\Theta, A^n) \) достигается на такой мере
$$\omega_0(\theta),$$ что избыточность модели \( D_n(P_\theta||Q^{\omega_0}) \) не зависит от \( \theta \in \Theta \). Тогда из (14), (15) и (17) получаем соотношения, которые известны как теорема о равенстве избыточности и пропускной способности:

$$\inf_Q \sup_{\theta \in \Theta} D_n(P_\theta||Q) = \sup_{\omega} \int \omega \, d\omega(\theta) = D_n(P_\theta||Q^{\omega_0}).$$

Эта теорема впервые была опубликована Б. Я. Рябко [16] и затем Л. Дэвиссоном и А. Леон-Гарсией [52]. Таким образом, распределение \( Q^{\omega_0} = Q^0 \) минимизирует избыточность модели.

Введем обозначение \( \alpha_n(M) = D_n(P_\theta||Q^0) \), где распределение \( Q^0 \) минимизирует избыточность модели \( M \). Для множества \( M_k \) источников без памяти в \( k \)-буквенном алфавите получены следующие результаты. Сначала Б. П. Фитингоф [31] показал, что \( \alpha_n(M_k)/n \to 0 \) при \( n \to \infty \). Затем Р. Е. Кричевский [8, 9, 10] и независимо Л. Дэвиссон с соавторами [53] показали, что \( \alpha_n(M_k) = \frac{k-1}{2} \log (1 + o(1)) \).

Существенным шагом в исследовании задачи универсального кодирования явились работы Р. Е. Кричевского и В. К. Трофимова [67], в которой предложена вероятностная мера \( J'(\theta) \) на \( (k - 1) \)-мерном симплексе со стороной 1:

$$dJ'(\theta) = \frac{\Gamma(k/2)}{\pi^{k/2}} \theta_1^{-1/2} \ldots \theta_{k-1}^{-1/2} (1 - \theta_1 - \ldots - \theta_{k-1})^{-1/2} d\theta,$$

gде \( \Gamma \) — гамма-функция Эйлера. Распределение

$$Q^{J'}(x^n) = \frac{\Gamma(k/2) \prod_{i=1}^{k-1} \Gamma(r_i + 1/2)}{\pi^{k/2} \Gamma(n + k/2)}.$$  
(18)

gде \( r_i \) — число вхождений буквы \( a_i \) в слово \( x^n \), является приближением к оптимальному на \( M_k \) распределению \( Q^0 \) в том смысле, что

$$\lim_{n \to \infty} \sup_{\theta \in \Theta} \frac{D_n(P_\theta||Q^{J'})}{D_n(P_\theta||Q^0)} = 1.$$  
(19)

Источники, имеющие распределение со свойством (19), называют универсальными (см. [109]). Более того, Б. Кларк и А. Баррон [49] доказали, что для любой внутренней точки \( \theta \in \Theta \) при \( n \to \infty \) верно соотношение

$$D_n(P_\theta||Q^{J'}) - D_n(P_\theta||Q^0) \to 0.$$  
(20)

Впоследствии с помощью распределения \( Q^{J'} \) К. Ксай и А. Баррон [119] установили, что

$$\alpha_n(M_k) = \frac{k - 1}{2} \log \frac{n}{2\pi e} + \log \frac{\pi^{k/2}}{\Gamma(k/2)} + o(1).$$  
(21)

Для множества \( M'_k \) марковских источников порядка \( r \) В. К. Трофимов [28] показал, что

$$\alpha_n(M'_k) = (1/2) r^k (k - 1)(1 + o(1)) \log n.$$  
(22)

И. Риссанен обобщил этот результат на произвольную модель \( M \) размерности \( d \), удовлетворяющую некоторым дополнительным условиям гладкости распределения \( P_\theta(x^n) \) как функции от \( \theta \). В работе [89] он показал, что

$$\alpha_n(M) = \frac{d}{2} \log n + O(1).$$  
(23)

Кроме того, в работе [90] Й. Риссанен доказал теорему о строгой оптимальности распределения $Q^0$. А именно, если $d$ — размерность модели, то для произвольного распределения $Q$ и почти всех значений параметра $\theta$ выполнено неравенство

$$
\limsup_{n \to \infty} \frac{D_n(P_\theta || Q)}{\log n} \geq d/2.
$$

Обобщение последнего результата содержится в работе Н. Мерхева и М. Федера [80].

8. Распределение наибольшего правдоподобия

Задачу нахождения распределения $Q(x)$ минимизирующего избыточность кодирования неизвестного источника из параметрического семейства $M$ можно ставить и по другому. Естественно предполагать, что сообщение $x^n$ было порождено источником с параметром $\hat{\theta}(x^n) \in \Theta$ таким, что $P_{\hat{\theta}(x^n)}(x^n) = \max_{\theta \in \Theta} P_\theta(x^n)$. Рассмотрим множество $M_k$ источников без памяти, т.е. область $\Theta$ изменения параметра является $(k - 1)$-мерным симпликом со стороной 1. Величина $F(x^n) = -\log P_{\hat{\theta}(x^n)}(x^n)$ называется эмпирической энтропией слова $x^n$. Очевидно, что

$$
F(x^n) = \sum_{i=1}^{k} \frac{r_i}{n} \log \frac{n}{r_i},
$$

где $r_i$ — число вхождений буквы $a_i$ в слово $x^n$. Известно (см. [49]), что

$$
E_{P_\theta} \log \frac{P_{\hat{\theta}(x^n)}(X^n)}{P_\theta(X^n)} = \frac{k - 1}{2} \log e + o(1). \quad (24)
$$

Последнее равенство не противоречит (12), поскольку $\sum_{x^n \in A^n} P_{\hat{\theta}(x^n)}(x^n) > 1$.

Для кодирования неизвестного сообщения Ю. М. Штарьков [37] предложил использовать распределение $Q^*(x^n)$ наибольшего правдоподобия (максимальной вероятности), определенное равенством

$$
\max_{x^n \in A^n} \log \frac{P_{\hat{\theta}(x^n)}(x^n)}{Q^*(x^n)} = \min_{Q} \max_{x^n \in A^n} \log \frac{P_{\hat{\theta}(x^n)}(x^n)}{Q(x^n)}.
$$

Пусть $K_n(M) = \sum_{x^n \in A^n} P_{\hat{\theta}(x^n)}(x^n)$. Покажем, что

$$
Q^*(x^n) = \frac{P_{\hat{\theta}(x^n)}(x^n)}{K_n(M)}. \quad (25)
$$

Действительно

$$
\log \frac{P_{\hat{\theta}(x^n)}(x^n)}{Q(x^n)} = \log \frac{P_{\hat{\theta}(x^n)}(x^n)}{Q^*(x^n)} + \log \frac{Q^*(x^n)}{Q(x^n)}.
$$

Поскольку $Q^*(x^n)$ и $Q(x^n)$ — распределения, то найдется такое слово $x^n \in A^n$, что $Q^*(x^n) \geq Q(x^n)$. Тогда $\max_{x^n \in A^n} \log \frac{Q^*(x^n)}{Q(x^n)} \geq 0$ и равенство нулю достигается только при $Q^*(x^n) = Q(x^n)$, т.е. равенство (25) доказано. Величина $-\log Q^*(x^n)$ названа Й. Риссаненом [90] стохастической сложностью слова $x^n$ в соответствующей модели $M$. 

