УДК 519.163, 519.72

Быстрый алгоритм нумерационного кодирования для основных задач теории информации

Ю. С. Медведева

Предлагается быстрый алгоритм нумерационного кодирования для основных задач теории информации, таких как:

1) кодирование двоичных слов заданной длины с заданным количеством единиц и частный случай этой задачи, когда количество единиц в слове равно количеству нулей;

2) кодирование слов с ограничением на количество подряд идущих одинаковых символов. Эта задача имеет приложение в магнитной записи и некоторых других областях;

3) кодирование элементов грассманиана и кодирование слов языков Дика.

Алгоритм является модификацией метода быстрой нумерации комбинаторных объектов, предложенного Б. Рябко. Предлагаемый нами алгоритм имеет меньшую вычислительную сложность, чем другие известные алгоритмы.

Ключевые слова: кодирование, нумерационное кодирование, теория информации.

1. Введение

В теории информации есть несколько задач создания быстрых алгоритмов кодирования и декодирования некоторых особых множеств слов.

Первая из таких задач — задача создания быстрого нумерационного кода для двоичных слов заданной длины с заданным количеством единиц.

Обозначим через S множество всех двоичных слов заданной длины, имеющих заданное количество единиц. Алгоритм нумерационного кодирования позволяет по данному слову из множества S находить его кодовое слово или номер, т. е. целое число из промежутка $[0,\, |S|-1]$. Алгоритм нумерационного декодирования позволяет по кодовому слову, т. е. целому числу из промежутка $[0,\, |S|-1]$, находить соответствующее ему слово из множества S. Особый интерес имеет частный случай этой задачи, когда количество единиц равно половине длины слова [7], [16].

Следующая задача, привлекающая внимание многих исследователей — задача создания быстрых нумерационных кодов для слов с заданным ограничением на количество подряд идущих одинаковых символов ($dklr$-последовательностей). Эта задача имеет важное приложение в магнитной и оптической записи, также как в оптической передаче данных, и была рассмотрена во множестве работ, таких как [15], [1], [3], [10] и др.

Третья задача — быстрые нумерационные коды для элементов грассманиана. Эта задача имеет приложение в сетевом кодировании [5], [14], [8], [12], [13].

Мы также рассматриваем быстрый нумерационный код для слов языков Дика. Кодируемое множество слов в этом случае — это множество сбалансированных последовательностей длины $2n$ открывающих и закрывающих скобок k типов. Необходимость быстрой нумерации и денумерации слов языков Дика возникает при работе трансляторов языков высокого уровня, для сжатия правильных последовательностей скобок и случайной генерации правильных последовательностей скобок [21], [17], [18].

В данной работе описывается модификация метода из работы [22] для быстрых алгоритмов нумерации и денумерации и предлагается основывающийся на данном методе алгоритм для названных задач. Алгоритм подробно описывается на примере кодирования слов языков Дика, также кратко описано применение этого алгоритма к кодированию двоичных слов, количество единиц в которых равно половине длины слова. Некоторые результаты, связанные с кодированием последовательностей с ограничением на количество подряд идущих одинаковых символов, опубликованы в работе автора [20]. Результаты, связанные с кодированием элементов грассманиана, опубликованы в работе [19].

2. Быстрый алгоритм нумерации на примере слов языков Дика над алфавитом $\{0, 1\}$

Словами языка Дика над алфавитом, состоящим из $2m$ букв, являются последовательности правильно вложенных скобок m типов. Рассмотрим в качестве примера все слова длины $n = 4$ языка Дика над шестью буквами, т. е. последовательности длины 4 правильно вложенных скобок трёх типов. Всего таких слов 18, см. табл. 1. Сопоставим им номера, записанные в двоичном виде, длины $\lceil \log_2 18 \rceil = 5$. В первый столбец запишем все такие слова, а во второй – их номера в двоичном виде.

Таблица 1.

Слово	Номер	Слово	Номер
(( ))	00000	[ ] [ ]	01001
( ) ( )	00001	[{ }]	01010
([ ])	00010	[ ] { }	01011
( ) [ ]	00011	{ ( ) }	01100
({ })	00100	{ } ( )	01101
( ) { }	00101	{ [ ] }	01110
[( )]	00110	{ } [ ]	01111
[ ] ( )	00111	{ { } }	10000
[[ ]]	01000	{ } { }	10001

Алгоритм нумерации ставит слову, принадлежащему языку Дика над алфавитом, состоящим из $2m$ букв длины n, последовательность из нулей и единиц, т. е. его номер. Например, для множества слов языка Дика над алфавитом 6 длины 4, расположенных в порядке, указанном в таблице, по данному слову $(\,)\{\,\}$ алгоритм должен находить его номер 00101.

В данной статье мы рассмотрим быстрый алгоритм нумерации и денумерации на примере множества слов языков Дика над алфавитом $\{0, 1\}$.

Алгоритм нумерации слов длины n языков Дика над алфавитом $2m$, основанный на методе Ковера [2], имеет сложность $O(n^2)$ битовых операций на одно нумеруемое слово, или $O(n)$ битовых операций на один символ нумеруемого слова.

Метод нумерации слов длины n языков Дика над алфавитом $2m$, предлагаемый в данной работе и основанный на подходе из работы [22], имеет сложность $O(\log n / n\, M(n \log n))$ битовых операций на один символ нумеруемого слова, где $M(n \log n)$ – время умножения или деления слов длины $n \log n$. Если использовать метод Шёнхаге – Штрассена [11], сложность

которого $n \log n \log\log n$ при умножении или делении слов длины n, то сложность рассматриваемого метода $O(\log^3 n \log\log n)$ на один символ нумеруемого слова. Если использовать метод Фюрера [4], сложность которого $n \log n 2^{O(\log^* n)}$ при умножении или делении слов длины n, то сложность рассматриваемого метода $O(\log^3 n 2^{O(\log^* n)})$ на один символ нумеруемого слова.

Обозначим D_n^{2m} множество слов языка Дика над алфавитом, состоящим из $2m$ букв длины n.

Покажем, как применяется быстрый алгоритм нумерации, предлагаемый в данной статье, для нумерации слов языка Дика длины n над алфавитом $\{0, 1\}$, т. е. слов множества D_n^2.

В качестве примера будем искать номер слова $w = 01010011 \in D_8^2$.

Описание будет удобно начать с описания нахождения номера слова w среди множества S, где S — произвольное множество слов длины n, который мы обозначим через $code_S(w)$ с помощью метода Ковера из [2].

Согласно [2], номер слова $w = x_1 \dots x_n$ из заданного множества слов S длины n, упорядоченного лексикографически, можно найти по формуле

$$code_S(w) = \sum_{i=1}^{n} \sum_{\chi < x_i} N_S(x_1 x_2 \dots x_{i-1} \chi), \tag{1}$$

где $N_S(x_1 x_2 \dots x_{i-1}\chi)$ — количество слов множества S, начинающихся с $x_1 x_2 \dots x_{i-1}\chi$.

Для того чтобы использовать эту формулу для нумерации слов множества D_n^2, нужно опеределить, чему равны значения $N_{D_n^2}(x_1 \dots x_i)$, $0 < i \le n$.

Найдем, чему равно $N_{D_8^2}(01)$, т. е. сколько слов множества D_8^2 начинаются на 01. Словами множества D_8^2, начинающимися на 01, будут являться слова, состоящие из четырёх нулей и четырёх единиц, которые начинаются на 01, кроме тех, которые не соответствуют правильным расстановкам скобок. Количество всех слов, состоящих из четырёх нулей и четырёх единиц, начинающихся на 01, легко найти: оно равно количеству всех слов, состоящих из трёх нулей и трёх единиц, т. е. $C_6^3 = 20$.

Слова, начинающиеся на 01, состоящие из четырёх нулей и четырёх единиц и не соответствующие правильным расстановкам скобок, это такие слова из четырёх нулей и четырёх единиц, для которых существует такое j, $2 < j \le 8$, что количество единиц в последовательности $x_1 x_2 \dots x_j$ превышает количество нулей в этой последовательности. Существует взаимооднозначное соответствие таких слов и всех слов, состоящих из трёх нулей и пяти единиц и начинающихся на 01. Это отображение осуществляется следующим образом. Для слова, не соответствующего правильной расстановке скобок, есть такие j, $2 < j \le 8$, что количество единиц в последовательности $x_1 x_2 \dots x_j$ превышает количество нулей в этой последовательности. Для каждого такого слова можно найти минимальное среди всех j. Можно видеть, что для такого j количество единиц в последовательности $x_1 x_2 \dots x_j$ превышает количество нулей на один символ. Заменим теперь в слове все символы после j-го на противоположные. Получаем слово из трёх нулей и пяти единиц, начинающееся на 01. Т. к. это отображение взаимооднозначное, количество слов, начинающихся на 01, состоящих из четырёх нулей и четырёх единиц и не соответствующих правильным расстановкам скобок, равно количеству всех слов, начинающихся на 01 и состоящих из трёх нулей и пяти единиц. Таких слов столько же, сколько слов, состоящих из двух нулей и четырёх единиц, т. е. $C_6^2 = 15$. Таким образом, $N_{D_8^2}(01) = C_6^3 - C_6^2 = 5$.

В общем виде, $C_{n-i}^{n/2-z} - C_{n-i}^{n/2-z-1} = (n-i)!/((n/2-z)!(n/2-i+z)!) - (n-i)!/$
$/((n/2-z-1)!\,(n/2-i+z+1)!) = (n-i)!(2z-i+1)/((n/2-z)!\,(n/2-i+z+1)!)$,

т. е.

$$N_{D_n^2}(x_1x_2\dots x_i)=\frac{(n-i)!(2z-i+1)}{(n/2-z)!\,(n/2-i+z+1)!}, \tag{2}$$

где z — количество нулей в $x_1x_2\dots x_i$, при том, что $x_1x_2\dots x_i$ может быть началом слова, соответствующего правильной расстановке скобок длины n. Если $x_1x_2\dots x_i$ не может быть началом слов, соответствующих правильной расстановке скобок длины n, то очевидно $N_{D_n^2}(x_1\dots x_i)=0$.

