Теория информации и кодирование

В. Н. Потапов

Институт математики им. С. Л. Соболева,
Новосибирский государственный университет, Новосибирск

XII летняя школа «Современная математика»,
г. Дубна, 19-30 июля 2012 г.

Алгоритмический подход к понятию сложности

Рассмотрим некоторую *вычислимую функцию* (*программу*) $\Pi : \{0,1\}^* \to A^*$, где $A^* = \bigcup\limits_{i=1}^{\infty} A^i$.

Определение

Двоичное слово $v \in \{0,1\}^*$ будем называть *кодом слова* $w \in A^*$ относительно программы Π, если $\Pi(v) = w$.

Определение

Сложностью слова w относительно программы Π называется

$$K(w|\Pi) = \min_{w=\Pi(v)} |v|.$$

Если $w \notin \Pi(\{0,1\}^*)$, т. е. слово w не может быть получено применением программы Π, то $K(w|\Pi) = \infty$.

Теорема (Колмогоров)

Существует универсальная программа Π_0 такая, что для любой программы Π и слова $w \in A^*$ выполнено неравенство $K(w|\Pi_0) \leq K(w|\Pi) + C_\Pi$, где константа C_Π зависит от программы Π, но не от слова w.

Теорема (Колмогоров)

Существует универсальная программа Π_0 такая, что для любой программы Π и слова $w \in A^*$ выполнено неравенство $K(w|\Pi_0) \leq K(w|\Pi) + C_\Pi$, где константа C_Π зависит от программы Π, но не от слова w.

Доказательство.

Произвольная программа Π является набором команд, т. е. задаётся некоторым словом u_Π. Рассмотрим программу Π_0, которая вначале по слову u_Π записывает программу Π, а потом запускает программу Π на слове v. Тогда кодом слова w относительно Π_0 будет пара слов: код программы Π и код слова w относительно программы Π. Преобразуем слово u_Π в двоичное слово $\varphi(u_\Pi)$ так, чтобы его можно было отделить от кода слова w относительно программы Π. По построению получаем $K(w|\Pi_0) \leq |\varphi(u_\Pi)| + K(w|\Pi)$.

Определение

Сложность слова w относительно некоторой универсальной программы Π_0 называется *колмогоровской сложностью*.

Утверждение 1.1

Если программа Π определена на всём множестве $\{0,1\}^*$, то функция $K(\cdot|\Pi)$ является частично вычислимой.

Теорема (Колмогоров)

Колмогоровская сложность $K(\cdot|\Pi_0)$ является невычислимой функцией.

Доказательство. Предположим противное. Пусть найдётся такая программа Π_1, которая вычисляет колмогоровскую сложность двоичных слов, т. е. ставит в соответствие двоичному слову w двоичную запись числа $K(w|\Pi_0)$. Заметим, что программа Π_1 должна быть всюду определена, поскольку для любого слова $w \in A^*$ найдётся вычисляющая его программа Π_w и
$K(w|\Pi_0) \leq K(w|\Pi_w) + C_{\Pi_w} < \infty$.
Построим программу Π_2, которая ставит в соответствие двоичной записи числа $m \in \mathbb{N}$ наименьшее из слов, удовлетворяющих неравенству $K(w|\Pi_0) \geq m$. Программа Π_2 перебирает двоичные слова в порядке возрастания и посредством программы Π_1 вычисляет их сложность, пока не встретится первое слово удовлетворяющее неравенству $K(w|\Pi_0) \geq m$. Обозначим его w_m.

Доказательство. Предположим противное. Пусть найдётся такая программа Π_1, которая вычисляет колмогоровскую сложность двоичных слов, т. е. ставит в соответствие двоичному слову w двоичную запись числа $K(w|\Pi_0)$. Заметим, что программа Π_1 должна быть всюду определена, поскольку для любого слова $w \in A^*$ найдётся вычисляющая его программа Π_w и
$K(w|\Pi_0) \leq K(w|\Pi_w) + C_{\Pi_w} < \infty$.
Построим программу Π_2, которая ставит в соответствие двоичной записи числа $m \in \mathbb{N}$ наименьшее из слов, удовлетворяющих неравенству $K(w|\Pi_0) \geq m$. Программа Π_2 перебирает двоичные слова в порядке возрастания и посредством программы Π_1 вычисляет их сложность, пока не встретится первое слово удовлетворяющее неравенству $K(w|\Pi_0) \geq m$. Обозначим его w_m. Из вычислимости (на любом аргументе) программы Π_1 следует, что и программа Π_2 вычислима на любом аргументе, так как множество слов сложности менее m конечно. Тогда $K(w_m|\Pi_2) = \lfloor \log m \rfloor + 1$, поскольку кодом w_m относительно программы Π_2 является двоичная запись числа m. Получаем неравенство

$$m \leq K(w_m|\Pi_0) \leq K(w_m|\Pi_2) + C_{\Pi_2} \leq \log m + 1 + C_{\Pi_2}.$$

Задача 1.1

Докажите, что $K(w|\Pi_0) \leq |w| + c$, где w — двоичное слово, c — некоторая константа.

Задача 1.2

Докажите, что существует менее 2^n двоичных слов w, для которых $K(w|\Pi_0) < n$.

Комбинаторный подход к понятию сложности

Пусть $M \subseteq A^*$ — некоторое множество слов и $w \in M$. Для однозначного задания слова из множества M достаточно указать его номер во множестве M относительно некоторого, например, лексикографического порядка. Поскольку словам из M присваиваются номера от 0 до $|M| - 1$, то для записи номера достаточно $\lceil \log |M| \rceil$ битов.

Определение

Комбинаторной сложностью слова w относительно множества M называется $L(w|M) = \log |M|$.

Определим множество $M(\overline{r})$ как множество слов с частотным составом $\overline{r} = (r_1, \ldots, r_k)$, т. е. если $w \in M(\overline{r})$, то в слове w имеется ровно r_i букв a_i для любого $i = 1, \ldots, k$.

Утверждение 1.2

$$|M(\bar{r})| = \frac{(r_1+\cdots+r_k)!}{r_1!\cdots r_k!}.$$

Утверждение 1.2

$|M(\overline{r})| = \frac{(r_1+\cdots+r_k)!}{r_1!\cdots r_k!}$.

Доказательство.
Обозначим $n = r_1 + \cdots + r_k$. Сначала рассмотрим случай $k = 2$. Докажем формулу

$$|M(r_1, n - r_1)| = \frac{n!}{r_1!(n - r_1)!} \tag{1}$$

методом индукции по n. Очевидно, что $|M(n, 0)| = |M(0, n)| = 1$. Поскольку любое слово из $M(r_1, n + 1 - r_1)$ получается приписыванием буквы a_2 к слову из $M(r_1, n - r_1)$ или буквы a_1 к слову из $M(r_1 - 1, n - (r_1 - 1))$, то из предположения индукции (1) имеем

$$|M(r_1, n + 1 - r_1)| = |M(r_1, n - r_1)| + |M(r_1 - 1, n - (r_1 - 1))| =$$

$$= \frac{n!}{r_1!(n - r_1)!} + \frac{n!}{(r_1 - 1)!(n + 1 - r_1)!} = \frac{(n + 1)!}{r_1!(n + 1 - r_1)!}.$$

Таким образом формула (1) доказана.

Утверждение 1.2

$|M(\overline{r})| = \frac{(r_1+\cdots+r_k)!}{r_1!\cdots r_k!}$.

При $k > 2$ будем доказывать утверждение методом индукции по k. Любое слово $w \in M(r_1, \ldots, r_k, r_{k+1})$ можно получить из слова $u \in M(r_1, \ldots, r_k + r_{k+1})$ заменив r_{k+1} букв a_k в слове u на буквы a_{k+1}. По доказанному выше это можно сделать $|M(r_k, r_{k+1})| = \frac{(r_k+r_{k+1})!}{r_k!r_{k+1}!}$ способами. Тогда

$$|M(r_1, \ldots, r_k, r_{k+1})| = |M(r_1, \ldots, r_k + r_{k+1})| \frac{(r_k + r_{k+1})!}{r_k!r_{k+1}!} =$$

$$\frac{(r_1 + \cdots + r_k + r_{k+1})!}{r_1! \ldots (r_k + r_{k+1})!} \frac{(r_k + r_{k+1})!}{r_k!r_{k+1}!} = \frac{(r_1 + \cdots + r_k + r_{k+1})!}{r_1! \ldots r_k!r_{k+1}!}.$$

формула Стирлинга

$n! = n^n e^{-n}\sqrt{2\pi n}(1 + o(1))$ при $n \to \infty$.

Утверждение 1.3

Пусть $w \in M(\bar{r})$ и $r_1 + \cdots + r_k = n$. Тогда $L(w|M) =$

$= n\left(\sum_{i=1}^{k} \frac{r_i}{n} \log \frac{n}{r_i} + \alpha_n\right)$, где

$0 \geq \alpha_n = \frac{1}{2n} \sum_{i=1}^{k} \log \frac{n}{r_i} - \frac{k-1}{2n} \log n + O\left(\frac{1}{n}\right)$ при $r_i \to \infty$ для всех $i = 1, \ldots, k$.