13
Пусть \( \beta_n(M) = \log K_n(M) \). По определению параметра \( \hat{\theta}(x^n) \) имеем неравенство
\[ P_{\hat{\theta}}(x^n) \leq P_{\hat{\theta}(x^n)}(x^n), \]
Тогда
\[ D_n(P_{\theta}||Q^*) \leq E_{P_\theta} \log \frac{P_{\hat{\theta}(X^n)}(X^n)}{Q^*(X^n)} = \beta_n(M), \] (26)
т.е. \( \alpha_n(M) \leq \beta_n(M) \). Ю. М. Штарквов [37] получил асимптотику для величины \( \beta_n(M) \) для множеств марковских источников конечного порядка и источников без памяти. В частности, им показано, что \( \beta_n(M_k) = \frac{k-1}{2} \log n + o(1) \) при \( n \to \infty \).
Й. Риссен [91] получил асимптотику величины \( \beta_n(M) \) для гладких параметрических семейств \( M \) размерности \( d \):
\[ \beta_n(M) = \frac{d}{2} \log n + c(M) + o(1), \] (27)
где \( c(M) \) — точно определенная константа. В частности,
\[ \beta_n(M_k) = \frac{k-1}{2} \log n + \log \frac{\pi^{k/2}}{\Gamma(k/2)} + o(1). \] (28)
Из соотношений (23), (26) и (27) следует, что
\[ \lim_{n \to \infty} \frac{\sup_{\theta \in \Theta} D_n(P_\theta||Q^*)}{D_n(P_\theta||Q^0)} = 1. \]
Таким образом, распределение \( Q^* \) определяет универсальный источник. Покажем, что для источников без памяти распределения \( Q^* \) и \( Q^{\omega'} \) асимптотически совпадают в среднем. Справедливы равенства
\[ E_{P_\theta}(\log Q^*(X^n) - \log Q^{\omega'}(X^n)) = E_{P_\theta} \log \left( \frac{P_{\hat{\theta}}(X^n)}{Q^{\omega'}(X^n)} \right) \]
\[ = E_{P_\theta} \log \left( \frac{P_{\hat{\theta}(X^n)}(X^n)}{Q^{\omega'}(X^n)} \frac{Q^*(X^n)}{P_{\hat{\theta}(X^n)}(X^n)} \right) \]
\[ = D_n(P_\theta||Q^{\omega'}) + E_{P_\theta} \log \frac{P_{\hat{\theta}(X^n)}}{P_{\hat{\theta}(X^n)}} - \beta_n(M_k). \]
Отсюда и из соотношений (20), (21), (24), (28) для произвольной внутренней точки \( \theta \) симплекса \( \Theta \) получаем
\[ E_{P_\theta}(\log Q^*(X^n) - \log Q^{\omega'}(X^n)) = \alpha_n(M_k) - \beta_n(M_k) + \frac{k-1}{2} \log \epsilon + o(1) = o(1). \]

9. **Кодирование на основе статистики сообщений**

На практике статистические характеристики источников, порождающих сообщения, чаще всего неизвестны. Сведения о свойствах источника как правило извлекают из самого сообщения и затем их явно или неявно используют в процессе кодирования. Кодирование очередной буквы сообщения происходит на основе статистики некоторой части сообщения, которая называется окном. Окно, соответствующее \( i \)-й букве сообщения, будем обозначать через \( w_i \). В качестве окна можно использовать все сообщение: \( w_i = x \). Такое кодирование называется двухпроходным: на первом
проходе определяется статистика сообщения, на втором — сообщение кодируется на основе этой статистики. В этом случае задержка передачи сообщения равняется длине всего сообщения и помимо закодированного сообщения необходимо дополнительно передавать кодер (функцию, осуществляющую кодирование) или статистику, на основе которой он был построен. Необходимость в дополнительной информации отсутствует, если использовать универсальное кодирование. Однако непосредственное применение оптимального универсального кодирования для сжатия достаточно больших сообщений невозможно, поскольку трудоемкость построения префиксного кодирования с длинами кодовых слов, равными \( \log Q^0(x^i) \), экспоненциальном растет как функция от длины сообщения. Метод универсального кодирования с полиномиальной трудоемкостью был предложен В. Ф. Бабкиным и Ю. М. Штарковым [1, 101].

Избыточность этого кодирования вдвое больше избыточности оптимального кодирования. В частности, \( R_n = \frac{(k-1)\log n}{n}(1 + o(1)) \) для источников без памяти. Аналогичные методы кодирования были предложены также Т. Линчем [74], Л. Дэвиссоном [51] и Т. Ковером [50]. Эти универсальные методы кодирования можно также рассматривать как разновидность блочного кодирования: достаточно разделить сообщение на части одинаковой длины и кодировать их независимо. Из работ В. К. Трофимова [27] и Дж. Лоуренса [71] известно универсальное равномерное по выходу кодирование с полиномиальной трудоемкостью, избыточность которого в конечное число раз больше избыточности оптимального универсального кодирования.

Использование удлиняющегося окна \( w_i = x^{i-1} \) также позволяет обойтись без передачи дополнительной информации. В этом случае при кодировании очередной буквы используются все предыдущие. В частности, при кодировании источников без памяти \((i+1)\)-ую букву сообщения кодируют, основываясь на распределении вероятностей

\[
P(a_j) = \frac{r_j(w_i) + \varepsilon}{|w_i| + k\varepsilon},
\]

где \( r_j(w_i) \) — число вхождений буквы \( a_j \) в окне \( w_i \) длины \( |w_i| \). Величина \( \varepsilon, \varepsilon \geq 0 \), называется сдвигом. Положительный сдвиг позволяет избежать неопределенности при кодировании первых вхождений букв в сообщение. Из формулы (18) следует, что

\[
Q^{w'}(a_j|w_i) = \frac{Q^{w'}(x^{i-1}a_j)}{Q^{w'}(x^{i-1})} = \frac{r_j(w_i) + 1/2}{(i-1) + k/2},
\]

t.е. асимптотически оптимальным является сдвиг \( \varepsilon = 1/2 \).

Д. Кнут [66] предложил конструкцию динамического кода Хаффмена. Каждая очередь буква \( x_i \) кодируется кодом Хаффмена, построенным по статистике окна \( w_i = x^{i-1} \), а затем строится новый код Хаффмена, соответствующий окну \( w_{i+1} \). Динамические коды Хаффмена были впоследствии исследованы Р. Галлгером [58] и Дж. Витгером [107]. Последний получил оценку \( R \leq 2 \) избыточности кодирования источников без памяти для специального класса этих кодов. Трудоемкость динамических кодов Хаффмена невелика — не более конечного числа операций над \( \log n \)-значными числами на букву сообщения. Ниже будет рассмотрена схема кодирования Лемпела–Зива, стандартное описание которой также использует удлиняющееся окно.