При применении метода Ковера используется вспомогательная таблица, в которой хранятся все возможные значения $N_S(x_1x_2\dots x_{i-1}\chi)$ или все возможные значения $\sum\chi<x_iN_S(x_1x_2\dots x_{i-1}\chi)$, $0<i\le n$. Эта таблица строится один раз и затем используется для всех последующих поисков номера слов множества S.

В случае множества D_n^2 достаточно таблицы, в которой каждой паре i, $0<i\le n$, и z, $0\le z\le i$ сопоставляется значение $N_{D_n^2}(x_1\dots x_i)=\frac{(n-i)!(2z-i+1)}{(n/2-z)!\,(n/2-i+z+1)!}$ (2). Размер такой таблицы равен $O(n^3)$.

Для получения номера слова $w\in D_n^2$ согласно (1) для каждого i, $0<i\le n$, такого, что $x_i=1$, находится значение z, равное количеству нулей в слове $x_1\dots x_{i-1}0$, затем находится с помощью таблицы соответствующее паре i и z значение $N_{D_n^2}(x_1\dots x_i)$, затем складываются все найденные значения $N_{D_n^2}(x_1\dots x_i)$.

В данном примере:

$$\begin{aligned}code_{D_8^2}(01010011)=N_{D_8^2}(00)+N_{D_8^2}(0100)+N_{D_8^2}(0101000)+\\ +N_{D_8^2}(01010010)=9+3+0+0=12.\end{aligned} \tag{3}$$

Значения $N_{D_8^2}(00)$, $N_{D_8^2}(0100)$ при этом берутся из заранее построенной таблицы. При значениях i, равных 2, 4, 6, 8, выполняется $x_i=1$. При $i=2$ слово $x_{i-1}0$ равно 00, следовательно, $z=2$, поэтому из таблицы берётся значение $N_{D_8^2}(00)$, соответствующее паре $i=2\,z=2)$ и равное $(6!\,3)/(2!\,5!)=9$. При $i=4$ слово $x_1\dots x_{i-1}0$ равно 0100, следовательно, $z=3$, поэтому по таблице находится значение $N_{D_8^2}(0100)$, соответствующее паре $i=4$, $z=3)$ и равное $(4!\,3)/(1!\,4!)=3$. Значения же $N_{D_8^2}(0101000)$ и $N_{D_8^2}(01010010)$ равны нулю, т. к. не существует слов множества D_8^2, начинающихся на 0101000 или 01010010.

Мы видим, что для такого вычисления требуется совершить максимум n операций сложения слов длин от 1 до n. Т. о., если использовать вспомогательную таблицу, то сложность вычисления номера слова по методу Ковера равно $O(n^2)$ или $O(n)$ на один символ нумеруемого слова.

Перейдём теперь к описанию нумерации слова $w=x_1\dots x_n$ заданного множества S слов длины n предлагаемым быстрым алгоритмом нумерации, затем покажем, как применяется этот алгоритм для нумерации слов множества D_n^2.

Определим величины $P(x_i|x_1\dots x_{i-1})$, $q(x_i|x_1\dots x_{i-1})$ при $0<i\le n$

$$\begin{aligned}P(x_1)=\frac{N_S(x_1)}{|S|},\ P(x_i|x_1x_2\dots x_{i-1})=\frac{N_S(x_1x_2\dots x_i)}{N_S(x_1x_2\dots x_{i-1})},\\ q(x_1)=\sum_{\chi<x_1}P(\chi),\ q(x_i|x_1\dots x_{i-1})=\sum_{\chi<x_i}P(\chi|x_1\dots x_{i-1}).\end{aligned} \tag{4}$$

Можно видеть, что по (1)

$$code_S(x_1\dots x_n)=|S|(q(x_1)+q(x_2|x_1)P(x_1)+q(x_3|x_1x_2)P(x_2|x_1)P(x_1)+\dots). \tag{5}$$

Идея метода заключается в расстановке скобок в этом выражении таким образом, что при вычислении номера большинство операций производится над короткими числами. Такой расстановкой скобок будет являться

$$code_S(x_1 \dots x_n) = |S|((q(x_1) + q(x_2|x_1)P(x_1)) + ((q(x_3|x_1x_2)+ \\ +q(x_4|x_1 \dots x_3)P(x_3|x_1x_2))P(x_2|x_1)P(x_1)) + \dots). \tag{6}$$

Определим величины ρ_b^a, λ_b^a при $0 \le a \le \log n$, $1 \le b \le n/2^a$ следующим образом:

$$\rho_b^0 = P(x_b|x_1 \dots x_{b-1}),\ \lambda_b^0 = q(x_b|x_1 \dots x_{b-1}), \\ \rho_b^a = \rho_{2b-1}^{a-1}\rho_{2b}^{a-1},\ \lambda_b^a = \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1}\lambda_{2b}^{a-1}. \tag{7}$$

Тогда $\lambda^{\log(n)} = ((q(x_1) + q(x_2|x_1)P(x_1)) + ((q(x_3|x_1x_2) + q(x_4|x_1 \dots x_3)P(x_3|x_1x_2))\cdot$
$\cdot P(x_2|x_1)P(x_1)) + \dots)$.

Отсюда и (6):

$$code_S(x_1x_2 \dots x_n) = \lambda_1^{\log n}\,|S|. \tag{8}$$

Алгоритм заключается в том, что сначала находятся значения $P(x_i|x_1 \dots x_{i-1})$, $q(x_i|x_1 \dots x_{i-1})$ при $0 < i \le n$, определённые в (4), затем находится $\lambda_1^{\log n}$ последовательным вычислением значений ρ_b^a, λ_b^a при $0 \le a \le \log n$, $1 \le b \le n/2^a$ формулам (7), затем находится искомый номер $code_S(w)$ по формуле (8), при этом значение $|S|$ находится до начала нумерации и используется затем при всех последующих нумерациях.

Покажем, как применить быстрый алгоритм нумерации для нумерации слов множества D_n^2.

До начала нумерации необходимо вычислить мощность D_n^2. Мощность такого множества равна $n/2$-ому числу Каталана [23],

$$|D_n^2| = C_n = C_n^{n/2} - C_n^{n/2-1}. \tag{9}$$

В нашем примере $|D_8^2| = C_8^4 - C_8^3 = 14$.

Можно видеть из (2) и определения (4), что для множества D_n^2 значения $P(x_i|x_1 \dots x_{i-1})$ $(0 < i \le n)$ находятся следующим образом:

$$P(x_i|x_1x_2 \dots x_{i-1}) = \frac{(n-i)!(2z-i+1)}{(n/2-z)!\,(n/2-i+z+1)!} : \\ : \frac{(n-i+1)!(2z-i)}{(n/2-z+1)!\,(n/2-i+z+1)!} = \frac{(2z-i+1)(n/2-z+1)}{(2z-i)(n-i+1)} \tag{10}$$

при $x_i = 0$,

$$P(x_i|x_1x_2 \dots x_{i-1}) = \frac{(n-i)!(2z-i+1)}{(n/2-z)!\,(n/2-i+z+1)!} : \\ : \frac{(n-i+1)!(2z-i+2)}{(n/2-z)!\,(n/2-i+z+2)!} = \frac{(2z-i+1)(n/2-i+z+2)}{(n-i+1)(2z-i+2)} \tag{11}$$

при $x_i = 1$.

Таким образом, для нашего примера находим по формулам (10) и (11) значения $P(x_1) = \rho_1^0$, $P(x_2|x_1) = \rho_2^0, \ldots, P(x_8|x_1x_2\ldots x_7) = \rho_8^0$, $q(x_1) = \lambda_1^0$, $q(x_2) = \lambda_2^0, \ldots, q(x_8) = \lambda_8^0$,

$$\begin{gathered}
P(x_1) = 1,\ P(x_2|x_1) = P(1|0) = \frac{5}{14},\\
P(x_3|x_1x_2) = P(0|01) = \frac{6}{6},\ P(x_4|x_1x_2x_3) = P(1|010) = \frac{4}{10},\\
P(x_5|x_1\ldots x_4) = P(0|0101) = \frac{4}{4},\ P(x_6|x_1\ldots x_5) = P(0|01010) = \frac{3}{6},\\
P(x_7|x_1\ldots x_6) = P(1|010100) = \frac{6}{6},\ P(x_8|x_1\ldots x_7) = P(1|0101001) = \frac{2}{2},\\
q(x_1) = q(0) = 0,\ q(x_2|x_1) = q(1|0) = \frac{9}{14},\\
q(x_3|x_1x_2) = q(0|01) = 0,\ q(x_4|x_1x_2x_3) = q(1|010) = \frac{6}{10},\\
q(x_5|x_1\ldots x_4) = q(0|0101) = 0,\ q(x_6|x_1\ldots x_5) = q(0|01010) = 0,\\
q(x_7|x_1\ldots x_6) = q(1|010100) = 0,\ q(x_8|x_1\ldots x_7) = q(1|0101001) = 0.
\end{gathered} \tag{12}$$

Соответственно, по (7)

$$\begin{gathered}
\rho_1^0 = 1,\ \rho_2^0 = \frac{5}{14},\ \rho_3^0 = 1,\ \rho_4^0 = \frac{2}{5},\ \rho_5^0 = 1,\ \rho_6^0 = \frac{1}{2},\ \rho_7^0 = 1,\ \rho_8^0 = 1,\\
\lambda_1^0 = 0,\ \lambda_2^0 = \frac{9}{14},\ \lambda_3^0 = 0,\ \lambda_4^0 = \frac{3}{5}, \lambda_5^0 = 0,\ \lambda_6^0 = 0,\ \lambda_7^0 = 0,\ \lambda_8^0 = 0.
\end{gathered} \tag{13}$$

Затем по (7) вычисляем

$$\begin{gathered}
\rho_1^1 = \frac{5}{14},\ \rho_2^1 = \frac{2}{5},\ \rho_3^1 = \frac{1}{2},\ \rho_4^1 = 1,\ \lambda_1^1 = \frac{9}{14},\ \lambda_2^0 = \frac{3}{5},\ \lambda_3^0 = 0,\ \lambda_4^0 = 0,\\
\rho_1^2 = \frac{1}{7},\ \rho_2^1 = \frac{1}{2},\ \lambda_1^2 = \frac{6}{7},\ \lambda_2^2 = 0,\ \rho_1^3 = \frac{1}{14},\ \lambda_1^3 = \frac{6}{7}.
\end{gathered} \tag{14}$$

По (8) $code_{D_8^2}(01010011) = \lambda_1^3 \cdot |D_8^2| = \frac{6}{7} \cdot 14 = 12$. Таким образом, мы получили $code_{D_8^2}(w)$, номер слова $01010011 \in D_8^2$.