формула Стирлинга

$n! = n^n e^{-n}\sqrt{2\pi n}(1 + o(1))$ при $n \to \infty$.

Утверждение 1.3

Пусть $w \in M(\bar{r})$ и $r_1 + \cdots + r_k = n$. Тогда $L(w|M) =$
$= n\left(\sum\limits_{i=1}^{k} \frac{r_i}{n} \log \frac{n}{r_i} + \alpha_n\right)$, где
$0 \geq \alpha_n = \frac{1}{2n} \sum\limits_{i=1}^{k} \log \frac{n}{r_i} - \frac{k-1}{2n} \log n + O\left(\frac{1}{n}\right)$ при $r_i \to \infty$ для всех $i = 1, \ldots, k$.

Доказательство. Используя неравенство Бернулли $(1 + x)^n \geq 1 + nx$, справедливое для любого $x \geq -1$ и $n \in \mathbb{N}$, можно показать, что $(1 + \frac{1}{n})^n \leq (1 + \frac{1}{n+1})^{n+1}$. Отсюда методом индукции по n нетрудно вывести неравенство $\frac{n!}{r_1! \cdots r_k!} \leq \frac{n^n}{r_1^{r_1} \cdots r_k^{r_k}}$.

$$\log |M(\bar{r})| = \log \frac{n^n}{r_1^{r_1} \cdots r_k^{r_k}} + \frac{1}{2} \log \frac{2\pi n}{(2\pi)^k r_1 \cdots r_k}(1 + o(1)).$$

Задача 1.3

Выведите формулу для числа двоичных слов длины n с k единицами, никакие две из которых не встречаются в слове подряд.

Вероятностный подход к понятию сложности

Найдём функцию $h(p_1, \ldots, p_k)$, которая определяет количество информации, содержащееся в результатах опыта, имеющего k случайных исходов с вероятностями $(p_1, \ldots, p_k)$. Сначала предположим, что все исходы опыта равновероятны, тогда $h(1/k, 1/k, \ldots, 1/k) = \phi(k)$.

Пусть опыт B состоит в последовательном выполнении двух независимых экспериментов A' и A'', имеющих по k равновероятных исходов. Опыт B имеет k^2 равновероятных исходов, и для записи его результата нужно записать результаты опытов A' и A''. Следовательно должно быть справедливо равенство $\phi(k^2) = 2\phi(k)$. Аналогичным образом заключаем, что $\phi(k^i) = i\phi(k)$ для любых натуральных чисел $i > 0$.

Утверждение 1.4

$\phi(k) = C \log k$, где $C > 0$ — постоянная величина.

Утверждение 1.4

$\phi(k) = C \log k$, где $C > 0$ — постоянная величина.

Доказательство. Пусть $n > 1$ и $a > 2$ — целые числа. Найдется такое натуральное m, что справедливы неравенства

$$2^m \leq a^n < 2^{m+1}.$$

Тогда из монотонности функции ϕ имеем неравенство $m\phi(2) \leq n\phi(a) < (m+1)\phi(2)$. Кроме того, имеем $m \leq n \log a < m + 1$. Из двух последних неравенств получаем, что

$$\left| \frac{\phi(a)}{\log a} - \phi(2) \right| \leq \frac{\phi(2)}{m}.$$

Рассмотрим опыт B, имеющий n равновероятных исходов. Сгруппируем исходы опыта B в k групп так, чтобы i-я группа имела вероятность $\frac{m_i}{n}$. Тогда исход опыта B можно указать, определив сначала группу, в которую попал данный исход, а затем найдя место нужного исхода в этой группе. Проделаем n раз опыт B. Поскольку исходы опытов, содержащихся в каждой группе, равновероятны и опыт по определению места нужного исхода в i-й группе необходимо проводить в среднем в m_i из n случаях, то функция h должна удовлетворять равенству

$$nh\left(\frac{1}{n},\ldots,\frac{1}{n}\right) = nh\left(\frac{m_1}{n},\frac{m_2}{n},\ldots,\frac{m_k}{n}\right) + m_1 h\left(\frac{1}{m_1},\ldots,\frac{1}{m_1}\right) + \ldots$$

$$+m_k h\left(\frac{1}{m_k},\ldots,\frac{1}{m_k}\right).$$ Тогда

$$h\left(\frac{m_1}{n},\frac{m_2}{n},\ldots,\frac{m_k}{n}\right) = \log n - \frac{m_1}{n}\log m_1 - \cdots - \frac{m_k}{n}\log m_k = \sum_{i=1}^{k}\frac{m_i}{n}\log\frac{n}{m_i}.$$

Определение

Источником сообщений называется пара $\langle A, P\rangle$ из алфавита A и распределения вероятностей P, удовлетворяющего условиям

$$\sum_{a\in A} P(wa) = P(w) \text{ для любого слова } w \in A^* \text{ и } \sum_{a\in A} P(a) = 1.$$

Определение

Сложность слова $w \in A^*$ относительно вероятности P определяется равенством $I(w|P) = -\log P(w)$.

Определение

Энтропией источника сообщений называется

$$H(\langle A, P\rangle) = \lim_{n\to\infty} \frac{1}{n} \sum_{|w|=n} P(w) I(w|P).$$

Определение

Источником Бернулли называется стационарный источник последовательностей независимых испытаний. Т. е. источник сообщений для которого справедливо равенство $P(wa) = P(w)P(a)$ для любых $w \in A^n$, $a \in A$.

Утверждение 1.5

Пусть $\langle A, P \rangle$ — источник Бернулли. Тогда энтропия источника $\langle A, P \rangle$ существует и удовлетворяет равенству $H(\langle A, P \rangle) = -\sum\limits_{a \in A} P(a) \log P(a)$.

Определение

Источником Бернулли называется стационарный источник последовательностей независимых испытаний. Т. е. источник сообщений для которого справедливо равенство $P(wa) = P(w)P(a)$ для любых $w \in A^n$, $a \in A$.

Утверждение 1.5

Пусть $\langle A, P\rangle$ — источник Бернулли. Тогда энтропия источника $\langle A, P\rangle$ существует и удовлетворяет равенству $H(\langle A, P\rangle) = -\sum\limits_{a\in A} P(a)\log P(a)$.

Доказательство. Индукцией по n докажем формулу

$$-\sum_{|w|=n} P(w)\log P(w) = -n\sum_{a\in A} P(a)\log P(a) = nh(A).$$

$$\sum_{|w|=n}\sum_{a\in A} P(wa)\log P(wa) = \sum_{|w|=n}\sum_{a\in A} P(w)P(a)(\log P(w) + \log P(a)) =$$

$$\sum_{a\in A} P(a)\sum_{|w|=n}(P(w)\log P(w)) + \sum_{|w|=n} P(w)(\sum_{a\in A} P(a)\log P(a)) =$$

$$-nh(a) - h(a).$$

Пусть $w \in M(\overline{r})$, где $\overline{r} = (r_1, \ldots, r_k)$ и $r_1 + \cdots + r_k = n$. Определим *эмпирическую вероятность* каждой буквы как частоту её встречаемости в слове w, а эмпирическую вероятность слова как произведение вероятностей букв, составляющих слово. Имеем равенства:

$$P_{\overline{r}}(a_i) = \frac{r_i}{n} \quad \text{и} \quad P_{\overline{r}}(a_{i_1} \ldots a_{i_m}) = P_{\overline{r}}(a_{i_1}) \ldots P_{\overline{r}}(a_{i_m}),$$

Эмпирические вероятности всех слов $w \in M(\overline{r})$ равны и для любого слова $w \in M(\overline{r})$ имеется равенство

$$I(w|P_{\overline{r}}) = -\log((P_{\overline{r}}(a_1))^{r_1} \cdots (P_{\overline{r}}(a_k))^{r_k}) = -n\left(\sum_{i=1}^{k} \frac{r_i}{n} \log \frac{r_i}{n}\right).$$

Тогда из утверждения 1.3 имеем
$|I(w_n|P_{\overline{r}}) - L(w_n|M(\overline{r}))| = o(n) = o(I(w_n|P_{\overline{r_n}}))$, когда длина n слова w_n стремится к бесконечности.
Занумеруем слова из $M(\overline{r})$ в лексикографическом порядке. Каждое слово из $M(\overline{r})$ получит номер из целочисленного промежутка $[1, |M(\overline{r})|]$, который можно записать в виде двоичного слова длины $\lfloor \log |M(\overline{r})| \rfloor + 1$. Определим программу $\Pi_{\overline{r}}$, которая ставит в соответствие этому номеру слово из $M(\overline{r})$. Тогда $|K(w|\Pi_{\overline{r}}) - L(w|M(\overline{r}))| \leq 1$.

Пусть A — некоторый конечный алфавит.

Определение

Кодированием называется инъективное отображение $f : A^* \to \{0,1\}^*$.

Обратная функция f^{-1} — *декодирование* должна быть частично вычислимой, т. е. быть программой.

Определение

Префиксом слова w называется произвольное начало u слова w, т. е. если $w = uv$, то u — префикс слова w. Кодирование f называется *префиксным*, если из того, что кодовое слово $f(u)$ является префиксом кодового слова $f(w)$ следует, что и слово u является префиксом слова w.