Таким образом, алгоритмы сжатия данных обычно используют скользящее окно \( w_i = x_{i-1}^{i-1} \) постоянной длины \( m \), которое сдвигается на одну букву вправо при кодировании очередной буквы (такое кодирование часто называют адаптивным).
Широкое применение скользящего окна обусловлено тем, что, во-первых, ресурсы оперативной памяти кодирующих устройств ограничены и, во-вторых, реальные источники почти никогда не являются стационарными. Скользящее окно требует фиксированного объема памяти и непрерывно позволяет следить за изменяющимися статистическими свойствами сообщения. Р. Е. Кричевский [68] показал, что при использовании скользящего окна асимптотически оптимальным будет сдвиг \( \varepsilon = 0, 50922 \ldots \) (см. формулу (29)).

Особенно эффективным при кодировании источников без памяти является использование скользящего окна и арифметического кодирования (вероятность очередной буквы определяется по формуле (29)). Б. Я. Рябько и А. Н. Финостов [24] показали, что надлежащий выбор точности арифметических операций и длины окна позволяет достигнуть избыточности \( R \leq (1 + \frac{1}{\log m}) \frac{1}{m} \) при \( V = O(m) \) и \( T = O(\log m \log \log m) \), где \( m \) — длина скользящего окна.

Чтобы сократить число операций можно сдвигать окно не после поступления каждой очередной буквы, а через определенное число букв. Можно и совсем зафиксировать окно. Фиксированное окно позволяет экономить память, поскольку вместо окна можно хранить только частоты букв в окне. Конструкция, объединяющая достоинства фиксированного и скользящего окна, — "мнимое" скользящее окно — предложена Б. Я. Рябько [22]. Отличие "мнимого" от реального скользящего окна состоит в том, что на каждом шаге из окна удаляется не последняя, а случайная буква. Это дает возможность хранить не само окно, а только частоты букв. Случайная буква может выбираться как с помощью источника случайных величин, так и на основе уже закодированной части сообщения, поскольку код сообщения хорошо аппроксимирует источник Бернулли с вероятностями \( P(0) = P(1) = 1/2 \). В [22] для сообщений, порожденных источником без памяти, доказано, что

\[
\lim_{i \to \infty} Pr(v_1(w_i) = n_1, \ldots, v_k(w_i) = n_k) = Pr(r_1(w_i) = n_1, \ldots, r_k(w_i) = n_k),
\]

где \( r_j(w_i) \) — число вхождений буквы \( a_j \) в реальное скользящее окно \( w_i = x_{i-1}^{i-1} \), а \( v_j(w_i) \) — число вхождений буквы \( a_j \) в "мнимое" окно.

10. Выбор модели источника и принцип кратчайшего описания

В предыдущих разделах мы исследовали кодирование сообщений, порожденных известным источником или источником, принадлежащим некоторому параметрическому семейству. Теперь мы рассмотрим более сложную, но более реальную задачу выбора параметрического семейства, в рамках которого следует рассматривать сообщение, чтобы получить наиболее короткий код сообщения. Предположим, что имеется счетное семейство источников \( M_r = \{P_\theta(x) : \theta \in \Theta \subset R^{d_r}\} \). Будем считать, что множества \( M_r \) вложены друг в друга \( M_1 \subset M_2 \subset \ldots \subset M_r \subset \ldots \) и соответственным \( d_1 < d_2 < \ldots < d_r < \ldots \). Примером такой последовательности могут служить, например, множества марковских источников конечного порядка. Задача определения наименьшего семейства \( M_r \), содержащего нужный источник, имеет смысл и если множество источников конечно, поскольку избыточность оптимального универсального кодирования линейно зависит от размерности параметра \( d_r \) модели \( M_r \) (см. формулу (23)).

Пусть имеется некоторый префиксный код натурального ряда, закодируем этим кодом индексы моделей. Пусть \( l(r) \) — длина кода индекса модели \( M_r \). Код сообщения можно составить из двух частей: кода индекса модели \( M_r \) и универсального...
на модели $M_r$ кода сообщения. Предложенный Й. Риссаненом [88] принцип кратчайшего описания состоит в выборе индекса $r$, минимизирующего длину $L(x)$ кода сообщения $x$, т. е.

$$L(x) = \min_r (l(r) - \log Q_r^0(x)),$$

где $Q_r^0$ — оптимальное универсальное распределение на $M_r$. Последствия принятия кратчайшего описания были применены Б. Я. Рябко [19] при построении дважды универсального кодирования, которое является асимптотически оптимальным для всех классов марковских источников конечного порядка. Пусть $r$ — порядок марковского источника, существует (см. (9)) префиксный код натурального ряда с длинами кодовых слов

$$l(r) = \log r + O(\log \log r).$$

Выбирая $r$ по принципу кратчайшего описания, получим дважды универсальное кодирование с длинами кодовых слов как в (30), меньшими чем $l(i) - \log Q_i^0(x)$, где $i$ — истинный порядок марковского источника. Очевидно, что для избыточности (модели) дважды универсального кодирования справедливы неравенства

$$\alpha_n(M_i) \leq D_n \leq \alpha_n(M_i) + l(i).$$

Из формул (23) и (31) видно, что избыточность дважды универсального кодирования асимптотически эквивалентна избыточности оптимального кодирования источников из $M_r$. Аналогичный результат получается, если в качестве кода для слова $x$ выбирать слово длины $-\log Q(x)$, где

$$Q(x) = \sum_{k=1}^{\infty} 2^{-l(r)} Q_r^0(x).$$

Действительно

$$-\log Q(x) \leq -\log \left( \max_r (2^{-l(r)} Q_r^0(x)) \right) = \min_r (l(r) - \log Q_r^0(x)).$$

Л. Дэвисон [51] предложил называть кодирование $f$ сильно универсальным для источников из $M$, если $\sup_{X \in M} R_n(f, X) \to 0$ при $n \to \infty$, и слабо универсальным, если для произвольного источника $X \in M$ верно, что $R_n(f, X) \to 0$ при $n \to \infty$. Из формул (23) и (32) следует, что дважды универсальное кодирование является только слабо универсальным для всех марковских источников конечного порядка, в то время как оптимальное кодирование для марковских источников фиксированного порядка сильно универсально (см. раздел 7). Из (23) следует, что сильно универсальное кодирование для всех марковских источников конечного порядка невозможно, так как размерность модели неограничена.

Наиболее простое кодирование, использующее принцип кратчайшего описания, предложил Р. Ньюхорд и Р. Шилдс [84]. Разделим сообщения $x^n$ на блоки длины $m$. Составим словарь $U_m$ из всех встречающихся в сообщении блоков. Каждый блок в сообщении $x^n$ будем кодировать его номером в словаре. Естественно код сообщения помимо кодов блоков должен содержать словарь. Длина кода $L_m(x^n)$ сообщения $x^n$ удовлетворяет равенству

$$L_m(x^n) = |U_m| \cdot m \cdot \log k + \frac{n}{m} \log |U_m|,$$
поскольку для записи слова длины $m$ в $k$-буквенном алфавите достаточно $m \lfloor \log k \rfloor$ битов. Выберем длину блока $m(n)$, минимизирующую величину $I_m(x^n)$. Как показано в [84] это кодирование с длиной блока $m(n)$ является слабо универсальным для марковских источников.