Для исследования свойств алгоритма нумерации слов, принадлежащих множеству S, мы определим несколько величин.

Обозначим через q_{max} максимальный знаменатель дробей $N_S(x_1x_2\ldots x_i)/N_S(x_1x_2\ldots x_{i-1})$, $x_1\ldots x_n \in S$, $i = 1, \ldots, n$.

Из (7) получаем, что знаменатели рациональных дробей λ_b^a и ρ_b^a не превосходят $q_{max}^{2^b}$ при всех $b = 1, 2, \ldots, n/2^a$ и следовательно,

$$\rho_a^b \geq 1/q_{max}^{2^a}. \tag{15}$$

Свойства предложенного метода характеризует следующая теорема.

Теорема 1. *1) Скорость кодирования (то есть время кодирования на букву, измеряемое числом операций над однобитовыми словами) слов множества S длины n равна*

$$O(\log n M(n \log q_{max})/n), \tag{16}$$

где $M(n)$ время умножения двух слов длины n.

Следствие 1. При использовании алгоритма быстрого умножения Шёнхаге – Штрассена, для которого $M(n) = O(n \log n \log\log n)$, скорость нумерации равна $O(\log n \log q_{max} \log(n \log q_{max}) \log\log(n \log q_{max}))$.

Следствие 2. При использовании алгоритма быстрого умножения Фюрера, для которого $M(n) = O(n \log n 2^{O(\log^* n)})$*, скорость нумерации равна* $O(\log n \log(n \log q_{max}) 2^{O(\log^*(n \log q_{max}))})$.

2) Объём памяти в битах, используемый при кодировании слов множества S *длины* n*, не превосходит*

$$O((n \log q_{max}) \log n). \tag{17}$$

Доказательство. 1) Из определения q_{max} и (4) мы видим, что для записи каждой дроби $P(x_i|x_1 \dots x_{i-1})$ и $q(x_i|x_1 \dots x_{i-1})$, $i = 1, \dots, n$ достаточно $2 \log q_{max}$ бит, $\log q_{max}$ для числителя и столько же для знаменателя. Поэтому вычисление одной величины ρ_b^1 или λ_b^1 в соответствии с (7) при любом $b = 1, \dots, n/2$ потребует одной операции умножения чисел, длина которых не превосходит $\log q_{max}$ бит, а общее число операций умножения при вычислении всех λ_b^1, ρ_b^1, $b = 1, \dots, n/2$, равно $5n/2$. При вычислении λ_b^1 используется обычное равенство $a/b + c/d = \frac{(ad+bc)}{bd}$, требующее три умножения. В результате будут получены дроби, у которых для записи числителя и знаменателя требуется не более $2 \log q_{max}$ бит. Аналогично, для вычисления λ_b^2, ρ_b^2, $b = 1, \dots, n/4$, требуется $5n/4$ операций умножения над числами длины $2 \log q_{max}$ и так далее, для вычисления λ_b^a, ρ_b^a, $b = 1, \dots, n/2^a$, требуется $5n/2^a$ операций над числами длины $2^{a-1} \log q_{max}$ бит. Обозначим через $M(a)$ время умножения двух слов длины a. Получаем, что общее время вычислений λ_b^a, ρ_b^a по формулам (7) требует

$$\begin{aligned} \frac{5n}{2} M(\log q_{max}) + \frac{5n}{4} M(2 \log q_{max}) + \dots + \\ + \frac{5n}{2^a} M(2^a \log q_{max}) + \dots + \\ 5M(n \log q_{max}). \end{aligned} \tag{18}$$

Обозначим через $M^*(n)$ время умножения двух чисел длины n, делённое на длину этих чисел: $M^*(n) = \frac{M(n)}{n}$.

Тогда время вычислений λ_b^a, ρ_b^a по формулам (7) требует

$$\begin{aligned} \frac{5n}{2} \log q_{max} M^*(\log q_{max}) + \frac{5n}{4} 2 \log q_{max} M(2 \log q_{max}) + \dots + \\ + \frac{5n}{2^a} 2^a \log q_{max} M^*(2^a \log q_{max}) + \dots + \\ 5n \log q_{max} M^*(n \log q_{max}). \end{aligned} \tag{19}$$

В этой сумме $\log n$ слагаемых, каждое из которых не превышает $5n \log q_{max} M^*(n \log q_{max}) = 5M(n \log q_{max})$, следовательно сумма есть $O(\log n M(n \log q_{max}))$. Отсюда число операций на букву слова $O(\log n M(n \log q_{max})/n)$.

2) Оценим необходимый для осуществления кодирования объем памяти. Заметим, что при вычислении λ_b^a и ρ_b^a, $a > 1$, $b = 1, \dots, n/2^a$, используются только величины λ_b^{a-1} и ρ_b^{a-1}, $b = 1, \dots, n/2^{a-1}$. Поэтому при кодировании достаточно иметь память для хранения двух наборов $\{\lambda_b^a, \rho_b^a\}$, $b = 1, \dots, 2^a$, и $\{\lambda_b^a, \rho_b^a\}$, $b = 1, \dots, 2^{a+1}$, $a = 1, \dots, \log n - 1$. Длина каждой дроби λ_b^a, ρ_b^a не превосходит $2^{a+1} \log q_{max}$ бит. Отсюда получаем второе утверждение теоремы. □

Из (10) и (11) следует, что для слов множества D_n^2

$$q_{max} = n^2. \tag{20}$$

Опишем свойства алгоритма нумерации слов языков Дика из одного вида скобок.

Теорема 2. *1)Скорость кодирования слова длины n языка Дика над алфавитом мощности 2, т. е. время, требуемое для кодирования одной буквы, равна $O(\log nM(n\log n)/n)$ битовых операций, где $M(n)$ – время умножения двух слов длины n.*

2)Объём памяти, требуемый для кодирования слова длины n языка Дика над алфавитом мощности 2 равен $O(n\log n)$ бит.

Следствие 1. При использовании алгоритма быстрого умножения Шёнхаге – Штрассена, для которого $M(n) = O(n\log n\log\log n)$, скорость нумерации равна $O(\log^3 n\log\log n)$.

Следствие 2. При использовании алгоритма быстрого умножения Фюрера, для которого $M(n) = O(n\log n2^{O(\log^* n)})$, скорость нумерации равна $O(\log^3 n2^{O(\log^* n)})$.

Доказательство. Следует из теоремы 1 и (20). □

3. Быстрый алгоритм денумерации на примере слов языков Дика над алфавитом $\{0,1\}$

Алгоритм денумерации слов языков Дика над алфавитом $\{0,1\}$, т. е. слов множества D_n^2, позволяет найти по данному номеру слова $code_{D_n^2}(w)$ слово $w \in D_n^2$.

Начнём с описания общего алгоритма денумерации слов длины n любого заданного множества S, т. е. поиску слова $w = x_1\dots x_n \in S$ по его номеру $code_S(w)$.

Для описания алгоритма введём вспомогательные функции для верхней и нижней оценки λ_b^a. Пусть p/q — рациональное число, представленное как пара целых положительных чисел p, q, $p \le q$, и пусть $t > 1$ — целое число. Положим $l = \lfloor\log q\rfloor$. Пусть $(q_lq_{l-1}\dots q_0)$ и $(p_lp_{l-1}\dots p_0)$ — двоичные представления чисел q и p. Тогда определим $\phi_t^+\left(\frac{p}{q}\right)$ и $\phi_t^-\left(\frac{p}{q}\right)$ следующим образом:

$$\phi_t^+\left(\frac{p}{q}\right) = \left(\sum_{i=l-t}^{l} p_i2^i + 2^{l-t}\right) / \sum_{i=l-t}^{l} q_i2^i,\ \phi_t^-\left(\frac{p}{q}\right) = \sum_{i=l-t}^{l} p_i2^i / \left(\sum_{i=l-t}^{l} q_i2^i + 2^{l-t}\right). \tag{21}$$

Если $l - t < 0$, домножаем числитель и знаменатель полученной дроби на $2^{-(l-t)}$.

Например, $\phi_3^+(9/17) = 5/8$, $\phi_3^-(9/17) = 4/9$.

Докажем утверждение, которое понадобится в дальнейшем.

Лемма 1. Пусть p, q, t — целые положительные числа и $0 < p \le q$ и $t \ge 3$. Тогда

$$0 \le \phi_t^+(p/q) - p/q < 2^{3-t},\quad 0 \le p/q - \phi_t^-(p/q) < 2^{3-t}. \tag{22}$$

Доказательство. Легко видеть, что $1/(1-x) < 1+2x$ при $x < 1/2$. Это неравенство сразу следует из формулы для суммы геометрической прогрессии. Отсюда оценка первого неравенства в (22) следует из цепочки неравенств

$$\begin{aligned}\phi_t^+\left(\frac{p}{q}\right) < \frac{p+2^{l-t}}{q-2^{l-t}} < \frac{p}{q}\left(1+\frac{2^{l-t}}{p}\right)\left(1+2\cdot\frac{2^{l-t}}{q}\right) = \\ = \frac{p}{q} + \frac{2^{l-t}}{q} + \frac{p\cdot 2\cdot 2^{l-t}}{q^2} + \frac{2^{l-t}\cdot 2\cdot 2^{l-t}}{q^2} < \\ < \frac{p}{q} + 2^{-t} + 2^{2-t} + 2^{1-2t} < \frac{p}{q} + 2^{3-t}\end{aligned} \tag{23}$$

(здесь мы использовали очевидные соотношения $2^l \le q$, $p < 2^{l+1}$). Второе неравенство в (22) получается точно так же. □

Согласно данному ранее определению, для заданного множества слов S q_{max} равно максимальному знаменателю чисел $N_S(x_1 \dots x_{i+1})/N_S(x_1 \dots x_i)$, $x_1 \dots x_n \in S$, $1 \le i \le n-1$.