Определение

Множество $D \subset A^*$ называется *разделимым*, если любая последовательность записанных подряд слов из D разделяется на слова из D единственным образом.

Т.е. из равенства $w_1 w_2 \dots w_m = w'_1 w'_2 \dots w'_n$ при $w_i, w'_i \in D$ следует, что $n = m$ и $w_i = w'_i$ при $i = 1, \dots, n$.

Задача 2.1

Докажите, что префиксное множество является разделимым.

Пусть $\varphi : A \to \{0,1\}^*$ некоторое отображение. Определим отображение $f : A^* \to \{0,1\}^*$ равенством $f(a_{i_1} \dots a_{i_n}) = \varphi(a_{i_1}) \dots \varphi(a_{i_n})$.

Определение

Кодирование f называется *побуквенным*.

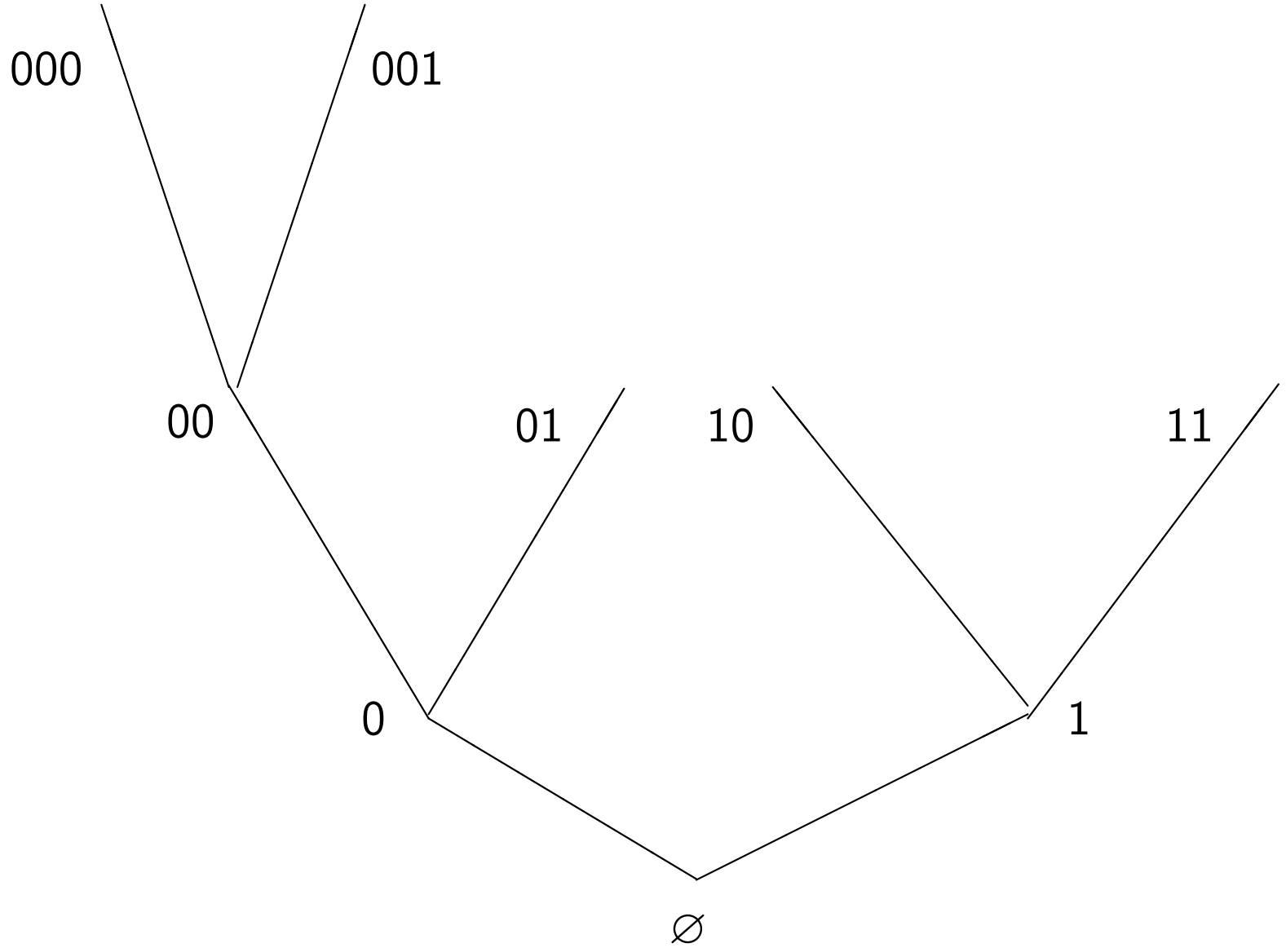
Задача 2.2

Отображение f является кодированием тогда и только тогда, когда $\varphi(A)$ разделимое множество.

Задача 2.3

Отображение f является префиксным кодированием тогда и только тогда, когда $\varphi(A)$ префиксное множество.

Рассмотрим некоторое двоичное дерево T. Вершины двоичного дерева T пометим двоичными словами (т. е. построим функцию $\psi : T \to \{0,1\}^*$) по следующему правилу: корень дерева пометим пустым словом, если вершина $t \in T$ уже помечена словом u, то её левого сына пометим словом $u0$, а правого — $u1$. Вершины k-ичного дерева можно аналогичным образом пометить словами k-буквенного алфавита.
Обозначим через $L(T)$ множество листьев (висячих вершин) дерева T.


000
001
00
01
10
11
0
1
∅

Утверждение 2.1

Множество $D \subset \{0,1\}^*$ является префиксным тогда и только тогда, когда $D \subseteq \psi(L(T))$ для некоторого двоичного дерева T.

Утверждение 2.1

Множество $D \subset \{0,1\}^*$ является префиксным тогда и только тогда, когда $D \subseteq \psi(L(T))$ для некоторого двоичного дерева T.

Доказательство.

Необходимость ($\Rightarrow$). Рассмотрим такое дерево $\overline{T}$, что $D \subseteq \psi(\overline{T})$. Из определения отображения ψ видно, что если вершина t помечена словом $\psi(t)$, то префиксами слова $\psi(t)$ помечены предки вершины t и только они. Значит вершины дерева $\overline{T}$, помеченные словами из множества D, не являются предками друг друга. Удалив из дерева $\overline{T}$ всех потомков вершин, помеченных словами из D, получим искомое дерево T.

Утверждение 2.1

Множество $D \subset \{0,1\}^*$ является префиксным тогда и только тогда, когда $D \subseteq \psi(L(T))$ для некоторого двоичного дерева T.

Достаточность ($\Leftarrow$). Поскольку листья дерева T не являются предками друг друга, то и соответствующие им слова не являются префиксами друг друга. Таким образом, $\psi(L(T))$ является префиксным множеством.

Теорема (неравенство) Крафта — Макмиллана

1) Пусть $D \subset \{0,1\}^*$ — разделимое множество, $|D| = k$ и l_i — длина i-го слова из D. Тогда справедливо неравенство $\sum_{i=1}^{k} 2^{-l_i} \leq 1$ (*).

2) Если выполнено неравенство (*), то найдётся префиксное множество $D \subset \{0,1\}^*$ с длинами кодовых слов $l_1, l_2, \ldots, l_k$.

Теорема (неравенство) Крафта — Макмиллана

1) Пусть $D \subset \{0,1\}^*$ — разделимое множество, $|D| = k$ и l_i — длина i-го слова из D. Тогда справедливо неравенство $\sum_{i=1}^{k} 2^{-l_i} \leq 1$ (*).
2) Если выполнено неравенство (*), то найдётся префиксное множество $D \subset \{0,1\}^*$ с длинами кодовых слов $l_1, l_2, \ldots, l_k$.

Доказательство. Пусть $S(n,t)$ — количество различных упорядоченных наборов по n слов из множества D, суммарная длина которых равняется t. Из определения разделимости множества видно, что различным наборам $(w_1, \ldots, w_n) \in D^n$ соответствуют различные слова $W = w_1 \ldots w_n$, $|W| = t$. Поэтому $S(n,t) \leq 2^t$. Кроме того, $S(n,t) = 0$ при $t > N = n \max\limits_i l_i$.

Пусть $x = \sum\limits_{i=1}^{k} 2^{-l_i}$. Тогда

$$x^n = \sum\nolimits_{1 \leq i_1, \ldots, i_n \leq k} 2^{-(l_{i_1} + \cdots + l_{i_n})} = \sum_{t=1}^{N} S(n,t) 2^{-t} \leq N = n \max\nolimits_i l_i.$$

Следовательно $x^n = O(n)$ при $n \to \infty$.

Будем считать, что $l_1 \leq l_2 \leq \cdots \leq l_k$. Достаточно рассмотреть случай, когда в формуле (*) имеет место равенство. Действительно, число $m = 2^{l_k} - \sum_{i=1}^{k} 2^{l_k - l_i}$ является целым и, добавив к набору длин $l_1, l_2, \ldots, l_k$ ещё m чисел l_k, получим равенство $\sum_{i=1}^{k} 2^{-l_i} + m2^{-l_k} = 1$ для расширенного набора.