Размерность параметра $d_r$ семейства марковских источников порядка $r$ растет экспоненциально в зависимости от $r$, а именно $d_r = k^r(k - 1)$. Следовательно избыточность оптимального кодирования быстро растет в зависимости от порядка источника. Экспоненционально от порядка источника растет и объем памяти, требуемый для реализации метода (если кодовые слова строить с помощью явно использующих статистику методов кодирования).

Выбор в качестве множества моделей семейства источников-деревьев (см. раздел 1) позволяет находить более адекватную, а значит и более эффективную модель для кодирования каждого сообщения. Марковский источник $r$-го порядка может иметь много эквивалентных состояний, сокращение которых позволяет значительно уменьшить размерность модели. Ф. Виллемс, Ю. М. Штарьков и Ч. Чокенс в работе [112] предложили алгоритм построения взвешенного контекстного дерева, который в классе источников-деревьев позволяет найти близкое к оптимальному распределение $Q(x)$ такое, что

$$D_n(P_\theta || Q) = \frac{1}{2} \log n + O(1)$$

при $n \to \infty$, где $|\theta|$ — размерность модели источника-дерева, которой принадлежит источник с распределением $P_\theta$, равная произведению числа листьев минимального дерева на размер алфавита без единицы.

Предлагаемый в [112] метод состоит в выборе некоторой вероятностной меры на множестве деревьев ограниченной высоты и определении затем целевого распределения согласно равенству (33). Рассмотрим множество $U$ всех полных двоичных (будем рассматривать двухбуквенный алфавит) деревьев глубины не более $d > 0$. Пусть $T'$ — множество листьев дерева $T$, а $x(s)$ — слово, состоящее из букв двоичного слова $x$, непосредственно следующих за вхождениями подслова $s$, $|s| \leq d$ в слове $x$. Определим

$$P_T(x^n) = \prod_{s \in T'} Q^{\omega'}(x(s)),$$

где распределение $Q^{\omega'}$ задано равенством (18) и $T \in U$. Рассмотрим произвольное распределение $P_W(T)$ вероятностей на множестве $U$, т.е. $P_W(T) \geq 0$ и $\sum_{T \in D} P_W(T) = 1$. Пусть

$$Q_W(x^n) = \sum_{T \in U} P_W(T) P_T(x^n).$$

Тогда для произвольного источника-дерева с деревом $T$ и распределением $P_\theta(x)$ справедливы неравенства

$$\log \frac{P_\theta(x^n)}{Q_W(x^n)} \leq \log \frac{P_\theta(x^n)}{P_W(T) P_T(x^n)} \leq \log \frac{1}{P_W(T)}$$

$$+ \sum_{s \in T'} \log \frac{P_\theta(x^n(s))}{Q^{\omega'}(x^n(s))} \leq \log \frac{1}{P_W(T)} + |T'| \sum_{s \in T'} \frac{\log |x^n(s)| + 2}{2|T'|}$$

$$\leq \log \frac{1}{P_W(T)} + \frac{|T'|}{2} \log \frac{n}{|T'|} + c.$$
где \( c > 0 \) — некоторая константа, \( |\theta| = |T'| \) — размерность модели. Здесь первое неравенство следует из (36), второе — из (35), третье — из неравенства ( \( P_\theta \) — распределение произвольного источника Бернулли)

\[
\log \frac{P_\theta(x^n)}{Q_\omega(x^n)} \leq \frac{1}{2} \log n + 1,
\]

dоказанного в \[37\] (см. также формулу (21)), четвертое неравенство следует из выпуклости верх функции \( \log t \). Поскольку \( U \) — конечное множество, то даже использование равномерного распределения \( P_W(T) = 1/|U| \) позволяет получить распределение \( P_W(x^n) \), избыточность которого отличается от избыточности оптимального в классе источников-деревьев распределения не более чем на некоторую константу. Ф. Виллекс с соавторами в работе [112] предложили распределение \( P_W(U) \), минимизирующую эту константу асимптотически, и эффективное даже при малых \( n \): 

\[
\log \frac{P_W(x^n)}{P_W(x^n)} \leq n + 2|\theta| - 1.
\]

Кроме того, они предложили рекуррентную процедуру для получения этого распределения:

\[
P_W(x(s)) = Q_\omega(x^n(s)), \text{ если } |s| = d;
\]

\[
P_W(x(s)) = \frac{1}{2}(Q_\omega(x^n(s)) + P_W(x(0s))P_W(x(1s))) \text{ в остальных случаях.}
\]

Распределение \( P_W(x) = P_W(x(\emptyset)) \) — искомое.

Ф. Виллекс с соавторами в работе [113] показали, что аналогичный метод эффективен и для других, более богатых по сравнению с источниками-деревьями множеств моделей источников и даже, как показал Ф. Виллекс [111], для множеств неограниченных по высоте источников-деревьев.

Предложенный Й. Риссаненом [89] алгоритм "Контекст" дает возможность (см. [90, 109]) найти распределение \( Q(x) \), близкое к оптимальному в классе источников-деревьев в смысле формулы (34) путем непосредственного построения источника-дерева по сообщению. Опишем быстрый вариант [92] алгоритма "Контекст" для двухбуквенного алфавита. Сначала построим рекуррентное некоторое множество \( B(x^n) \) подслов слова \( x^n \) и множество контекстов \( C(x^n) \), состоящее из начал этих подслов без последней буквы. Обозначим через \( \delta(0|s) \) и \( \delta(1|s) \) — индексы контекста \( s \), принимающие значения 0 и 1.

Положим \( B(\emptyset) = C(\emptyset) = \{\emptyset\} \) и \( \delta(0|\emptyset) = \delta(1|\emptyset) = 1 \). Пусть множество \( B(x^n-1) \) построено. Найдем наибольшее целое число \( i \geq 0 \) такое, что подслово \( x_{n-i} \) содержится в \( B(x^n-1) \). Тогда \( B(x^n) = B(x^n-1) \cup \{x_{n-i}\} \). Если контекст \( s = x_{n-i} \ldots x_{n-1} \) не содержится в \( C(x^n-1) \), то определим \( \delta(x_n|s) = 0 \) и \( \delta(\lambda|s) = 1 \), где \( \lambda \neq x_n \). Если \( s \in C(x^n-1) \), то изменем индекс: \( \delta(x_n|s) = 0 \). Затем определим \( C(x^n) = C(x^n-1) \cup \{s\} \) и перейдем к следующей букве \( x_{n+1} \). Можно показать, что множество \( B(x^n) \) содержит все суффиксы и префиксы своих элементов. Например, пусть

\[
x = 000011001100.
\]