Из (7) получаем, что знаменатели рациональных дробей λ_b^a и ρ_b^a не превосходят $q_{max}^{2^a}$ при всех $b = 1, 2, \dots, n/2^a$ и, следовательно,

$$\rho_b^a \ge 1/q_{max}^{2^a}. \tag{24}$$

Из (10) и (11) следует, что для множества D_2^n $q_{max} = n^2$.

Первый шаг поиска слова v из множества S по данному номеру $code_S(v)$ заключается в вычислении оценок $\lambda^+(\log n, 1)$, $\lambda^-(\log n, 1)$ по формулам:

$$\begin{aligned} \lambda^+(\log n, 1) &= \phi^+_{4n\lceil \log q_{max}\rceil+4}\Big(\frac{code_S(v)}{|S|}\Big), \\ \lambda^-(\log n, 1) &= \phi^-_{4n\lceil \log q_{max}\rceil+4}\Big(\frac{code_S(v)}{|S|}\Big). \end{aligned} \tag{25}$$

Процедура 1. Опишем рекурсивную процедуру получения по известным префиксу $x_1 \dots x_{2^a(b-1)}$ и оценкам $\lambda^+(a, b)$, $\lambda^-(a, b)$ $(0 < a \le \log n,\ 1 \le b \le n/2^a)$ подслова $x_{2^a(b-1)+1} \dots x_{2^a b}$ и значений λ_b^a, ρ_b^a или такой пары слов, что одно из них равно $x_{2^a(b-1)+1} \dots x_{2^a b}$, и соответствующих им пар предположительных значений λ_b^a, ρ_b^a. Причём, если $b = n/2^a$, то выполняя эту процедуру, можно однозначно определить подслово $x_{n-2^a+1} \dots x_n$.

Вычисляем оценки $\lambda^+(a-1,\ 2b-1)$, $\lambda^-(a-1,\ 2b-1)$ по формулам:

$$\begin{aligned} \lambda^+(a-1,\ 2b-1) &= \phi^+_{2^{a+1}\lceil \log q_{max}\rceil+4}(\lambda^+(a,\ b)), \\ \lambda^-(a-1,\ 2b-1) &= \phi^-_{2^{a+1}\lceil \log q_{max}\rceil+4}(\lambda^-(a,\ b)). \end{aligned} \tag{26}$$

Если $a > 1$, то выполняем процедуру 1 получения по префиксу $x_1 \dots x_{2^{a-1}(2b-2)}$ и этим оценкам подслова $x_{2^{a-1}(2b-2)+1} \dots x_{2^{a-1}(2b-1)}$ и значений λ_{2b-1}^{a-1}, ρ_{2b-1}^{a-1} или пары слов, одно из которых равно $x_{2^{a-1}(2b-2)+1} \dots x_{2^{a-1}(2b-1)}$, и соответствующих им пары предположительных λ_{2b-1}^{a-1}, ρ_{2b-1}^{a-1}. Обозначим лексикографически меньшее слово из пары слов через $x'_{2^{a-1}(2b-2)+1} \dots x'_{2^{a-1}(2b-1)}$, большее – через $x''_{2^{a-1}2(b-2)+1} \dots x''_{2^{a-1}(2b-1)}$, а соответствующие им предположительные λ_{2b-1}^{a-1}, ρ_{2b-1}^{a-1} – через λ'^{a-1}_{2b-1}, λ''^{a-1}_{2b-1}, ρ'^{a-1}_{2b-1} и ρ''^{a-1}_{2b-1}.

Если $a = 1$, то выполняем процедуру 2 получения по префиксу $x_1 \dots x_{2^{a-1}(2b-2)}$ и оценкам $\lambda^+(a-1,\ 2b-1)$, $\lambda^-(a-1,\ 2b-1)$ буквы x_{2b-1} и пары λ_{2b-1}^0, ρ_{2b-1}^a или пары букв x'_{2b-1}, x''_{2b-1} и пар значений λ'^0_{2b-1}, ρ'^0_{2b-1}, λ''^0_{2b-1}, ρ''^0_{2b-1}.

Таким образом, на данном этапе выполнения процедуры мы получили подслово $x_1 \dots x_{2^{a-1}(2b-2)}$ и значения λ_{2b-1}^{a-1} и ρ_{2b-1}^{a-1} или слова $x'_1 \dots x'_{2^{a-1}(2b-2)}$, $x'_1 \dots x''_{2^{a-1}(2b-2)}$ и значения λ'^{a-1}_{2b-1}, ρ'^{a-1}_{2b-1}, λ''^{a-1}_{2b-1} и ρ''^{a-1}_{2b-1}. Во втором случае проверяем, выполняются ли следующие неравенства:

$$\begin{aligned} \lambda'^{a-1}_{2b-1} + \rho'^{a-1}_{2b-1} &\le \lambda^-(a,\ b), \\ \lambda''^{a-1}_{2b-1} &> \lambda^+(a,\ b). \end{aligned} \tag{27}$$

Если выполняется первое неравенство, то $x_1 \dots x_{2^{a-1}(2b-2)} = x''_1 \dots x''_{2^{a-1}(2b-2)}$, $\lambda_{2b-1}^{a-1} = \lambda''^{a-1}_{2b-1}$, $\rho_{2b-1}^{a-1} = \rho''^{a-1}_{2b-1}$. Если выполняется второе неравенство, то $x_1 \dots x_{2^{a-1}(2b-2)} = x'_1 \dots x'_{2^{a-1}(2b-2)}$, $\lambda_{2b-1}^{a-1} = \lambda'^{a-1}_{2b-1}$, $\rho_{2b-1}^{a-1} = \rho'^{a-1}_{2b-1}$.

Если в результате сравнений (27) или без них нам на этом этапе вычислений известны точные значения λ_{2b-1}^{a-1} и ρ_{2b-1}^{a-1}, вычисляем оценки $\lambda^+(a-1,\,2b)$, $\lambda^-(a-1,\,2b)$ по формулам:

$$\begin{aligned}\lambda^+(a-1,\,2b) &= \phi^+_{2^{a+1}\lceil\log q_{max}\rceil+4}\Big(\frac{\lambda^+(a,\,b)-\lambda_{2b-1}^{a-1}}{\rho_{2b-1}^{a-1}}\Big),\\ \lambda^-(a-1,\,2b) &= \phi^-_{2^{a+1}\lceil\log q_{max}\rceil+4}\Big(\frac{\lambda^-(a,\,b)-\lambda_{2b-1}^{a-1}}{\rho_{2b-1}^{a-1}}\Big).\end{aligned} \tag{28}$$

Применяем к ним рекурсивную процедуру, если $a>1$ и находим подслово $x_{2^{a-1}(2b-1)+1}\dots x_{2^{a-1}2b}$ и значения λ_{2b}^{a-1}, ρ_{2b}^{a-1} (или, в случае, если $b=n/2^a$, только подслово $x_{2^{a-1}(2b-1)+1}\dots x_{2^{a-1}2b}$) или пару подслов, такую, что одно из этой пары равно $x_{2^{a-1}(2b-1)+1}\dots x_{2^{a-1}2b}$ и соответствующие пары предположительных значений λ_{2b}^{a-1}, ρ_{2b}^{a-1}. Если же $a=1$, выполняем процедуру 2 получения по префиксу $x_1\dots x_{2^{a-1}(2b-1)}$ и оценкам $\lambda^+(a-1,\,2b)$, $\lambda^-(a-1,\,2b)$ буквы x_{2b} и пары λ_{2b}^0, ρ_{2b}^a или пары букв x'_{2b}, x''_{2b} и пар значений λ'^0_{2b}, ρ'^0_{2b}, λ''^0_{2b}, ρ''^0_{2b}. При этом если $2b=n$, то мы однозначно находим x_{2b}.

Если ни одно из неравенств (27) не выполняется, при этом $b\neq n/2^a$, то находим последовательно буквы x'_i, $2^{a-1}(2b-1)+1\le i\le 2^{a-1}2b$, такие, что

$$\begin{aligned}q(x'_i|x_1\dots x_{2^a(b-1)}x'_{2^a(b-1)+1}\dots x'_{i-1})+P(x'_t|x_1\dots x_{2^a(b-1)}x'_{2^a(b-1)+1}\dots x'_{i-1})=1,\\ P(x'_i|x_1\dots x_{2^a(b-1)}x'_{2^a(b-1)+1}\dots x'_{i-1})>0\end{aligned} \tag{29}$$

и находим последовательно буквы x''_i, $2^{a-1}(2b-1)+1\le i\le 2^{a-1}2b$, такие, что

$$\begin{aligned}q(x''_i|x_1\dots x_{2^a(b-1)}x''_{2^a(b-1)+1}\dots x''_{i-1})=0,\\ P(x''_i|x_1\dots x_{2^a(b-1)}x''_{2^a(b-1)+1}\dots x''_{i-1})>0.\end{aligned} \tag{30}$$

Считаем рекурсивно по формулам, аналогичным (7), пару чисел λ'^{a-1}_{2b}, ρ'^{a-1}_{2b} (в формулах используется префикс $x_1\dots x_{2^a(b-1)}x'_{2^a(b-1)+1}\dots x'_{2^ab}$) и пару чисел λ''^{a-1}_{2b}, ρ''^{a-1}_{2b} (в формулах используется префикс $x_1\dots x_{2^a(b-1)}x''_{2^a(b-1)+1}\dots x''_{2^ab}$).