Проведём доказательство методом индукции по k. При $k = 2$ имеем $l_1 = l_2 = 1$ и $D = \{0, 1\}$. Пусть неравенство (*) доказано для k слагаемых. Поскольку мы рассматриваем случай, когда $\sum_{i=1}^{k+1} 2^{-l_i} = 1$, последние два слагаемых равны: $l_k = l_{k+1}$.

Рассмотрим набор длин $l_1, l_2, \ldots, l_{k-1}, l_k - 1$. Поскольку для этого набора справедливо равенство (*), по предположению индукции найдётся префиксное множество D' с соответствующими длинами слов. Пусть $u \in D'$ и $|u| = l_k - 1$. Тогда определим $D = (D' \setminus \{u\}) \cup \{u0, u1\}$.

Пусть заданы некоторый источник сообщений $\langle A, P \rangle$ и кодирование $f : A^* \to \{0,1\}^*$.

Определение

Стоимостью кодирования f источника $\langle A, P \rangle$ называется $C_n(f,P) = \sum\limits_{|w|=n} |f(w)|P(w)$.

Стоимость кодирования $C_n(f,P)$ равна средней длине кодового слова, вычисленной по всем словам длины n.

Определение

Разность $R_n(f,P) = C_n(f,P) - H_n(P)$ называется *избыточностью*. Величину $r(f,P) = \overline{\lim\limits_{n\to\infty}} \frac{1}{n} R_n(f,P)$ будем называть *предельной избыточностью* кодирования f.

Теорема Шеннона

1) Для любого префиксного кодирования f и любого источника сообщений $\langle A, P \rangle$ избыточность неотрицательна, т. е. $R_n(f, P) \geq 0$.
2) Для любого источника сообщений $\langle A, P \rangle$ найдётся такое префиксное кодирование f, что $r(f, P) = 0$.

Теорема Шеннона

1) Для любого префиксного кодирования f и любого источника сообщений $\langle A, P\rangle$ избыточность неотрицательна, т. е. $R_n(f, P) \geq 0$.
2) Для любого источника сообщений $\langle A, P\rangle$ найдётся такое префиксное кодирование f, что $r(f, P) = 0$.

Доказательство. Из определений, неравенства Йенсена для функции $\log t$ и неравенства Крафта — Макмиллана для произвольного префиксного кодирования f имеем

$$-R_n(f, P) = \sum_{|w|=n} P(w) \log \frac{1}{P(w)} - \sum_{|w|=n} |f(w)| P(w) =$$

$$= \sum_{|w|=n} P(w) \log \frac{2^{-|f(w)|}}{P(w)} \leq \log \sum_{|w|=n} 2^{-|f(w)|} \leq \log 1 = 0.$$

Для любого слова $w \in A^n$ определим $l_w = \lceil \log \frac{1}{P(w)} \rceil$. Имеем

$$\sum_{|w|=n} 2^{-l_w} \leq \sum_{|w|=n} P(w) = 1.$$

Из неравенства Крафта — Макмиллана следует, что найдётся префиксное кодирование $f_n : A^n \to \{0,1\}^*$ с длинами кодовых слов $|f_n(w)| = l_w$. Определим кодирование f равенством $f(w) = 2\mathrm{Bin}(|w|)f_{|w|}(w)$. Слово $f(u)$ не может быть префиксом $f(v)$ при $|u| \neq |v|$, из-за того, что $2\mathrm{Bin}$ — префиксное кодирование натурального ряда, а при $|u| = |v| = n$ и $u \neq v$, из-за того, что f_n — префиксное кодирование A^n.

$$|f(w)| = \left\lceil \log \frac{1}{P(w)} \right\rceil + |2\mathrm{Bin}(w)| \leq \log \frac{1}{P(w)} + 2\log|w| + 2,$$

$$R_n(f,P) \leq \sum_{|w|=n} P(w)\left(\log \frac{1}{P(w)} + 2\log|w| + 2\right) - \sum_{|w|=n} P(w)\log\frac{1}{P(w)}$$

$$= \sum_{|w|=n} P(w)(2\log n + 2).$$

$$2\mathrm{Bin}(n) = \underbrace{0\ldots0}_{\mathrm{Bin}(n)}\mathrm{Bin}(n)$$

Следующее кодирование множества $\mathbb{N}\cup\{0\}$ было предложено В. И. Левенштейном. Введем обозначение $\lambda(n) = |\mathrm{Bin}'(n)|$ и $\lambda^k(n) = \lambda(\lambda^{k-1}(n))$, где $\lambda^1(n) = \lambda(n)$. Если $\lambda^k(n) = 0$ и $\lambda^{k-1}(n) > 0$, то определим

$$\mathrm{Lev}(n) = \underbrace{1\ldots1}_{k}0\mathrm{Bin}'(\lambda^{k-2}(n))\ldots\mathrm{Bin}'(\lambda(n))\mathrm{Bin}'(n).$$

Пример

$\mathrm{Lev}(0) = 0$, $\mathrm{Lev}(1) = 10$, $\mathrm{Lev}(5) = 1110\,0\,01$, $\mathrm{Lev}(62) = 11110\,0\,01\,11110$.

Преобразование двоичной записи числа n в код $\mathrm{Lev}(n)$ осуществляется достаточно просто, поскольку $\lambda(n) = |\mathrm{Bin}'(n)|$, $\lambda^2(n) = |\mathrm{Bin}'(\lambda(n))|$ и так далее.

Задача 2.4

Кодирование Lev является префиксным.

Задача 2.5

Для любого натурального k справедливо асимптотическое равенство при $n \to \infty$
$|\mathrm{Lev}(n)| = \log n + \log\log n + \cdots + \underbrace{\log\log\ldots\log}_{k} n(1 + o(1))$.

Задача 2.6

Пусть $\mathrm{B} : \mathbb{N} \to \{0,1\}^*$ — некоторое кодирование натурального ряда, $\alpha < 1$ и найдётся такое $k \in \mathbb{N}$, что
$|\mathrm{B}(n)| = \log n + \log\log n + \cdots + \underbrace{\log\log\ldots\log}_{k} n(\alpha + o(1))$
при $n \to \infty$. Тогда множество $\mathrm{B}(\mathbb{N})$ не разделимое.

Кодирование Шеннона

Пронумеруем буквы алфавита так, чтобы $P(a_1) \geq P(a_2) \cdots \geq P(a_k) > 0$. Определим числа σ_i рекуррентно: $\sigma_1 = 0, \sigma_{i+1} = \sigma_i + P(a_i)$ при $1 \leq i \leq k$. Ясно, что $0 \leq \sigma_i < 1$ для любого $i = 1, \ldots, k$. В качестве кодового слова $f_{Sh}(a_i)$ возьмем $\lceil \log \frac{1}{P(a_i)} \rceil$ первых после запятой символов в позиционной двоичной записи числа σ_i.

Пример

Пусть $P(a_1) = 1/2, P(a_2) = 1/3, P(a_3) = 1/8, P(a_4) = 1/24$. Тогда имеем двоичные записи чисел: $\sigma_1 = 0,000.., \sigma_2 = 0,1000.., \sigma_3 = 0,1101..,$ $\sigma_4 = 0,11110...$ По определению кода Шеннона получаем $f_{Sh}(a_1) = 0$, $f_{Sh}(a_2) = 10$, $f_{Sh}(a_3) = 110, f_{Sh}(a_4) = 11110$.

Утверждение 3.1

1) Кодирование Шеннона f_{Sh} — префиксное.
2) $R_1(f_{Sh}, P) \leq 1$ для любого источника сообщений $\langle A, P \rangle$.

Утверждение 3.1

1) Кодирование Шеннона f_{Sh} — префиксное.
2) $R_1(f_{Sh}, P) \leq 1$ для любого источника сообщений $\langle A, P \rangle$.

Доказательство. Из определения позиционной двоичной записи числа следует, что n первых после запятой двоичных знаков чисел a и b $(1 > a > b > 0)$ совпадают, если и только если $a - b < 2^{-n}$. Пусть $j > i$, тогда

$$\sigma_j - \sigma_i \geq P(a_i) = 2^{-\log(1/P(a_i))} \geq 2^{-\lceil \log(1/P(a_i)) \rceil} = 2^{-|f_{Sh}(a_i)|}.$$

Таким образом, первые после запятой $|f_{Sh}(a_i)|$ символов в двоичной записи числа σ_j не совпадают с кодовым словом $f_{Sh}(a_i)$. Поскольку $P(a_j) \leq P(a_i)$, имеем $|f_{Sh}(a_j)| \geq |f_{Sh}(a_i)|$, и префиксность множества $f_{Sh}(A)$ доказана.