Тогда \( B(x) = \{\emptyset, 0, 00, 000, 0000, 0, 1, 11, 10, 100, 01, 011, 110, 1100\} \). Соответствующее множество контекстов \( C(x) = \{\emptyset, 0, 00, 000, 0, 10, 11, 110, 11, 100\} \) можно представить в виде дерева. Оно изображено на рис. 2. Контекст \( s \) задает путь к соответствующей ему вершине 0 — налево, 1 — направо, причем контекст читается справа налево. Справа от каждой вершины, соответствующей контексту \( s \), указан итоговый индекс \( \delta(0|s) \) и слева — \( \delta(1|s) \).
К каждому листу дерева $T_1$, соответствующему множеству контекстов $C(x)$, добавим по два листа и затем добавим еще недостающие листья так, чтобы дерево стало полным. Каждому листу $\tau$ полученного дерева $T_2$ припишем вес $c(0|\tau) = c(1|\tau) = 1$. Каждой внутренней вершине $s \in T_2$ припишем вес

$$c(\lambda|s) = c(\lambda|0s) + c(\lambda|1s) - \delta(\lambda|s),$$

где $\lambda = 0, 1$. Вес $c(\lambda|s)$ отличается от количества символов $\lambda$, встречающихся в контексте $s$ в слое $x$ не более чем на 1. Дерево $T_2$, соответствующее слову (37), изображено на рис. 3, слева от вершины $s$ указан вес $c(0|s)$, справа — $c(1|s)$.

Сократим некоторые узлы дерева $T_2$ так, чтобы слово $x$ в полученной модели источника-дерева имело наименьшую стохастическую сложность. Пусть $c(s) = c(0|s) + c(1|s)$. Тогда в соответствии с формулой (28) стохастическая сложность $L(s)$
(длина кода, удовлетворяющего критерию наибольшего правдоподобия) слова \(x(s)\) равняется
\[
L(s) = c(0|s) \log \frac{c(s)}{c(0|s)} + c(1|s) \log \frac{c(s)}{c(1|s)} + \frac{1}{2} \log \frac{c(s)}{2}.
\]
Оптимальный набор состояний-контекстов будем выбирать рекуррентно, начиная с листьев дерева \(T_2\). Определим \(J(s) = L(s)\) для всех листов, а для внутренних вершин дерева положим
\[
J(s) = \min\{L(s), J(0s) + J(1s)\}.
\]
Если первый элемент меньше или равен второму, то из дерева \(T_2\) удалим потомков вершины \(s\). Полученное дерево \(T\) — искомое. Распределение \(Q(x^n) = \prod_{i=1}^n Q(x_i|x_i^{-1})\), индуцированное деревом \(T\), определяется в соответствии с формулой (29):
\[
Q(x_i|x_i^{-1}) = Q(x_i|s) = \frac{c(x_i|s) + 1/2}{c(s) + 1},
\]
где \(s\) — суффикс \(x_i^{-1}\), являющийся листом дерева \(T\). Распределение \(O(x^n)\) является близким к оптимальному в классе источников-деревьев в смысле формулы (33).

Как показано Й. Риссененом [92] рациональная организация выполнения алгоритма "Контекст" требует \(O(n \log \log n)\) арифметических операций с числами длины \(O(\log n)\) при кодировании слова длины \(n\).

Арифметическое кодирование позволяет строить эффективные коды слов, которые реализуют распределения, построенные как алгоритмом "Контекст", так и с помощью взведенного контекстного дерева.

11. Преобразование Барроуда–Уилера

Другим практически полезным методом кодирования последовательности, основанным на использовании контекстов, является преобразование Барроуда–Уилера [45]. Пусть \(x^n\) — некоторое слово в двоичном алфавите, превратим его в цикл, определив \(x_0 = x_n, x_{n-1} = x_{n-1}, \ldots, x_i = x_{i-1}, \ldots\). Каждой букве \(x_i\) поставим в соответствие контекст \(s(x_i)\) длины \(n\): \(s(x_i) = x_{i-1}x_{i-2} \ldots x_{i-n}\). Упорядочим контексты лексикографически (читая контекст справа — налево). Преобразованием Барроуда–Уилера \(BW(x^n)\) называется слово длины \(n\), составленное из букв слова \(x^n\) в порядке их контекстов, т.е. \(BW(x^n) = x_{i_1}x_{i_2} \ldots x_{i_n}\), где \(s(x_{i_1}) \leq s(x_{i_2}) \leq \ldots \leq s(x_{i_n})\). Другими словами, рассмотрим матрицу вращений слова \(x^n\) и расставим строки матрицы в лексикографическом (начиная с конца) порядке. Первым столбец получившейся матрицы и есть слово \(BW(x^n)\). Например, пусть \(x = 110110100\). Тогда матрица вращений слова \(x\) имеет вид
\[
\begin{array}{cccccccccc}
1&1&0&1&1&0&1&0&0&0 \\
1&0&1&1&0&1&0&0&1 &1 \\
0&1&1&0&1&0&0&1 &1 \\
1&1&0&1&0&0&1 &1 &0 \\
1&0&1&0&0&0 &1 &1 &0 &1 \\
0&1&0&0&1 &1 &0 &1 &1 &1 \\
1&0&0&1 &1 &0 &1 &1 &0 &1 \\
0&0&1 &1 &0 &1 &1 &0 &1 &1 \\
0&1&1 &0 &1 &1 &0 &1 &0 &0 \\
\end{array}
\]
После упорядочивания получаем матрицу

\[
\begin{pmatrix}
1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\end{pmatrix}
\]

Тогда \( BW(11011010) = 101111000 \). Оказывается, что по слову \( BW(x) \) можно восстановить всю упорядоченную матрицу. Во всех столбцах (как и в строках) матрицы содержится одинаковое число 0 и 1. Последний столбец вследствии лексикографического порядка строк имеет вид 00...011...1, где число нулей и единиц известно. Поскольку строки зашифрованы парой, состоящей из последней и первой буквы каждой строки упорядоченной матрицы, содержится в слове \( x \). Упорядочив множество пар, мы получим предпоследний столбец упорядоченной матрицы. Аналогичным образом получаем тройки, четверки и т.д., восстанавливая всю матрицу. Теперь чтобы найти исходное слово \( x \) нам нужно лишь номер строки упорядоченной матрицы совпадающий с \( x \).

Если рассмотреть слово \( x \), порожденное некоторым источником-деревом, то его преобразование \( BW(x) \) будет состоять из последовательности \( x(s_1)x(s_2)...x(s_t) \), где \( x(s_i) \) — подслово, порожденное в состоянии \( s_i \), причем состояния \( s_i \) лексикографически упорядочены.

Методы кодирования преобразования Барргоуза-Уилера \( BW(x) \) двоичного слова \( x \), а также эффективные алгоритмы выполнения преобразования рассмотрены А. В. Кадачем [5]. Выполнение преобразования Барргоуза-Уилера слова длины \( n \) требует \( O(n \log^2 n) \) операций над битами, а обратное преобразование требует \( O(n \log^2 n) \) операций над битами в среднем по сообщениям, порожденным стационарным источником с энтропией \( H \). Эмпирические исследования [3] показывают, что слова \( BW(x) \) можно эффективно кодировать интервальными методами, которые будут рассмотрены ниже.

12. Схемы кодирования Лемпела-Зива

Помимо описанных в предыдущих параграфах методов универсального кодирования, которые в явном виде используют контексты и статистику сообщений, существуют алгоритмические методы сжатия данных, использующие статистические характеристики источников неявно.