Если ни одно из неравенств (27) не выполняется и при этом $b=n/2^a$, это означает, что $x_{2^a(b-1)+1}\dots x_{2^{a-1}}$ найдено и равно $x''_{2^a(b-1)+1}\dots x''_{2^{a-1}}$. Находим последовательно буквы x_i, $2^{a-1}(2b-1)+1\le i\le 2^{a-1}2b$, такие, что

$$\begin{aligned}q(x_i|x_1\dots x_{i-1})=0,\\ P(x_i|x_1\dots x_{i-1})>0.\end{aligned} \tag{31}$$

Таким образом, на данном этапе выполнения процедуры у нас возможно несколько случаев.

В первом случае нам известно подслово $x_{2^{a-1}(2b-2)+1}\dots x_{2^{a-1}(2b-1)}$ и подслово $x_{2^{a-1}(2b-1)+1}\dots x_{2^{a-1}2b}$, что означает, что нам известно подслово $x_{2^a(b-1)+1}\dots x_{2^ab}$. Если $b\neq n/2^a$, вычисляем значения λ_b^a и ρ_b^a по формулам

$$\begin{aligned}\lambda_b^a &= \lambda_{2b-1}^{a-1}+\rho_{2b-1}^{a-1}\cdot\lambda_{2b}^{a-1},\\ \rho_b^a &= \rho_{2b-1}^{a-1}\cdot\rho_{2b}^{a-1}.\end{aligned} \tag{32}$$

Во втором случае нам известно подслово $x_{2^{a-1}(2b-2)+1}\dots x_{2^{a-1}(2b-1)}$ и пара слов $x'_{2^{a-1}(2b-1)+1}\dots x'_{2^{a-1}2b}$, $x''_{2^{a-1}(2b-1)+1}\dots x''_{2^{a-1}2b}$. Тогда слово $x'_{2^a(b-1)+1}\dots x'_{2^ab}$ считаем равным

конкатенации $x_{2^{a-1}(2b-2)+1}\dots x_{2^{a-1}(2b-1)}$ и $x'_{2^{a-1}(2b-1)+1}\dots x'_{2^{a-1}2b}$, слово $x''_{2^a(b-1)+1}\dots x''_{2^ab}$ считаем равным конкатенации $x_{2^{a-1}(2b-2)+1}\dots x_{2^{a-1}(2b-1)}$ и $x''_{2^{a-1}(2b-1)+1}\dots x''_{2^{a-1}2b}$. Вычисляем значения λ'^a_b, ρ'^a_b, λ''^a_b, ρ''^a_b по формулам

$$\begin{aligned}\lambda'^a_b = \lambda^{a-1}_{2b-1} + \rho^{a-1}_{2b-1}\cdot\lambda'^{a-1}_{2b},\ \rho'^a_b = \rho^{a-1}_{2b-1}\cdot\rho'^{a-1}_{2b},\\ \lambda''^a_b = \lambda^{a-1}_{2b-1} + \rho^{a-1}_{2b-1}\cdot\lambda''^{a-1}_{2b},\ \rho''^a_b = \rho^{a-1}_{2b-1}\cdot\rho''^{a-1}_{2b}.\end{aligned} \tag{33}$$

В третьем случае нам известны пары следующих слов: $x'_{2^{a-1}(2b-2)+1}\dots x'_{2^{a-1}(2b-1)}$, $x''_{2^{a-1}(2b-2)+1}\dots x''_{2^{a-1}(2b-1)}$, $x'_{2^{a-1}(2b-1)+1}\dots x'_{2^{a-1}2b}$, $x''_{2^{a-1}(2b-1)+1}\dots x''_{2^{a-1}2b}$. Слово $x'_{2^a(b-1)+1}\dots x'_{2^ab}$ считаем равным конкатенации $x'_{2^{a-1}(2b-2)+1}\dots x'_{2^{a-1}(2b-1)}$ и $x'_{2^{a-1}(2b-1)+1}\dots x'_{2^{a-1}2b}$, а слово $x''_{2^a(b-1)+1}\dots x''_{2^ab}$ равным конкатенации $x''_{2^{a-1}(2b-2)+1}\dots x''_{2^{a-1}(2b-1)}$ и $x''_{2^{a-1}(2b-1)+1}\dots x''_{2^{a-1}2b}$. Вычисляем значения λ'^a_b, ρ'^a_b, λ''^a_b, ρ''^a_b по формулам

$$\begin{aligned}\lambda'^a_b = \lambda'^{a-1}_{2b-1} + \rho'^{a-1}_{2b-1}\cdot\lambda'^{a-1}_{2b},\ \rho'^a_b = \rho'^{a-1}_{2b-1}\cdot\rho'^{a-1}_{2b},\\ \lambda''^a_b = \lambda''^{a-1}_{2b-1} + \rho''^{a-1}_{2b-1}\cdot\lambda''^{a-1}_{2b},\ \rho''^a_b = \rho''^{a-1}_{2b-1}\cdot\rho''^{a-1}_{2b}.\end{aligned} \tag{34}$$

Процедура 2. Опишем процедуру получения по оценкам $\lambda^+(0,b)$, $\lambda^-(0,b)$ и префиксу $x_1\dots x_{b-1}$ буквы x_b и соответствующей ей пары значений λ^0_b, ρ^0_b или пары букв x'_b, x''_b и соответствующих им пар значений λ'^0_b, ρ'^0_b, λ''^0_b и ρ''^0_b.

Находим такие буквы χ, $\chi\in\{0,1\}$ для которых одновременно выполнялись бы условия

$$\begin{aligned}P(\chi|x_1\dots x_{b-1}) > 0,\\ \lambda^+(0,b) \ge q(\chi|x_1\dots x_{b-1}),\\ \lambda^-(0,b) < q(\chi|x_1\dots x_{b-1}) + P(\chi|x_1\dots x_{b-1}).\end{aligned} \tag{35}$$

Если существует одно такое χ, то $x_b=\chi$. Если существует два таких χ и $b<n$, считаем x'_b равным меньшему такому χ, x''_b большему такому χ. Если $b=n$, то x_b равно большему такому χ. Если x_b найдено однозначно, вычисляем значения λ^0_b и ρ^0_b по формулам (7). Если найдена пара x'_b и x''_b, вычисляем значения λ'^0_b, ρ'^0_b, λ''^0_b и ρ''^0_b по следующим формулам:

$$\begin{aligned}\lambda'^0_b = q(x'_b|x_1\dots x_{b-1}),\\ \lambda''^0_b = q(x''_b|x_1\dots x_{b-1}),\\ \rho'^0_b = P(x'_b|x_1\dots x_{b-1}),\\ \rho''^0_b = P(x''_b|x_1\dots x_{b-1}).\end{aligned} \tag{36}$$

Применяя процедуру 1 для $a=\log n$, $b=1$ находим по оценкам $\lambda^+(\log n,\,1)$, $\lambda^-(\log n,\,1)$ слово $x_1\dots x_n$. (При этом считаем, что нам известен префикс нулевой длины.)

Покажем, как применяется описанный алгоритм денумерации для денумерации слов множества D^2_n.

Сделаем это на примере поиска слова $w\in D^2_8$ по данному номеру $code_{D^2_8}(w)=12$.

По (20) $q_{max}=n^2=64$.

Находим $\lambda^+(3,\,1)$, $\lambda^-(3,\,1)$ по формулам (25):
$\lambda^+(3,1)=\phi^+_{196}(12/14)=(12\cdot 2^{193}+1)/(14\cdot 2^{193})$,
$\lambda^-(3,1)=12\cdot 2^{193}/(14\cdot 2^{193}+1)$.

Находим $\lambda^+(2,\,1)$, $\lambda^-(2,\,1)$, а затем $\lambda^+(1,\,1)$, $\lambda^-(1,\,1)$, $\lambda^+(0,\,1)$, $\lambda^-(0,\,1)$ по формулам (26):

$\lambda^+(2,1) = \phi^+_{100}((12\cdot 2^{193}+1)/(14\cdot 2^{193})) = (12\cdot 2^{97}+1)/(14\cdot 2^{97})$,
$\lambda^-(2,1) = 12\cdot 2^{97}/(14\cdot 2^{97}+1)$,
$\lambda^+(1,1) = \phi^+_{52}((12\cdot 2^{97}+1)/(14\cdot 2^{97})) = (12\cdot 2^{49}+1)/(14\cdot 2^{49})$,
$\lambda^-(1,1) = 12\cdot 2^{49}/(14\cdot 2^{49}+1)$,
$\lambda^+(0,1) = \phi^+_{28}((12\cdot 2^{49}+1)/(14\cdot 2^{49})) = (12\cdot 2^{25}+1)/(14\cdot 2^{25})$,
$\lambda^-(0,1) = 12\cdot 2^{25}/(14\cdot 2^{25}+1)$.

Находим χ такие, для которых выполняются неравенства (35), т. е. $\lambda^+(0,1) \geq q(\chi)$, $\lambda^-(0,1) < q(\chi)+P(\chi)$, $P(\chi) > 0$. Единственное χ, которое удовлетворяет этим неравенствам, это ноль, можно видеть, что $\lambda^+(0,1) > q(0) = 0$, $\lambda^-(0,1) < q(0) + P(0) = 1$, $P(0) = 1 > 0$, в то же время для единицы эти неравенства не выполняются, т. к. $P(1) = 0$. Поэтому $x_1 = 0$.

Находим λ^0_1 и ρ^0_1 по формулам (7): $\lambda^0_1 = q(0) = 0$, $\rho^0_1 = P(0) = 1$.

Находим $\lambda^+(0,2)$, $\lambda^-(0,2)$ по формулам (28):
$\lambda^+(0,2) = \phi^+_{28}(((12\cdot 2^{49}+1)/(14\cdot 2^{49}) - 0)/1) = (12\cdot 2^{25}+1)/(14\cdot 2^{25})$,
$\lambda^-(0,2) = 12\cdot 2^{25}/(14\cdot 2^{25}+1)$.