Арифметическое кодирование

Все наборы из n букв алфавита A упорядочим лексикографически, т. е. $a_{i_1} a_{i_2} \ldots a_{i_n} \prec a_{j_1} a_{j_2} \ldots a_{j_n}$, если $i_k = j_k$ при $k < l$ и $i_l \leq j_l$. Для каждого слова $x \in A^n$ определим величины

$$L(x) = \sum_{y \prec x, y \in A^n} P(y) \quad \text{и} \quad R(x) = L(x) + P(x).$$

Разделим полуинтервал $[0, 1)$ на полуинтервалы $[L(x), R(x))$. В каждом полуинтервале длины l, $l \leq 1$ найдется двоично-рациональное число со знаменателем не большим, чем $2^{\lceil -\log l \rceil}$, поскольку разность между любыми двумя ближайшими числами с таким знаменателем не превосходит l ($2^{-\lceil -\log l \rceil} \leq 2^{\log l} = l$). Каждому слову $x \in A^n$ поставим в соответствие двоично-рациональное число $q(x) \in [L(x), R(x))$ со знаменателем, не большим $2^{\lceil -\log P(x) \rceil}$.

В качестве кода $f(x)$ рассмотрим двоичную запись числителя числа $q(x)$, которая имеет длину, равную $\lceil -\log P(x) \rceil$. Тогда $|f(x)| = \lceil -\log P(x) \rceil$.

Поскольку полуинтервалы $[L(x), R(x))$, соответствующие различным словам x одинаковой длины, не пересекаются, все числа $q(x)$, $x \in A^n$ попарно различны, т. е. отображение $f : A^n \to E^*$ инъективно. Если для всех $x \in A^n$ и $a_i \in A$ справедливо неравенство $P(a_i|x) \leq 1/2$, то при добавлении каждой следующей буквы интервал уменьшается не менее чем в два раза. Тогда $|f(xa_i)| > |f(x)|$ и отображение $f : A^* \to \{0,1\}^*$ оказывается инъективным.

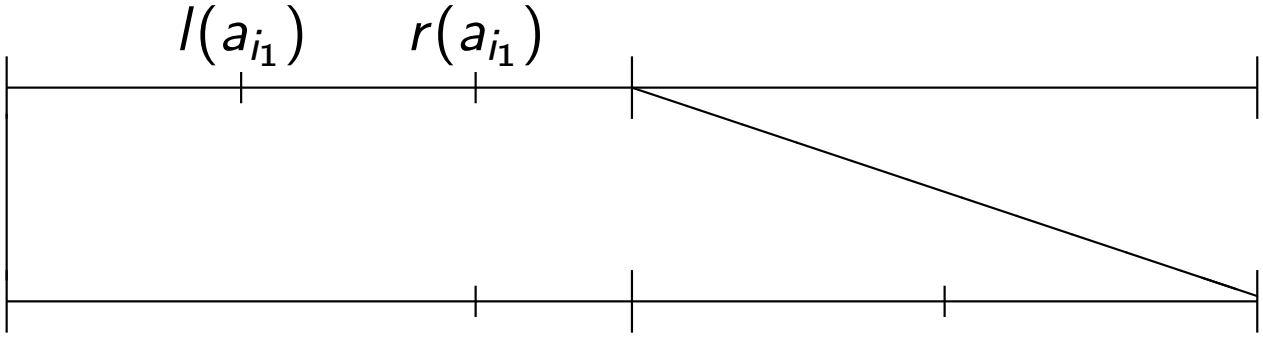
Избыточность $R_n(f, P)$ не превосходит единицы и предельная избыточность кодирования f равняется нулю.

Положим, что $\langle A, P\rangle$ — источник Бернулли и $P(a_i) \leq 1/4$ для всех i, $1 \leq i \leq |A|$. Длина $t > 0$ двоичной записи чисел, с которыми в процессе кодирования выполняются арифметические операции, является параметром кодирования и должна удовлетворять неравенству $P(a_i) \geq 1/2^{t-2}$ для всех букв $a_i \in A$.

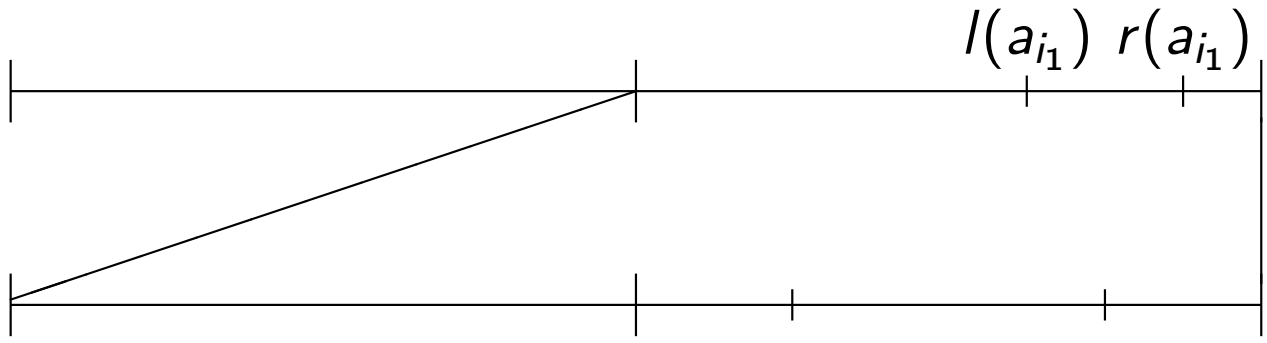
Определим величины $\sigma_1 = 0$, $\sigma_i = \sigma_{i-1} + P(a_{i-1})$. Разделим полуинтервал $[0, 1)$ на полуинтервалы $i(a_i) = [l(a_i), r(a_i))$, где $l(a_i) = \lfloor \sigma_i 2^t \rfloor / 2^t$ и $r(a_i) = \lfloor \sigma_{i+1} 2^t \rfloor / 2^t$.

Каждой букве $a_i \in A$ соответствует i-й полуинтервал, длина которого приблизительно равна $P(a_i)$. Пусть $x = a_{i_1} a_{i_2} \dots a_{i_n}$. Поскольку $P(a_{i_1}) \leq 1/4$, т. е. длина полуинтервала не превышает $1/4$, возможны три случая:
1) $i(a_{i_1}) \subset [0, 1/2)$;
2) $i(a_{i_1}) \subset [1/2, 1)$;
3) $i(a_{i_1}) \subset [1/4, 3/4)$.
Пусть $I(x)$ — полуинтервал, соответствующий слову $x \in A^n$. Тогда в первом случае имеем $I(x) \subset i(a_{i_1}) \subset [0, 1/2)$ и первым символом после запятой в двоичной записи любого числа $q \in I(x)$ является 0 . Поэтому 0 — первый символ кода $f_A(x)$. Во втором случае первым символом кода $f_A(x)$ является 1. В третьем случае первый символ пока неизвестен.

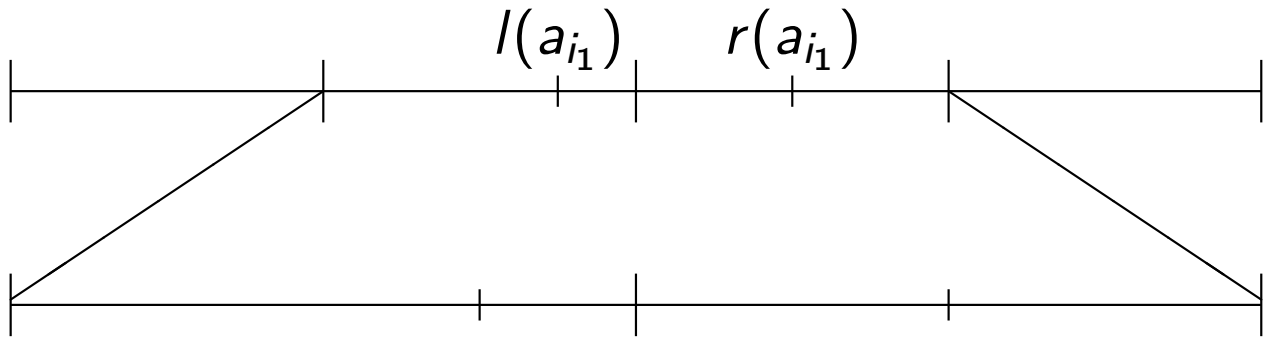
Проведем операцию изменения масштаба (растяжения) полуинтервала. В первом случае новый полуинтервал — $[2l(a_{i_1}), 2r(a_{i_1}))$.



Во втором случае — $[2l(a_{i_1}) - 1, 2r(a_{i_1}) - 1)$.