Пусть \( A = \{a_1, \ldots, a_k\} \) — некоторый алфавит и \( x \in A^n \). Схема кодирования, предложенная А. Лемпелем и Я. Зивом в 1977 году [122] (в дальнейшем именуемая LZ77), состоит в разделении кодируемого слова \( x^n \in A^n \) на подслова \( \sigma_i, \ i = 1 \ldots m \), по следующему правилу. Пусть начало слова \( x^n \) уже разделено на подслова, т.е. представляет собой конкатенацию подслов \( \sigma_1 \sigma_2 \ldots \sigma_i \) и \( x^n = \sigma_1 \ldots \sigma_i x_{n_i}^n \). Выберем следующее подслово \( \sigma_{i+1} = x_{i+1}^{r_i} \) как наиболее длинное начало остатка \( x_{n_i}^n \), которое уже встречалось в \( x_{1}^{r_i-1} \), т.е.

\[
\sigma_{i+1} = x_{i+1}^{r_i} = x_{n_i+r_i-i}^{n_i},
\]
где \( n_i < l_i \). Кодом каждого подслова \( \sigma_{i+1} \) будет пара чисел \((n_i, r_i - l_i)\). Например, слово \((a_1a_2)a_2a_1a_2a_1a_1a_2a_1a_2a_1\) разделяется на подслова \( a_2, a_1a_2, a_1, a_1a_2a_1, a_2a_1a_2a_1\) и кодируется последовательностью пар чисел \((2, 1), (1, 2), (1, 1), (4, 3), (3, 3)\). Первое число в каждой паре целесообразно записывать в двоичном виде с использованием \([\log l_i] + 1\) битов, второе можно кодировать произвольным префиксным кодом чисел натурального ряда.

В схеме кодирования LZ77 можно использовать фиксированное или скользящее окно длины \( w \), что чаще всего и делается на практике (см., например, [5]). Тогда подслово \( \sigma_{i+1} \) выбирается как наиболее длинное подходящее начало остатка сообщения, которое содержит в окне. В этом случае для записи первого числа в каждой паре достаточно \([\log w]\) битов. Второе число в паре можно записывать с помощью произвольного префиксного кода натурального ряда, например, кода Левенштейна [13]. Из формуллы (9) следует, что длина кода \( L(\sigma_i) \) подслова \( \sigma_i \) удовлетворяет неравенству

\[
L(\sigma_i) \leq \log w + \log |\sigma_i| + 2 \log \log |\sigma_i| + c,
\]

где \( c > 0 \) — некоторая константа и \( |\sigma_i| = r_i - l_i + 1 \) — длина подслова \( \sigma_i \).

Известна модификация схемы LZ77, предложенная П. Бендером и Д. Вольфом [43]. Она заключается в том, что после нахождения длиннейшего подходящего подслова \( \sigma_{i+1} \) в окне \( w_i \) нужно найти подходящее подслово \( \sigma'_i + 1 \subseteq w_i \) — наиболее длинное, не считая \( \sigma_{i+1} \). Вместо числа \( |\sigma_{i+1}\| \) нужно кодировать число \( |\sigma_{i+1}\| - |\sigma'_i + 1| \). При декодировании, зная начало подслова \( \sigma_{i+1} \), нетрудно найти в окне \( w_i \) наиболее длинное подслово \( \sigma'_i + 1 \), совпадающее с началом подслова \( \sigma_i + 1 \). Тогда длина подслова \( \sigma_i + 1 \) легко восстанавливается из равенства

\[
|\sigma_{i+1}| = |\sigma'_i + 1| + (|\sigma_{i+1}| - |\sigma'_i + 1|).
\]

Схема кодирования, предложенная А. Лемпелем и Я. Зивом в 1978 году [123] (в дальнейшем именуемая LZ78) отличается от описанной выше тем, что на каждом шаге выбирается наиболее длинное начало остатка \( x^m_1 \), которое совпадает с некоторым уже выделенным подслова \( \sigma_j, j < i \), и к нему добавляется еще одна буква, т.е. \( \sigma_{i+1} = \sigma_ja_p \). Кодом подслова \( \sigma_{i+1} \) будет пара чисел \((j, p_i)\). Например, слово \( a_2a_1a_2a_1a_2a_1a_2a_1 \) разделяется на подслова \( a_2, a_1a_2a_1, a_2a_1a_2a_1 \) и кодируется последовательностью пар чисел \((0, 2), (0, 1), (1, 1), (2, 2), (4, 1)\).

Схему LZ78 удобно рассматривать как кодирование с динамическим словарем \( U \). Сначала словарь \( U \) состоит из всех букв алфавита источника. На каждом шаге алгоритма отляем от остатка кодируемого слова наиболее длинное слово \( y \in U \) и добавляем в словарь \( U \) все слова вида \( ya_i \). Словарь \( U \) может произвольным образом нумеровать и удалять из него слова, все продолжения которых уже имеются в словаре.

Наиболее известная модификация этой схемы кодирования предложена Т. А. Велчем [110]. Она отличается от схемы LZ78 тем, что на каждом шаге в словарь добавляется только одно слово \( ya_i \), где \( a_i \) следующая за словом \( y \) буква в кодируемом слове \( x \).

Существуют еще несколько вариантов схем Лемпела–Зива (самые известные [102, 120]), которые оптимизируют алгоритм поиска и выделения подслова, способы записи натуральных чисел, кодирование первого вхождения буквы в слово и т.д. В частности, проблема оптимизации разделения слова на подслова рассмотрена

23
А. В. Кадачем [5], а задача эффективного кодирования длин выделенных подслог исследована Е. И. Ситняковской [26]. Подробный обзор вариантов схем Лемпела–Зива сделал Т. Белл с соавторами [42].

Непосредственно из алгоритмов LZ77 и LZ78 следует, что выполнение декодирования значительно менее трудоемко, чем выполнение кодирования. Известны версии схем кодирования LZ77 и LZ78 (см. [5]), трудоемкость декодирования которых линейна относительно длины n сообщения, а трудоемкость кодирования есть величина $O(n \log n)$.