Находим χ такие, для которых выполняются неравенства (35), т. е. $\lambda^+(0,2) \geq q(\chi|0)$, $\lambda^-(0,2) < q(\chi|0) + P(\chi|0)$, $P(\chi|0) > 0$. Неравенства выполняются для $\chi = 1$, т. к. $\lambda^+(0,2) > q(1|0) = 5/14$, $\lambda^-(0,2) < q(1|0) + P(1|0) = 1$, $P(1|0) = 5/14 > 0$, неравенства не выполняются для $\chi = 0$, т. к. $\lambda^-(0,2) > q(0|0) + P(0|0) = 9/14$. Поэтому $x_2 = 1$.

Находим λ^0_2 и ρ^0_2, а затем λ^1_1 и ρ^1_1 по формулам (7):
$\lambda^0_2 = q(1|0) = 9/14$,
$\rho^0_2 = P(1|0) = 5/14$,
$\lambda^1_1 = \lambda^0_1 + \lambda^0_2\cdot\rho^0_1 = 9/14$,
$\rho^1_1 = \rho^0_1\cdot\rho^0_2 = 5/14$.

Находим $\lambda^+(1,2)$, $\lambda^-(1,2)$ по формулам (28):
$\lambda^+(1,2) = (3\cdot 2^{50}+1)/(5\cdot 2^{50})$,
$\lambda^-(1,2) = 3\cdot 2^{50}/(5\cdot 2^{50}+1)$.

Находим $\lambda^+(0,3)$, $\lambda^-(0,3)$, по формулам (26):
$\lambda^+(0,3) = (3\cdot 2^{26}+1)/(5\cdot 2^{26})$,
$\lambda^-(0,3) = 3\cdot 2^{26}/(5\cdot 2^{26}+1)$.

Находим χ такие, для которых выполняются неравенства (35), т. е. $\lambda^+(0,3) \geq q(\chi|01)$, $\lambda^-(0,3) < q(\chi|01) + P(\chi|01)$, $P(\chi|01) > 0$. Неравенства выполняются для $\chi = 0$, т. к. $\lambda^+(0,3) > q(0|01) = 0$, $\lambda^-(0,3) < q(0|01) + P(0|01) = 1$, $P(0|01) = 1 > 0$, неравенства не выполняются для $\chi = 1$, т. к. $P(1|01) = 0$. Поэтому $x_3 = 0$.

Находим λ^0_3 и ρ^0_3 по формулам (7): $\lambda^0_3 = q(0|01) = 0$, $\rho^0_3 = P(0|01) = 1$.

Находим $\lambda^+(0,4)$, $\lambda^-(0,4)$ по формулам (28):
$\lambda^+(0,4) = (3\cdot 2^{26}+1)/(5\cdot 2^{26})$,
$\lambda^-(0,4) = 3\cdot 2^{26}/(5\cdot 2^{26}+1)$.

Находим χ такие, для которых выполняются неравенства (35), т. е. $\lambda^+(0,4) \geq q(\chi|010)$, $\lambda^-(0,4) < q(\chi|010) + P(\chi|010)$, $P(\chi|010) > 0$. Неравенства выполняются для $\chi = 0$, т. к. $\lambda^+(0,4) > q(0|010) = 0$, $\lambda^-(0,4) < q(0|010) + P(0|010) = 3/5$, $P(0|010) = 3/5 > 0$, неравенства также выполняются для $\chi = 1$, т. к. $\lambda^+(0,4) > q(1|010) = 3/5$,
$\lambda^-(0,4) < q(1|010) + P(1|010) = 1$, $P(1|010) = 2/5 > 0$. Таким образом, $x'_4 = 0$, $x''_4 = 1$. Находим λ'^0_4, ρ'^0_4, λ''^0_4, ρ''^0_4 по формулам (36): $\lambda'^0_4 = q(0|010) = 0$, $\rho'^0_4 = P(0|010) = 3/5$, $\lambda''^0_4 = q(1|010) = 3/5$, $\rho''^0_4 = P(1|010) = 2/5$.

Находим λ'^1_2, ρ'^1_2, λ''^1_2, ρ''^1_2, а затем λ'^2_1, ρ'^2_1, λ''^2_1, ρ''^2_1 по формулам (33): $\lambda'^1_2 = 0$, $\rho'^1_2 = 3/5$, $\lambda''^1_2 = 3/5$, $\rho''^1_2 = 2/5$, $\lambda'^2_1 = 9/14$, $\rho'^2_1 = 3/14$, $\lambda''^2_1 = 12/14$, $\rho''^2_1 = 2/14$.

Производим сравнения (27). Обнаруживаем, что одновременно выполняются оба неравенства: $\lambda'^2_1 + \rho'^2_1 > \lambda^-(3,1)$, $\lambda''^2_1 \leq \lambda^+(3,1)$.

При этом выполняется равенство $1 = n/2^3$, т. е. $b = n/2^a$. Поэтому буква $x_4 = x''_4 = 1$,

буквы $x_5, \ldots, x_8$ находятся последовательно. Находится такая буква x_5, чтобы выполнялись условия $q(x_5|0101) = 0,\ P(x_5|0101) > 0$. Эти условия выполняются при $x_5 = 0$. Находится такое x_6, чтобы выполнялись условия $q(x_6|01010) = 0,\ P(x_6|01010) > 0$. Эти условия выполняются при $x_6 = 0$. Находится такое x_7, чтобы выполнялись условия $q(x_7|010100) = 0$, $P(x_7|010100) > 0$. Эти условия выполняются при $x_7 = 1$. Находится такая буква x_8, чтобы выполнялись условия $q(x_8|0101001) = 0,\ P(x_8|0101001) > 0$. Эти условия выполняются при $x_8 = 1$. Таким образом, искомое слово 01010011.

Теорема 3. *Описанный алгоритм денумерации позволяет находить слово $v = x_1 \ldots x_n$ из множества S по данному номеру $code_S(v)$.*

Доказательство. Для доказательства потребуется доказать две вспомогательных леммы.

Лемма 2. Для любого α $(0 \le \alpha < \log n)$ верно утверждение, что если для всех $0 \le i \le n/2^\alpha$ по паре оценок $\lambda^+(\alpha, i)$, $\lambda^-(\alpha, i)$ таких, что

$$\begin{aligned} \lambda^+(\alpha, i) - \lambda^-(\alpha, i) &< \frac{1}{q_{max}^{2^{\alpha+1}}}, \\ \lambda^-(\alpha, i) &< \lambda_i^\alpha + \rho_i^\alpha, \\ \lambda^+(\alpha, i) &\ge \lambda_i^\alpha, \end{aligned} \tag{37}$$

и префиксу $x_1 \ldots x_{2^\alpha(i-1)}$ можно, с помощью процедур 1 или 2, найти подслово $x_{2^\alpha(i-1)+1} \ldots x_{2^\alpha i}$ и значения λ_i^α, ρ_i^α или такую пару слов, что одно из них будет равно $x_{2^\alpha(i-1)+1} \ldots x_{2^\alpha i}$ и соответствующие им значения $\lambda_i'^\alpha$, $\lambda_i''^\alpha$, $\rho_i'^\alpha$, $\rho_i''^\alpha$, причём для $i = n/2^\alpha$ при выполнении дополнительного условия

$$\lambda^-(\alpha, i) \le \lambda_i^\alpha \tag{38}$$

можно однозначно найти подслово $x_{2^\alpha(i-1)+1} \ldots x_{2^\alpha i}$ и значения λ_i^α, ρ_i^α то для всех $0 \le i \le n/2^{\alpha+1}$ по паре оценок $\lambda^+(\alpha+1, i)$, $\lambda^-(\alpha+1, i)$ таких, что

$$\begin{aligned} \lambda^+(\alpha+1, i) - \lambda^-(\alpha+1, i) &< \frac{1}{q_{max}^{2^{\alpha+2}}}, \\ \lambda^-(\alpha+1, i) &< \lambda_i^{\alpha+1} + \rho_i^{\alpha+1}, \\ \lambda^+(\alpha+1, i) &\ge \lambda_i^{\alpha+1}, \end{aligned} \tag{39}$$

и префиксу $x_1 \ldots x_{2^{\alpha+1}(i-1)}$ можно, с помощью процедуры 1, найти подслово $x_{2^{\alpha+1}(i-1)+1} \ldots x_{2^{\alpha+1}i}$ (и значения $\lambda_i^{\alpha+1}$, $\rho_i^{\alpha+1}$) или такую пару слов, что одно из них будет равно $x_{2^{\alpha+1}(i-1)+1} \ldots x_{2^{\alpha+1}i}$ (и соответствующие им значения $\lambda_i'^{\alpha+1}$, $\lambda_i''^{\alpha+1}$, $\rho_i'^{\alpha+1}$, $\rho_i''^{\alpha+1}$), причём для $i = n/2^{\alpha+1}$ при выполнении дополнительного условия

$$\lambda^-(\alpha+1, i) \le \lambda_i^{\alpha+1} \tag{40}$$

подслово $x_{2^{\alpha+1}(i-1)+1} \ldots x_{2^{\alpha+1}i}$ можно найти однозначно.

Доказательство. Для некоторого a $(0 \le \alpha < \log n)$ предположим, что для всех $0 \le i \le n/2^\alpha$ по паре оценок $\lambda^+(\alpha, i)$, $\lambda^-(\alpha, i)$ таких, что

$$\begin{aligned} \lambda^+(\alpha, i) - \lambda^-(\alpha, i) &< \frac{1}{q_{max}^{2^{\alpha+1}}}, \\ \lambda^-(\alpha, i) &< \lambda_i^\alpha + \rho_i^\alpha, \\ \lambda^+(\alpha, i) &\ge \lambda_i^\alpha, \end{aligned} \tag{41}$$

и префиксу $x_1 \dots x_{2^\alpha(i-1)}$ можно, с помощью процедур 1 или 2, найти подслово $x_{2^\alpha(i-1)+1} \dots x_{2^\alpha i}$ (и значения λ_i^α, ρ_i^α) или такую пару слов, что одно из них будет равно $x_{2^\alpha(i-1)+1} \dots x_{2^\alpha i}$ (и соответствующие им значения $\lambda_i'^\alpha$, $\lambda_i''^\alpha$, $\rho_i'^\alpha$, $\rho_i''^\alpha$), причём для $i = n/2^\alpha$ при выполнении условия

$$\lambda^-(\alpha,\, i) \le \lambda_i^\alpha \tag{42}$$

подслово $x_{2^\alpha(i-1)+1} \dots x_{2^\alpha i}$ и значения λ_i^α, ρ_i^α можно найти однозначно.