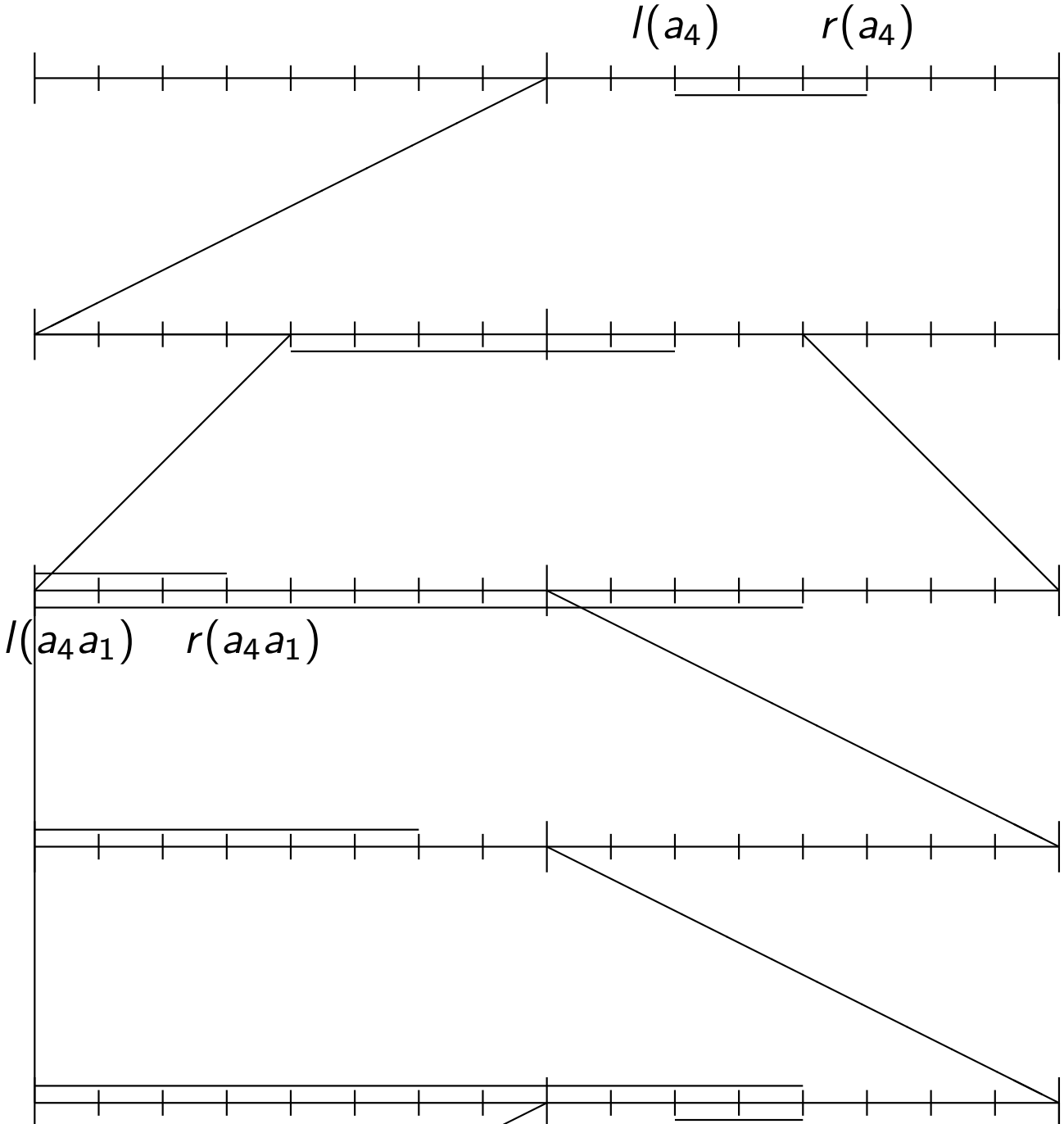
В третьем случае — $[2l(a_{i_1}) - 1/2, 2r(a_{i_1}) - 1/2)$.



В итоге получим полуинтервал $[l/2^t, r/2^t)$, где числа l и r — целые, причём $0 \leq l < r \leq 2^t$ и $r - l > 2^{t-2}$. Из $[l/2^t, r/2^t)$ выделим полуинтервал, соответствующий второй букве слова x: $i(a_{i_1}a_{i_2}) = [l(a_{i_1}a_{i_2}), r(a_{i_1}a_{i_2}))$, где
$l(a_{i_1}a_{i_2}) = (l + \lfloor (r-l)\sigma_{i_2} \rfloor)/2^t$ и
$r(a_{i_1}a_{i_2}) = (l + \lfloor (r-l)\sigma_{1+i_2} \rfloor)/2^t$. Проделаем с полуинтервалом $i(a_{i_1}a_{i_2})$ те же операции изменения масштаба, что и с полуинтервалом $i(a_{i_1})$. В результате получим начало кода $f_A(x)$, соответствующее двум начальным буквам. Выполнив такую же процедуру для букв $a_{i_3}, a_{i_4}, \ldots a_{i_n}$, получим код $f_A(x)$. Полуинтервалы, соответствующие словам одинаковой длины, не пересекаются. Соответствующий слову x полуинтервал целиком содержится в полуинтервале, соответствующем каждому префиксу слова x.

Пример

Рассмотрим арифметическое кодирование при $t = 4$ слова $a_4 a_1 a_2$, порождённого источником Бернулли с алфавитом $A = \{a_1, a_2, a_3, a_4, a_5\}$ и вероятностями букв $P(a_1) = P(a_3) = 1/4$, $P(a_2) = P(a_4) = P(a_5) = 1/6$.

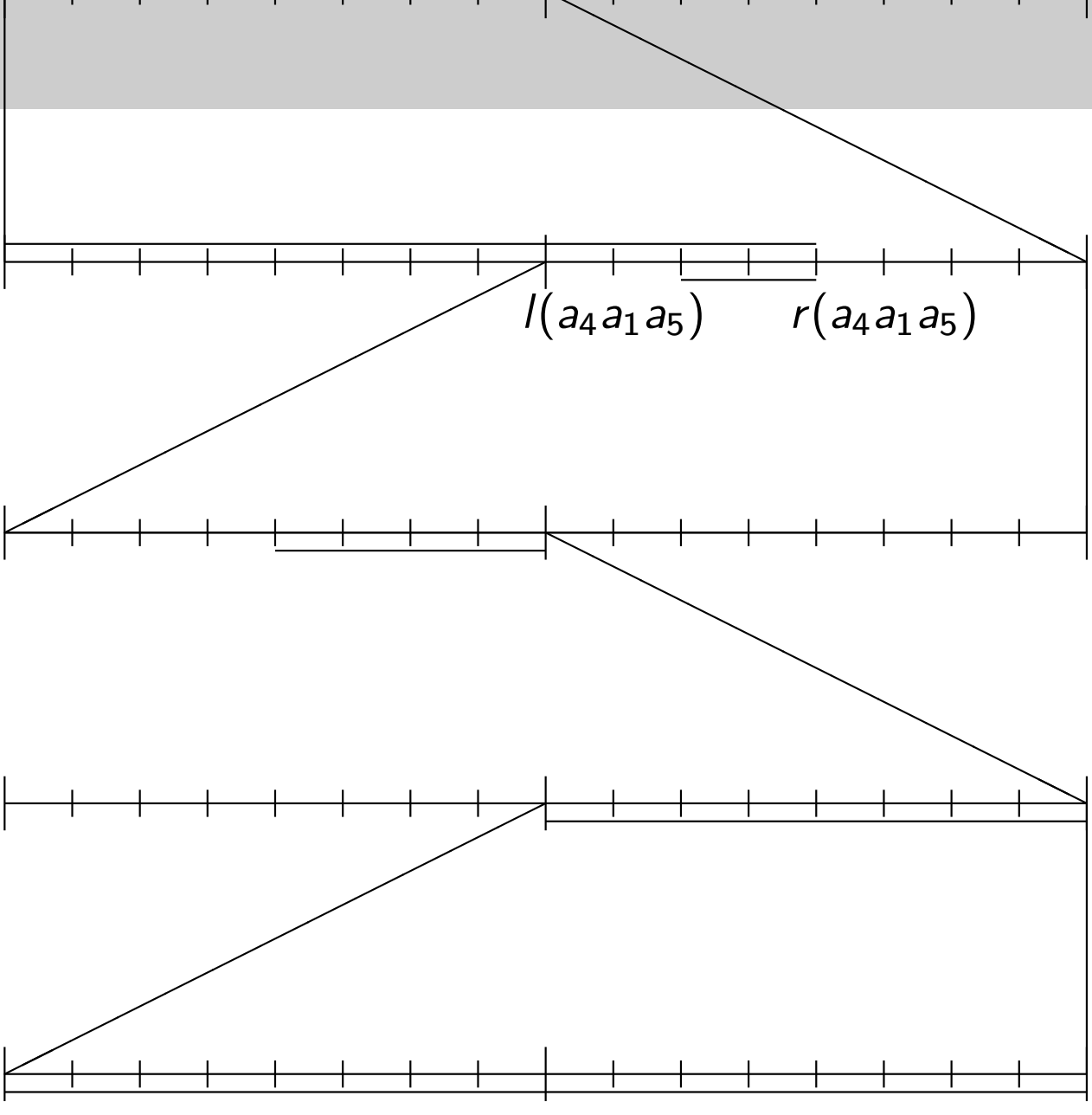


$$l(a_4) = \frac{\lfloor 16 \cdot 2/3 \rfloor}{16} = \frac{10}{16}$$

$$r(a_4) = \frac{\lfloor 16 \cdot 5/6 \rfloor}{16} = \frac{13}{16}$$

$$l(a_4a_1) = \frac{\lfloor 12 \cdot 0 \rfloor}{16} = 0$$

$$r(a_4a_1) = \frac{\lfloor 12/4 \rfloor}{16} = \frac{3}{16}$$



$l(a_4a_1a_5) = \frac{\lfloor 12 \cdot 5/6 \rfloor}{16} =$

$r(a_4a_1a_5) = \frac{\lfloor 12 \cdot 1 \rfloor}{16} = \frac{12}{16}$

$f_A(a_4a_1a_2) = 1010101.$

Интервальное кодирование

При интервальном кодировании каждая буква исходной последовательности заменяется на число, равное количеству букв до предыдущего включения той же буквы. Перед началом каждого слова помещается весь алфавит, чтобы можно было единообразно кодировать все, в том числе первое, включения буквы в слово.