Уже в 1978 г. А. Лемпел и Я. Зив [123] доказали, что схемы кодирования LZ77 и LZ78 являются слабо универсальными на множестве всех марковских источников с конечными алфавитом и множеством состояний. В дальнейшем оценки избыточности неоднократно уточнялись. Е. Плотник с соавторами [85] показали, что при $n \to \infty$

$$R_n(f^{78}, X) = O\left(\frac{\log \log n}{\log n}\right).$$

Здесь и далее $f^{78}$ — кодирование, построенное по схеме LZ78 с удлиняющимся окном (основной вариант), $X$ — произвольный марковский источник. Затем оценка избыточности была улучшена С. Савари [95]:

$$R_n(f^{78}, X) \leq O\left(\frac{1}{\log n}\right).$$

Г. Лоучард и В. Знаковский [73] установили, что эта оценка неулучшаема для источников Бернулли и нашли в этом случае асимптотику для $R_n(f^{78}, X)$ с точностью до эквивалентности. Для кодирования $f^{77}_w$, использующего схему LZ77 со скользящим окном длины $w$, Х. Морита и К. Кобаяши [83] показали что при $w \to \infty$

$$R(f^{77}_w, X) = O\left(\frac{\log \log w}{\log w}\right).$$

Затем для варианта схемы LZ77 с фиксированным окном А. Дж. Винер [118] для марковских источников конечного порядка установили, что при $w \to \infty$

$$R(f^{77}_w, X) = H(X) \frac{\log \log w}{\log w}(1 + o(1)).$$

Для схемы LZ77 с удлиняющимся окном С. Савари [96] получил следующую оценку избыточности:

$$R_n(f^{77}_w, X) \leq \frac{2H(X) \log \log n}{\log n}(1 + o(1)).$$

Кодирование $f^{91}_w$, использующее модификацию П. Бендера и Дж. Вольфа с фиксированным окном, оказались асимптотически эффективнее основного алгоритма. А. Д. Винер и А. Дж. Винер [116] показали, что

$$R_n(f^{91}_w, X) = O\left(\frac{1}{\log w}\right).$$

Избыточность модификации Т. А. Велча асимптотически совпадает с избыточностью основной схемы LZ78 (см. [95]).
Основные идеи, использованные при установлении оценок избыточности схем кодирования Лемпела–Зива, можно проследить на примере схемы LZ77 с фиксированным окном длины $w$. Пусть $X$ — некоторый марковский источник. В данном случае удобно рассмотреть последовательность $X_n$ бесконечную в обе стороны. Обозначим через $N_i(X)$ наименьшее натуральное число $N$ такое, что $X_i^l = X_{-N+l}^{N-1}$. Ясно, что случайная величина $N_i(X)$ в среднем должна быть близка к $\frac{1}{P(X_i)}$ при $l \to \infty$. Поскольку

$$H(X) = \lim_{l \to \infty} \frac{1}{l} E \log \frac{1}{P(X_i)},$$

можно предположить, что $\frac{\log N_i(X)}{l} \approx H(X)$. Действительно А.Д. Винер и Я. Зив [115] доказали, что по вероятности

$$H(X) = \lim_{l \to \infty} \frac{\log N_i(X)}{l}.$$  \hspace{1cm} (39)

Рассмотрим подслово $\sigma_1$, которое является длиннейшим началом последовательности $X_i^\infty$, содержащееся в $X_{2w+1}^\infty$. Из совпадения событий $\{N_i > w\} = \{|\sigma_1| < l\}$ и равенства (39) можно заключить (см. [115]), что по вероятности

$$H(X) = \lim_{w \to \infty} \frac{\log w}{|\sigma_1(X)|}.$$  \hspace{1cm} (40)

Из выпуклости вверх функции $\log t$ и неравенства (40) получаем неравенства

$$E \log |\sigma_1(X)| \leq \log(E|\sigma_1(X)|) \leq \log \log w + c,$$  \hspace{1cm} (41)

где $c > 0$ — константа.

Стоимостью кодирования называется отношение длины слова к длине кода. Поэтому справедливо подробно обсуждаемое в [118] равенство

$$C(f_w^{77}, X) = \lim_{t \to \infty} \frac{E L(\sigma_1(X))}{E|\sigma_1(X)|} = E \frac{L(\sigma_1(X))}{E|\sigma_1(X)|},$$  \hspace{1cm} (42)

gде $L(\sigma_1)$ — длина кода подслова $\sigma_1$, а второе равенство следует из стационарности источника $X$. Тогда из соотношений (38)–(41) при $w \to \infty$ получаем верхнюю оценку избыточности

$$R(f_w^{77}, X) = C(f_w^{77}, X) - H(X) \leq \frac{\log w + E(\log |\sigma_1(X)| + 2 \log \log |\sigma_1(X)| + c)}{E|\sigma_1(X)|} - H(X)$$

$$= \frac{H(X) \log \log w}{\log w}(1 + o(1)).$$

13. Интервальное кодирование

Еще одним алгоритмическим методом сжатия данных является интервальное кодирование, отличающееся быстродействием и простотой реализации. Интервальное кодирование состоит в следующем: в исходной последовательности каждая буква
заменился на число, равное количеству букв до предыдущего вхождения той же буквы. Например слово

\[(a_1a_2a_3)a_3a_3a_3a_2a_2a_2a_1a_1a_3\]  \hspace{1cm} (43)

будет преобразовано в последовательность чисел \(...0001008006\). Известны две модификации этого метода, позволяющие уменьшить стоимость кодирования. Первая из них была предложена Б. Я. Рябко [18] и названа им методом стопки книг. Этот метод отличается от интервального кодирования тем, что вместо числа всех букв между двумя одинаковыми указывается число различных букв между ними. Так, например, слово из (43) будет преобразовано в \(...0001002002\). Метод стопки книг можно рассматривать как упрощение схемы кодирования I.Z78. В этом случае слово \(U\) состоит только из букв (или слов одинаковой длины), которые при поступлении очередной буквы перенумеровываются. Впоследствии метод стопки книг был пере открыт П. Элайесом [55] и Дж. Бентли с соавторами [44].

Другая модификация интервального кодирования была предложена Э. Арнавутом и С. Маглиерасом [39]. Она заключается в том, что каждая буква исходного слова заменяется числом букв с большими номерами, разделяющих текущую и предыдущее включение буквы. Например, слово из (43) будет преобразовано в три последовательности, соответствующие трём различным буквам: \(a_3 : (...)000\), \(a_2 : (...)400\), \(a_1 : (...)800\). Декодирование нужно начинать с первой буквы алфавита, оставляя для других букв соответствующее количество пустых мест.

Для кодирования последовательности чисел, которая получается из исходной последовательности после применения интервального кода, можно использовать произвольный префиксный код для натуральных чисел. В частности, можно применить код Левенштейна [13].

Для интервального кодирования источника без памяти Б. Я. Рябко [18] получил оценку избыточности \(R \leq 2\log \log k + c\), где \(k\) — объем алфавита источника и \(c > 0\) — некоторая константа. Действительно, из равенства (9) следует оценка длины кода \(i\)-й буквы слова \(x_i\):

\[L(x_i) \leq \log N_i + 2\log \log N_i + c,\]  \hspace{1cm} (44)

где \(N_i\) — расстояние до предыдущего вхождения буквы \(x_i\) в слове \(x\) и \(c > 0\) — некоторая константа. Предположим что частота буквы близка к ее вероятности. Тогда аналогично (39) имеем \(E \log N_i(X) = H(X)\). Поскольку \(H(X) \leq \log k\), из неравенства (44) получаем неравенство

\[EL(x_i) \leq H(X) + 2\log \log k + c,\]

где \(c > 0\) — некоторая константа. Из последнего неравенства и формулы (42) следует искомая оценка избыточности.

Ясно, что время кодирования и декодирования интервальными методами линейно относительно длины сообщения и объем используемой памяти конечен, если ограничен объем алфавита источника.