Пусть для некоторого j, $1 \le j \le n/2^{\alpha+1}$, даны префикс $x_1 \dots x_{2^{\alpha+1}(j-1)}$ и оценки $\lambda^+(\alpha+1,\, j)$, $\lambda^-(\alpha+1,\, j)$, $1 \le j \le 2n/2^{\alpha+1}$ такие, что

$$\begin{aligned} \lambda^+(\alpha+1,\, j) - \lambda^-(\alpha+1,\, j) &< \frac{1}{q_{max}^{2^{\alpha+2}}}, \\ \lambda^-(\alpha+1,\, j) &< \lambda_j^{\alpha+1} + \rho_j^{\alpha+1}, \\ \lambda^+(\alpha+1,\, j) &\ge \lambda_j^{\alpha+1}, \end{aligned} \tag{43}$$

при этом если $j = n/2^{\alpha+1}$, то выполняется также условие

$$\lambda^-(\alpha+1,\, j) \le \lambda_j^{\alpha+1}. \tag{44}$$

Применяем процедуру 1 для $a = \alpha+1$, $b = j$. Находим $\lambda^+(\alpha, 2j-1)$, $\lambda^-(\alpha, 2j-1)$ по формулам (26). Покажем, что выполняются условия (41) для $i = 2j-1$.

По лемме (1) и по (43) и по тому, что $\alpha \ge 0$, $q_{max} > 1$, выполняются неравенства

$$\lambda^+(\alpha,\, 2j-1) - \lambda^-(\alpha,\, 2j-1) < \lambda^+(\alpha+1,\, j) - \lambda^-(\alpha+1,\, j) + 2 \cdot 2^{3-(2^{\alpha+2}\lceil \log q_{max} \rceil + 4)} <$$

$$< \frac{1}{q_{max}^{2^{\alpha+2}}} + \frac{1}{q_{max}^{2^{\alpha+2}}} < \frac{1}{q_{max}^{2^{\alpha+1}}}.$$

Таким образом,

$$\lambda^+(\alpha,\, 2j-1) - \lambda^-(\alpha,\, 2j-1) < \frac{1}{q_{max}^{2^{\alpha+1}}}. \tag{45}$$

По (7)

$$\lambda_j^{\alpha+1} = \lambda_{2j-1}^\alpha + \lambda_{2j}^\alpha \cdot \rho_{2j-1}^\alpha,\ \lambda_{2j}^\alpha \ge 0,\ \rho_{2j-1}^\alpha \ge 0,$$

отсюда следует, что

$$\lambda_{2j-1}^\alpha \le \lambda_j^{\alpha+1}.$$

Отсюда и по (26), (43) выполняются неравенства

$$\lambda^+(\alpha,\, 2j-1) \ge \lambda^+(\alpha+1, j) \ge \lambda_j^{\alpha+1} \ge \lambda_{2j-1}^\alpha,$$

таким образом верно неравенство

$$\lambda^+(\alpha,\, 2j-1) \ge \lambda_{2j-1}^\alpha. \tag{46}$$

По (7)

$$\lambda_j^{\alpha+1} + \rho_j^{\alpha+1} = \lambda_{2j-1}^\alpha + \lambda_{2j}^a \cdot \rho_{2j-1}^\alpha + \rho_{2j}^a \cdot \rho_{2j-1}^\alpha = \lambda_{2j-1}^\alpha + (\lambda_{2j}^\alpha + \rho_{2j}^\alpha) \cdot \rho_{2j-1}^\alpha.$$

Из (7) следует, что

$$\lambda_{2j}^\alpha + \rho_{2j}^\alpha \le 1.$$

Это означает, что

$$\lambda_j^{\alpha+1} + \rho_j^{\alpha+1} \le \lambda_{2j-1}^{\alpha} + \rho_{2j-1}^{\alpha}.$$

Отсюда и по (26), (43) выполняются неравенства

$$\lambda^-(\alpha,\, 2j-1) \le \lambda^-(\alpha+1, j) < \lambda_j^{\alpha+1} + \rho_j^{\alpha+1} \le \lambda_{2j-1}^{\alpha} + \rho_{2j-1}^{\alpha},$$

таким образом верно неравенство

$$\lambda^-(\alpha,\, 2j-1) < \lambda_{2j-1}^{\alpha} + \rho_{2j-1}^{\alpha}. \tag{47}$$

Из (45), (46), (50) следует выполнение условий (41) для $i = 2j-1$. Отсюда по предположению можно найти с помощью процедур 1 или 2 подслово $x_{2^\alpha(2j-2)+1} \dots x_{2^\alpha(2j-1)}$ и значения λ_{2j-1}^{α}, ρ_{2j-1}^{α} или пару подслов $x'_{2^\alpha(2j-2)+1} \dots x'_{2^\alpha(2j-1)}$ и $x''_{2^\alpha(2j-2)+1} \dots x''_{2^\alpha(2j-1)}$ и соотвествующие им значения $\lambda_{2j-1}^{\prime\alpha}$, $\rho_{2j-1}^{\prime\alpha}$, $\lambda_{2j-1}^{\prime\prime\alpha}$, $\rho_{2j-1}^{\prime\prime\alpha}$.

Во втором случае, согласно процедуре 1, проверяем, выполняются ли неравенства (27) для $a = \alpha+1$, $b = j$.

Покажем, что если выполняется первое неравенство, то, как и вычисляется при выполнении процедуры 1, $x_1 \dots x_{2^\alpha(2j-2)} = x''_1 \dots x''_{2^\alpha(2j-2)}$, $\lambda_{2j-1}^{\alpha} = \lambda''^{\alpha}_{2j-1}$, $\rho_{2j-1}^{\alpha} = \rho''^{\alpha}_{2j-1}$, если же выполняется второе неравенство, то $x_1 \dots x_{2^\alpha(2j-2)} = x'_1 \dots x'_{2^\alpha(2j-2)}$, $\lambda_{2j-1}^{\alpha} = \lambda'^{\alpha}_{2j-1}$, $\rho_{2j-1}^{\alpha} = \rho'^{\alpha}_{2j-1}$.

Из (7) следует, что

$$\lambda^+(\alpha+1,\, j) \ge \lambda_j^{\alpha+1} \ge \lambda_{2j-1}^{\alpha}.$$

Поэтому если

$$\lambda_{2j-1}^{\prime\alpha} > \lambda^+(\alpha+1,\, j),$$

то

$$\lambda_{2j-1}^{\prime\alpha} \ne \lambda_{2j-1}^{\alpha},$$

следовательно, $x'_{2^\alpha(2j-2)+1} \dots x'_{2^\alpha(2j-1)}$ не может быть равно $x_{2^\alpha(2j-2)+1} \dots x_{2^\alpha(2j-1)}$. Отсюда $x_{2^\alpha(2j-2)+1} \dots x_{2^\alpha(2j-1)} = x''_{2^\alpha(2j-2)+1} \dots x''_{2^\alpha(2j-1)}$ и соответственно $\lambda_{2j-1}^{\alpha} = \lambda_{2j-1}^{\prime\prime\alpha}$, $\rho_{2j-1}^{\alpha} = \rho_{2j-1}^{\prime\prime\alpha}$. Аналогично, если

$$\lambda'^{\alpha}_{2j-1} + \rho_{2j-1}^{\prime\alpha} \le \lambda^-(\alpha+1,\, j),$$

то это означает, что $x_{2^\alpha(2j-2)+1} \dots x_{2^\alpha(2j-1)} = x'_{2^\alpha(2j-2)+1} \dots x'_{2^\alpha(2j-1)}$ и соответственно $\lambda_{2j-1}^{\alpha} = \lambda_{2j-1}^{\prime\alpha}$, $\rho_{2j-1}^{\alpha} = \rho_{2j-1}^{\prime\alpha}$.

Если в результате сравнений (27) или без них нам на этом этапе согласно процедуре 1 нам стали известны точные значения λ_{2j-1}^{α} и ρ_{2j-1}^{α}, вычисляем оценки $\lambda^+(\alpha,\, 2j)$, $\lambda^-(\alpha,\, 2j)$ по формулам (28) для $a = \alpha+1$, $b = j$. Покажем, что выполняются условия (41) для $i = 2j$.

По лемме (1) и по (43) и по тому, что $\alpha \ge 0$, $q_{max} > 1$, выполняются неравенства

$$\lambda^+(\alpha,\, 2j) - \lambda^-(\alpha,\, 2j) < \frac{\lambda^+(\alpha+1,\, j) - \lambda^-(\alpha+1,\, j)}{\rho_{2j-1}^{a}} + 2 \cdot 2^{3-(2^{\alpha+2}\lceil \log q_{max} \rceil + 4)} <$$

$$< \frac{1}{q_{max}^{2^{\alpha+2}}} + \frac{1}{q_{max}^{2^{\alpha+2}}} < \frac{1}{q_{max}^{2^{\alpha+1}}}.$$

Таким образом,

$$\lambda^+(\alpha,\, 2j) - \lambda^-(\alpha,\, 2j) < \frac{1}{q_{max}^{2^{\alpha+1}}}. \tag{48}$$

По (28) и по (43) выполняются неравенства

$$\lambda^+(\alpha,\, 2j) \geq \frac{\lambda^+(\alpha+1,\, j) - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} \geq \frac{\lambda^{\alpha+1}_j - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} = \lambda^\alpha_{2j},$$

таким образом

$$\lambda^+(\alpha,\, 2j) \geq \lambda^\alpha_{2j}. \tag{49}$$

По (7)

$$\lambda^{\alpha+1}_j + \rho^{\alpha+1}_j = \lambda^\alpha_{2j-1} + \lambda^a_{2j} \cdot \rho^\alpha_{2j-1} + \rho^a_{2j} \cdot \rho^\alpha_{2j-1} = \lambda^\alpha_{2j-1} + (\lambda^\alpha_{2j} + \rho^\alpha_{2j}) \cdot \rho^\alpha_{2j-1}.$$