Пример

Слово $(a_1a_2a_3)a_3a_3a_3a_2a_2a_2a_1a_1a_1a_3$ будет преобразовано в последовательность чисел 1115119117.

Числа можно кодировать произвольным префиксным кодом натурального ряда.

Утверждение 4.2

Пусть $\langle A, P \rangle$ — источник Бернулли с алфавитом $A = \{a_1, \ldots, a_k\}$, а f_I — интервальное кодирование. Тогда $r(f_I, P) \leq \log \log k + C$.

Утверждение 4.2

Пусть $\langle A, P\rangle$ — источник Бернулли с алфавитом $A = \{a_1, \ldots, a_k\}$, а f_I — интервальное кодирование. Тогда $r(f_I, P) \leq \log\log k + C$.

Доказательство. Пусть в слове $w \in A^n$ буква a_i встречается r_i раз на $t_1, t_2, \ldots, t_{r_i}$ местах. Тогда количество битов, затраченное на кодирование всех, за исключением первого, вхождений буквы a_i, не превышает $\sum_{j=1}^{r_i-1}(\log(t_{j+1} - t_j) + \log\log(t_{j+1} - t_j + 1) + C)$.

Из неравенства Йенсена для выпуклых вверх функций $\log x$ и $\log\log(x+1)$ получаем, что количество битов, затраченных на кодирование всего слова w, за исключением первых вхождений каждой буквы, не превышает суммы

$\sum_{i=1}^{k} r_i \sum_{j=1}^{r_i-1} \frac{1}{r_i}(\log(t_{j+1} - t_j) + \log\log(t_{j+1} - t_j + 1)) + Cn \leq$

$Cn + n\sum_{i=1}^{k} \frac{r_i}{n} \log \frac{n}{r_i} + n\sum_{i=1}^{k} \frac{r_i}{n} \log\log\left(1 + \frac{n}{r_i}\right) \leq$

$n(C + \log\log(k+1)) + n\sum_{i=1}^{k} \frac{r_i}{n} \log \frac{n}{r_i}$.

Тогда $\frac{1}{n}|f_I(x)| - H(x) \leq \log\log k + C + \frac{C'(k)}{n}$.

Утверждение 3.3

Для источника Бернулли $\langle A, P \rangle$ справедливо неравенство $0 \leq H(P) - \sum\limits_{x \in A^n} P(x)H(x) \leq \frac{k-1}{n \ln 2}$.

Утверждение 3.3

Для источника Бернулли $\langle A, P\rangle$ справедливо неравенство $0 \le H(P) - \sum\limits_{x\in A^n} P(x)H(x) \le \frac{k-1}{n\ln 2}$.

Доказательство. Пусть $x \in A^n$ и $|A| = k$. Определим $Q(a_i) = \frac{r_i}{n}$, где r_i — количество букв a_i в слове x, $i = 1, \ldots, k$. Тогда из неравенства Йенсена следует неравенство

$$H(x) = \sum_{i=1}^{k} Q(a_i) \log \frac{1}{Q(a_i)} \le \sum_{i=1}^{k} Q(a_i) \log \frac{1}{P(a_i)} = \frac{1}{n} \log \frac{1}{P(x)}.$$

Следовательно

$$\sum_{|x|=n} P(x)H(x) \le \frac{1}{n} \sum_{|x|=n} P(x) \log \frac{1}{P(x)} = H(P).$$

Преобразование Барроуза – Уилера

Пусть $x = x_1 \dots x_n$ — некоторое слово в двоичном алфавите. Превратим слово x в циклическое слово, определив $x_0 = x_n, x_{-1} = x_{n-1}, \dots, x_{-i} = x_{n-i}, \dots$. Каждой букве x_i поставим в соответствие контекст $s(x_i)$ длины n: $s(x_i) = x_{i-n} \dots x_{i-2} x_{i-1}$. Упорядочим контексты лексикографически (читая контекст справа — налево).

Определение

Преобразованием Барроуза — Уилера слова x называется слово $\mathrm{BW}(x)$ длины n, составленное из букв слова x в порядке их контекстов, т. е. $\mathrm{BW}(x) = x_{i_1} x_{i_2} \dots x_{i_n}$, где $s(x_{i_1}) \le s(x_{i_2}) \le \dots \le s(x_{i_n})$.

Рассмотрим матрицу вращений слова x и расставим строки матрицы в лексикографическом (начиная справа) порядке. Первый столбец получившейся матрицы и есть слово $\mathrm{BW}(x)$.

Пусть $x = 110110100$. Тогда матрица вращений слова x имеет вид

$$
\begin{array}{ccccccccc}
1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0.
\end{array}
$$

После упорядочения получаем матрицу

$$
\begin{array}{ccccccccc}
1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1.
\end{array}
$$

Тогда $\mathrm{BW}(110110100) = 101111000$.

Кодирование Лемпела – Зива

Через x_l^r обозначим слово, состоящее из букв слова $x = a_{i_1} \dots a_{i_n}$, начиная с l-ой и заканчивая r-ой, т. е. $x_l^r = a_{i_l} \dots a_{i_r}$. Разделим слово $x_1^n \in A^n$ на подслова σ_i, $i = 1, \dots, m$ по следующему правилу. Пусть начало слова x_1^n уже разделено на подслова, т. е. представляет собой конкатенацию подслов $\sigma_1 \sigma_2 \dots \sigma_{i-1}$ и $x_1^n = \sigma_1 \dots \sigma_{i-1} x_{l_i}^n$. Выберем следующее подслово $\sigma_i = x_{l_i}^{l_{i+1}-1}$ так, чтобы слово $x_{l_i}^{l_{i+1}-2}$ было длиннейшим из префиксов слова $x_{l_i}^n$, которые уже содержатся как подслова в слове $x_1^{l_{i+1}-3}$, т. е. $\sigma_i = x_{l_i-d_i}^{l_{i+1}-d_i-2} a_{j_i}$, где $d_i \leq l_i$.

Каждое подслово σ_i определяется тройкой чисел $(d_i, l_{i+1} - l_i, j_i)$.

Пример

Слово $a_1a_2a_2a_1a_2a_1a_1a_2a_1a_2a_1a_2$ разделяется на подслова a_1, a_2, a_2a_1, $a_2a_1a_1$, $a_2a_1a_2a_1a_2$ и кодируется последовательностью троек чисел $(1,1,1)$, $(2,1,2)$, $(1,2,1)$, $(2,3,1)$, $(4,5,2)$.

Первое число в каждой тройке будем записывать в двоичном виде с использованием ровно $\lceil \log l_i \rceil$ битов, второе можно кодировать произвольным префиксным кодом чисел натурального ряда, для записи номера буквы достаточно $\lceil \log |A| \rceil$ битов.

Утверждение 4.1

Пусть слово $w \in A^*$ разделено на m подслов в соответствии со схемой Лемпела-Зива. Тогда

$$|f_{LZ}(w)| \leq m \log m + 2m \left(\log \frac{|w|}{m} + \log \log \frac{|w|}{m} + C \right).$$

Утверждение 4.1

Пусть слово $w \in A^*$ разделено на m подслов в соответствии со схемой Лемпела-Зива. Тогда

$$|f_{LZ}(w)| \leq m \log m + 2m \left(\log \frac{|w|}{m} + \log \log \frac{|w|}{m} + C \right).$$

Доказательство. Для записи тройки чисел $(d_i, l_{i+1} - l_i, j_i)$ достаточно
$\lceil \log l_i \rceil + \log(l_{i+1} - l_i) + 2 \log \log(l_{i+1} - l_i + 1) + c + \lceil \log |A| \rceil$ битов.
Применяя неравенство Йенсена к функциям $f_1(x) = \log x$,
$f_2(x) = \log \log(x + 1)$, получаем неравенства

$$|f_{LZ}(w)| \leq \sum_{i=1}^{m} (\log |w| + \log(l_{i+1} - l_i) + 2 \log \log(l_{i+1} - l_i + 1) + C') =$$

$$= m \log |w| + m \sum_{i=1}^{m} \frac{1}{m} \log(l_{i+1} - l_i) + 2m \left(\sum_{i=1}^{m} \frac{1}{m} \log \log(l_{i+1} - l_i + 1) + C' \right) \leq$$

$$\leq m \log m + 2m \left(\log \frac{|w|}{m} + \log \log \frac{|w|}{m} + C \right).$$

Утверждение 4.2

Пусть $m_n = \max\limits_{|w|=n} m(w)$, где $m(w)$ — число подслов, на которые разбивается слово $w \in A^n$ в соответствии со схемой Лемпела — Зива. Тогда $\lim\limits_{n\to\infty} \frac{m_n}{n} = 0$.