14. Кодирование текстов на естественных языках

Значительной частью реально сжимаемых данных являются тексты на естественных языках (русском, английском и др.). Описанные выше схемы кодирования
не используют в явном виде особенности текстов на естественных языках, в частности, не используют раздельение текста на слова, набор которых весьма ограничен. Кроме того, на практике часто необходимо обеспечить произвольный доступ к текстовым данным, в то время как рассмотренные выше методы обеспечивают только последовательный доступ к сжатым данным. При кодировании больших текстов часто оказывается целесообразным составить словарь и кодировать слова целиком, рассматривая их как буквы нового алфавита — словаря. Такие методы изучались Э. Шварцем [98], А. Мохо [81] и Т. Беллом с соавторами [41].

Исследуя естественные языки, Г. К. Ципф [121] обнаружил, что если занумеровать слова естественного языка в порядке убывания частоты встречаемости, то вероятность слова будет приблизительно обратно пропорциональна его номеру. Для объяснения этой закономерности было выдвинуто несколько гипотез. В частности, Б. Мандельброт [75] показал, что если пробел между словами рассматривать как случайный символ, то будет выполняться закон Ципфа. Он же получил это распределение, исходя из предположения, что эволюционный процесс выбора длин слов может быть описан как случайное блюхание [76]. Б. Я. Рэбко [16] показал, что распределение Ципфа близко к оптимальному универсальному распределению на множестве источников без памяти с упорядоченными вероятностями букв.

Предполагая, что вероятности букв подчиняются распределению Ципфа, нетрудно оценить энтропию источника. Пусть $X$ — источник без памяти с вероятностями букв $p(a_i) = \frac{1}{\gamma(k)}$, где $k$ — число букв в алфавите и

$$
\gamma(k) = \sum_{i=1}^{k} \frac{1}{i} = \ln k + c + O(1/k)
$$

(45)

и $c$ — постоянная Эйлера. Из (45) и известного неравенства

$$
\sum_{i=1}^{k} \frac{\log i}{i} \geq \frac{\ln^2 k}{2\ln 2} - \frac{2}{2}
$$

получаем

$$
H(X) = \sum_{i=1}^{k} \frac{\log i \gamma(k)}{i \gamma(k)} = \log \gamma(k) + \frac{1}{\gamma(k)} \sum_{i=1}^{k} \frac{\log i}{i} \geq \frac{\log k}{2}.
$$

Таким образом, применение пословного префиксного кодирования для текстов на естественных языках может сжать текст не более, чем в два раза лучше по сравнению с равномерным кодированием каждого слова $[\log k]/2$ битами. М. Гутман [63] нашел чрезвычайно простой код Хаффмена для источника с распределением Ципфа и $k = 2^t$, где $t > 0$ — целое. Код $i$-ой буквы состоит из двух частей: вторая — $Bin(i)$ (двоичная запись числа $i$), первая — двоичная запись длины слова $Bin(i)$ с использованием ровно $t$ битов.

При словарном кодировании необходимо хранить словарь, что требует значительного объема памяти. Редко встречающиеся слова записывать в словарь нецелесообразно. Вопрос, какую часть слов хранить, а какую кодировать побуквенно, исследован М. П. Шаровой [34]. Оказывается, что наиболее эффективным является включение в словарь только $O(k/\ln k)$ (при $k \to \infty$) наиболее вероятных слов, где $\varepsilon > 0$ — некоторая константа.

15. Кодирование источников с низкой энтропией
Другим практически важным классом источников являются источники с низкой энтропией. Возможность построения более простых, а значит и менее трудоемких по сравнению с общим случаем, кодов для источников с низкой энтропией была замечена еще К. Шенноном [100], который предложил первый специальный код для последовательности редких событий. Последовательности редких событий порождаются двуэнаконными источниками Бернулли с малой вероятностью единицы \( P(1) = p \). Они состоят из длинных серий нулей, разделенных редкими событиями — единицами.

К. Шеннон [100] предложил кодировать длины серий числами в \((2^m - 1\)-ичной системе, выделив один блок из \( m \) единиц как код запятой, разделяющей числа. Если выбрать \( m = \lfloor \log \frac{1}{p} \rfloor \), то можно получить оценку избыточности кодирования \( R = O(p \log \frac{1}{p}) \) при \( p \to 0 \).

Одним из наиболее известных (см. [35, 56]) способов сжатия таких источников является кодирование длин серий высоковероятных символов с помощью префиксного кода чисел натурального ряда, что является по-существу частным случаем интервального кодирования в модификации 3. Аранвуто и С. Маглипераса [39].

Оценка избыточности кодирования длин серий посредством кода Левенштейна [13]. Из равенства (9) заключаем, что длина кода \( i \)-ой серии удовлетворяет асимптотическому равенству

\[
L_i = \log l_i + \log \log l_i (1 + o(1)),
\]

когда длина серии \( l_i \to \infty \). Поскольку среднее расстояние между единицами равно числу \( 1/p \), то из (46) имеем неравенство

\[
R = p E L_i - H \leq p \log \log \frac{1}{p} (1 + o(1))
\]

при \( p \to 0 \). С другой стороны из неравенства (10) можно получить, что

\[
R \geq p \log \log \frac{1}{p} (1 + o(1))
\]

при \( p \to 0 \).

Другой эффективный способ кодирования длий серий был предложен С. Голомбом [61]. Согласно этому методу исходная последовательность разделяется на блоки \( A_i \), каждый из которых состоит из \( i \) нулей \( (0 \leq i \leq l - 1) \) и следующей за ними единицей. Блок \( A_l \) содержит \( l \) нулей, где \( l = [1/2 + \log \frac{1}{p}] \). Блоки кодируются специальным префиксным кодом. Избыточность кода Голомба равняется \( O(p) \) при \( p \to 0 \). Р. Галлгер и Д. ван Бурхис [59] показали, что если \( p \) таково, что \( p^i + p^{i+1} \leq 1 < p^i + p^{i+1} \), то код Голомба является оптимальным для префиксного кодирования длин серий. В работе В. Ф. Бабкина и Б. М. Книнского [2] предложен адаптивный вариант кода Голомба, позволяющий кодировать сообщения за один проход.

Б. Я. Рябо и М. П. Шарова [25, 33, 35] разработали несколько методов кодирования для марковских источников с низкой энтропией. Рассмотрим идею их метода на примере источника Бернулли с двуэнаконным алфавитом. Кодирование состоит из двух этапов. На первом этапе сообщение разбивается на блоки длины \( l = [1/\sqrt{p}] \). Блок из нулей кодируется одним нулем, если блок содержит хотя бы одну единицу, то его кодом будет весь блок с добавленной перед ним единицей. После
первого этапа длина сообщения сильно сокращается, а полученный код можно рассматривать как сообщение, порожденное марковским источником с согласованными вероятностями. На втором этапе к полученному после первого этапа слову применяется арифметический код, использующий вычисленные аналитически вероятности символов. Предложенный метод имеет в $1/\sqrt{p}$ раз большее быстродействие по сравнению с арифметическим кодированием при той же избыточности и используемой памяти, что и у арифметического кода.

Литература


[34] Шарова М. П. Влияние объема словаря на степень сжатия текста// Дискрет. анализ и исслед. операций. Сер. 1. 1999. Т. 6, №1. С. 86–96.