Утверждение 4.2

Пусть $m_n = \max\limits_{|w|=n} m(w)$, где $m(w)$ — число подслов, на которые разбивается слово $w \in A^n$ в соответствии со схемой Лемпела — Зива. Тогда $\lim\limits_{n\to\infty} \frac{m_n}{n} = 0$.

Доказательство. По определению схемы Лемпела — Зива, все слова, на которые разделяется слово w, различны. Если $m(w) \geq \sum\limits_{i=1}^{t} |A|^i$, то $|w| \geq \sum\limits_{i=1}^{t} i|A|^i$. Отсюда видно, что $|w| = n \geq Cm_n \log m_n$, где C — некоторая постоянная, зависящая только от мощности алфавита. Последовательность (m_n), а значит и (α_n), $\alpha_n = \log m_n$ неограниченно возрастает при $n \to \infty$. Тогда $\frac{m_n}{n} \leq \frac{1}{C \log m_n} \to 0$.

Теорема (Лемпел и Зив)

Пусть $\langle A, P \rangle$ — источник Бернулли. Тогда предельная избыточность кодирования Лемпела — Зива равняется нулю, т. е. $r(f_{LZ}, P) = 0$.

Теорема (Лемпел и Зив)

Пусть $\langle A, P\rangle$ — источник Бернулли. Тогда предельная избыточность кодирования Лемпела — Зива равняется нулю, т. е. $r(f_{LZ}, P) = 0$.

Доказательство.

Обозначим через a_0 некоторую букву, отсутствующую в алфавите A. Пусть слово $x = \sigma_1 \dots \sigma_m$ разделено на подслова σ_i в соответствии со схемой Лемпела — Зива и $\hat{x} = \hat{\sigma}_1 \dots \hat{\sigma}_m$, где $\hat{\sigma}_i = \sigma_i a_0$.

Рассмотрим множество S_m перестановок длины m. При различных $\tau \in S_m$ слова $y(\tau) = \hat{\sigma}_{\tau(1)} \dots \hat{\sigma}_{\tau(m)}$ различны, поскольку все слова σ_i получаются различными и не содержат добавочной буквы a_0.

Количество различных перестановок $y(\tau)$ не превосходит числа всевозможных различных перестановок букв в слове $\hat{x}$, т. е.

$$m! \leq \frac{(n + r_0)!}{r_0! r_1! r_2! \cdots r_k!},$$

где r_i — количество вхождений буквы a_i, $i = 0, \ldots, k$, в слово $\hat{x}$. Учитывая, что $r_0 = m$ и количество вхождений буквы a_i, $i = 1, \ldots, k$, в словах $\hat{x}$ и x совпадает, из предыдущего неравенства и утверждения 1.3 имеем неравенства

$$\frac{n^n}{r_1^{r_1} r_2^{r_2} \cdots r_k^{r_k}} \geq \frac{n!}{r_1! r_2! \cdots r_k!} \geq \frac{n!(m!)^2}{(m+n)!} \geq m! \frac{n^n m^m}{(m+n)^{m+n}}.$$

Тогда, применяя формулу Стирлинга и неравенство $\ln(1 + x) \leq x$ при $x > -1$, получаем неравенство для эмпирической энтропии

$$nH(x) = \log \frac{n^n}{r_1^{r_1} r_2^{r_2} \cdots r_k^{r^k}} \geq m \log m - m \log \frac{n}{m} - C'm,$$

где $C' > 0$ — некоторая постоянная.

Тогда из утверждения 4.1 и предыдущего неравенства имеем

$$\frac{1}{n}(|f_{LZ}(x)| - nH(x)) \leq 3\frac{m}{n}\log\frac{n}{m} + 2\frac{m}{n}\log\log\frac{n}{m} + C''\frac{m}{n} \leq$$

$$\leq 5\frac{m_n}{n}\log\frac{n}{m_n} + C''\frac{m_n}{n},$$

где число C'' зависит только от мощности алфавита, но не от распределения вероятностей P. Введём обозначение $\beta_n = 5\frac{m_n}{n}\log\frac{n}{m_n} + C''\frac{m_n}{n}$ Из утверждения 3.3 получаем неравенства

$$\frac{1}{n}R_n(f,P) = \frac{1}{n}(C_n(f,P) - H_n(P)) \leq \frac{1}{n}\sum_{|x|=n} P(x)(|f_{LZ}(x)| - nH(x)) \leq \beta_n.$$

Поскольку $\lim\limits_{t\to 0} t\ln t = 0$, из утверждения 4.2 следует, что $\lim\limits_{n\to\infty} \beta_n = 0$.

Модификация схемы LZ77, предложенная П. Бендером и Дж. Вольфом заключается в том, что после нахождения длиннейшего подходящего подслова σ_i в префиксе $x_1^{l_i-1}$ слова $x \in A^*$ нужно найти подходящее подслово σ'_i в слове $x_1^{l_i-1}$ — наиболее длинное, не считая σ_i. Вместо числа $|\sigma_i|$ нужно кодировать число $|\sigma_i| - |\sigma'_i|$. При декодировании, зная начало подслова σ_i, нетрудно найти в слове $x_1^{l_i-1}$ наиболее длинное подслово σ'_i, совпадающее с началом подслова σ_i. Тогда длина подслова σ_i легко восстанавливается из равенства

$$|\sigma_i| = |\sigma'_i| + (|\sigma_i| - |\sigma'_i|).$$

Схема кодирования, предложенная А. Лемпелем и Я. Зивом в 1978 году отличается от LZ77 тем, что на каждом шаге выбирается наиболее длинное начало суффикса $x_{l_i}^n$ слова $x \in A^*$, которое совпадает с некоторым уже выделенным подсловом σ_j, $j < i$, и к нему добавляется еще одна буква, т. е. $\sigma_{i+1} = \sigma_j a_{p_i}$. Кодом подслова σ_{i+1} будет пара чисел (j, p_i). Например, слово $a_2a_1a_2a_1a_1a_2a_1a_2a_1$ разделяется на подслова $a_2, a_1, a_2a_1, a_1a_2, a_1a_2a_1$ и кодируется последовательностью пар чисел $(0, 2), (0, 1), (1, 1), (2, 2), (4, 1)$.

Этот метод удобно рассматривать как кодирование с динамическим словарем U. Сначала словарь U состоит из всех букв алфавита. На каждом шаге алгоритма отделяем от остатка кодируемого слова наиболее длинное слово $y \in U$ и добавляем в словарь U все слова вида ya_i. Словарь U можно произвольным образом нумеровать и удалять из него слова, все продолжения которых уже имеются в словаре.
Ещё одна модификация кодирования Лемпела — Зива предложена Т. А. Велчем. Она отличается от последней схемы 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.

Рассмотрим схему разделения слова на подслова, совпадающую с первоначальной, только без перекрытий и добавления к каждому выделенному подслову последней буквы (при первом вхождении буквы в слово однобуквенное подслово выделяется), которой не было в более раннем вхождении этого подслова. Т. е. будем выбирать следующее подслово $\sigma_i = x_{l_i}^{l_{i+1}-1}$ так, чтобы слово $x_{l_i}^{l_{i+1}-1}$ было длиннейшим префиксом слова $x_{l_i}^{n}$, содержащимся как подслово в слове $x_1^{l_i-1}$. Обозначим через $l'_{LZ}(u)$ количество подслов в представлении слова $u \in A^*$ по этой схеме.

Определение

Слово u называется *максимальным суффиксом* слова $w \in A^*$, если слово w представимо в виде $w = vu$, где u является либо подсловом слова v, либо буквой, отсутствующей в слове v, причём слово u имеет максимально возможную длину. Пусть $w = = u(0)u(1)\dots u(m)$, где $u(i)$ — максимальный суффикс слова $u(0)u(1)\dots u(i)$ при любом $i = 0, \dots, m$.

Определение

Суффиксной сложностью слова w называется величина $l^*(w) = m$.

Задача 4.1

Докажите, что для любого слова $u \in A^*$ справедливо равенство $l'_{LZ}(u) = l^*(u)$.

Определение

Схемой конкатенации слова w называется такая последовательность слов $v(1), \ldots, v(m) = w$, что каждое слово $v(i)$ является буквой или конкатенацией двух, встречавшихся ранее слов $v(j_1)$ и $v(j_2)$, т. е. $v(i) = v(j_1)v(j_2)$, где $j_1, j_2 < i$.

Определение

Аддитивной сложностью слова w называется величина $l(w) = \min m$, где минимум берется по всем схемам конкатенации слова w.

Аналогичным образом можно определить аддитивную сложность $l(M)$ конечного множества $M \subset A^*$, как минимальную длину схемы конкатенации, содержащей все слова из M.

Задача 4.2

Пусть множество $T \subset A^*$ является множеством пометок вершин некоторого k-ичного дерева. Тогда $l(T) = |T| - 1$.

Задача 4.3

Для любого слова $w \in A^*$ справедливо неравенство $l^*(w) \leq l(w)$.