Из (7) следует, что

$$\lambda^\alpha_{2j} + \rho^\alpha_{2j} \leq 1.$$

Это означает, что

$$\lambda^{\alpha+1}_j + \rho^{\alpha+1}_j \leq \lambda^\alpha_{2j-1} + \rho^\alpha_{2j-1}.$$

Отсюда и по (26), (43) выполняются неравенства

$$\lambda^-(\alpha,\, 2j-1) \leq \lambda^-(\alpha+1, j) < \lambda^{\alpha+1}_j + \rho^{\alpha+1}_j \leq \lambda^\alpha_{2j-1} + \rho^\alpha_{2j-1},$$

таким образом, верно неравенство

$$\lambda^-(\alpha,\, 2j-1) < \lambda^\alpha_{2j-1} + \rho^\alpha_{2j-1}. \tag{50}$$

По (7), (28) и (43) выполняются неравенства

$$\lambda^-(\alpha,\, 2j) \leq \frac{\lambda^-(\alpha+1,\, j) - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} < \frac{\lambda^{\alpha+1}_j + \rho^{\alpha+1}_j - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} = \lambda^\alpha_{2j} + \rho^\alpha_{2j},$$

таким образом, верно неравенство

$$\lambda^-(\alpha,\, 2j) < \lambda^\alpha_{2j} + \rho^\alpha_{2j}. \tag{51}$$

Из (48), (49), (51) следует выполнение условий (41) для $i = 2j$. Аналогично доказывается выполнение условия (42) при $i = 2j = n/2^\alpha$. Отсюда по предположению можно найти с помощью процедур 1 или 2 подслово $x_{2^\alpha(2j-1)+1} \ldots x_{2^\alpha(2j)}$ и значения λ^α_{2j}, ρ^α_{2j} или пару подслов $x'_{2^\alpha(2j-1)+1} \ldots x'_{2^\alpha(2j)}$ и $x''_{2^\alpha(2j-1)+1} \ldots x''_{2^\alpha(2j)}$ и соотвествующие им значения λ'^α_{2j}, ρ'^α_{2j}, λ''^α_{2j}, ρ''^α_{2j}, причём для $j = n/2^{\alpha+1}$ можно найти подслово $x_{2^\alpha(2j-1)+1} \ldots x_{2^\alpha(2j)}$ однозначно.

Покажем, что если ни одно из неравенств (27) для $a = \alpha + 1$, $b = j$ не выполняется, то действительно, как и вычисляется в результате выполнения процедуры 1, буквы x'_i, x''_i $2^\alpha(2j-1) + 1 \leq i \leq 2^\alpha 2j$, можно найти последовательно, подбирая значения, удовлетворяющие условиям соответственно (29) и (30).

Очевидно, что невыполнение неравенств (27) равносильно одновременному выполнению неравенств

$$\begin{gathered} \lambda'^\alpha_{2j-1} + \rho'^\alpha_{2j-1} > \lambda^-(\alpha+1,\, j), \\ \lambda''^\alpha_{2j-1} \leq \lambda^+(\alpha+1,\, j). \end{gathered} \tag{52}$$

Так как $x'_{2^\alpha(2j-2)+1}\dots x'_{2^\alpha(2j-1)}$ лексикографически предшествует $x''_{2^\alpha(2j-2)+1}\dots x''_{2^\alpha(2j-1)}$, то существует такое минимальное t, $2^\alpha(2j-2)+1 \le t \le 2^\alpha(2j-1)$, что $x'_t < x''_t$. Это означает, по нахождению λ'^α_{2j-1} и λ''^α_{2j-1}, что

$$\lambda''^\alpha_{2j-1} - \lambda'^\alpha_{2j-1} \ge \frac{N(x_1\dots x'_{2^\alpha(2j-2)+1}\dots x'_t)}{N(x_1\dots x_{2^\alpha(2j-2)}} \ge \frac{N(x_1\dots x'_{2^\alpha(2j-2)+1}\dots x'_{2^\alpha(2j-1)})}{N(x_1\dots x_{2^\alpha(2j-2)}} = \rho'^\alpha_{2j-1},$$

то есть

$$\lambda'^\alpha_{2j-1} + \rho'^\alpha_{2j-1} \le \lambda''^\alpha_{2j-1}.$$

Отсюда и (52)

$$\begin{aligned} \lambda^-(\alpha+1,\, j) < \lambda'^\alpha_{2j-1} + \rho'^\alpha_{2j-1} \le \lambda^+(\alpha+1,\, j),\\ \lambda^-(\alpha+1,\, j) < \lambda''^\alpha_{2j-1} \le \lambda^+(\alpha+1,\, j). \end{aligned} \tag{53}$$

Отсюда и по (43)

$$\begin{aligned} \lambda^+(\alpha+1,j) - \lambda''^\alpha_{2j-1} < \frac{1}{q^{2^{\alpha+2}}_{max}},\\ (\lambda'^\alpha_{2j-1} + \rho'^\alpha_{2j-1}) - \lambda^-(\alpha+1,j) < \frac{1}{q^{2^{\alpha+2}}_{max}}. \end{aligned} \tag{54}$$

Пусть $x'_{2^\alpha(2j-2)+1}\dots x'_{2^{\alpha+1}j} = x_{2^\alpha(2j-2)+1}\dots x_{2^{\alpha+1}j}$. Тогда

$$\lambda^\alpha_{2j-1} = \lambda'^\alpha_{2j-1},\ \rho^\alpha_{2j-1} = \rho'^\alpha_{2j-1}.$$

По (7), (54), (43), (24)

$$\lambda^\alpha_{2j} + \rho^\alpha_{2j} = \frac{(\lambda^{\alpha+1}_j + \rho^{\alpha+1}_j) - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} >$$

$$> \frac{\lambda^-(\alpha+1,j) - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} > 1 - \frac{1}{q^{2^{\alpha+2}}_{max}} \cdot \frac{1}{\rho^\alpha_{2j-1}} \ge 1 - \frac{q^{2^\alpha}_{max}}{q^{2^{\alpha+2}}_{max}} > 1 - \frac{1}{q^{2^\alpha}_{max}}.$$

Таким образом,

$$\lambda^\alpha_{2j} + \rho^\alpha_{2j} > 1 - \frac{1}{q^{2^\alpha}_{max}}.$$

Легко убедиться, принимая во внимание (24), что это возможно только при

$$\lambda^\alpha_{2j} + \rho^\alpha_{2j} = 0.$$

Это означает, что если $x'_{2^\alpha(2j-2)+1}\dots x'_{2^\alpha(2j-1)} = x_{2^\alpha(2j-2)+1}\dots x_{2^{\alpha+1}(2j-1)}$,
то $x_{2^\alpha(2j-1)+1}\dots x_{2^\alpha(2j)}$ это слово, которое можно получить, находя последовательно буквы x'_i, $2^\alpha(2j-1)+1 \le i \le 2^\alpha 2j$, удовлетворяющие условиям (29).

Пусть $x_{2^\alpha(2j-2)+1}\dots x_{2^{\alpha+1}j} = x''_{2^\alpha(2j-2)+1}\dots x''_{2^{\alpha+1}j}$. Тогда

$$\lambda^\alpha_{2j-1} = \lambda''^\alpha_{2j-1},\ \rho^\alpha_{2j-1} = \rho''^\alpha_{2j-1}.$$

По (7), (54), (43), (24)

$$\lambda^\alpha_{2j} = \frac{\lambda^{\alpha+1}_j - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} \le$$

$$\le \frac{\lambda^+(\alpha+1,j) - \lambda^\alpha_{2j-1}}{\rho^\alpha_{2j-1}} < \frac{1}{q^{2^{\alpha+2}}_{max}} \cdot \frac{1}{\rho^\alpha_{2j-1}} \le \frac{q^{2^\alpha}_{max}}{q^{2^{\alpha+2}}_{max}} < \frac{1}{q^{2^\alpha}_{max}}.$$

Таким образом,

$$\lambda_{2j}^{\alpha} < \frac{1}{q_{max}^{2^\alpha}}.$$

Легко убедиться, принимая во внимание (24), что это возможно только при $\lambda_{2j}^{\alpha} = 0$. Это означает, что если $x_{2^\alpha(2j-2)+1} \dots x_{2^{\alpha+1}(2j-1)} = x''_{2^\alpha(2j-2)+1} \dots x''_{2^\alpha(2j-1)}$, то $x_{2^\alpha(2j-1)+1} \dots x_{2^\alpha(2j)}$ это слово, которое можно получить, находя последовательно буквы x''_i, $2^\alpha(2j-1)+1 \le k \le 2^\alpha 2j$, удовлетворяющие условиям (30).

Аналогично доказывается, что при $j = n/2^{\alpha+1}$ $x_{2^\alpha(2j-2)+1} \dots x_{2^{\alpha+1}j} = x''_{2^\alpha(2j-2)+1} \dots x''_{2^{\alpha+1}j}$, $x_{2^\alpha(2j-1)+1} \dots x_{2^\alpha(2j)}$ это слово, которое можно получить, находя последовательно буквы x''_i, $2^\alpha(2j-1)+1 \le k \le 2^\alpha 2j$, удовлетворяющие условиям (30).

Очевидно, что следующие шаги процедуры 1, заключающиеся в конкатенации полученных подслов, приводят к получению подслова $x_{2^{\alpha+1}(j-1)+1} \dots x_{2^{\alpha+1}j}$ или такой пары слов, что одно из них будет равно $x_{2^{\alpha+1}(j-1)+1} \dots x_{2^{\alpha+1}j}$, причём для $j = n/2^{\alpha+1}$ подслово $x_{2^{\alpha+1}(j-1)+1} \dots x_{2^{\alpha+1}j}$ можно найти однозначно.

Затем, согласно процедуре 1, находятся значения $\lambda_j^{\alpha+1}$, $\rho_j^{\alpha+1}$ или $\lambda_j'^{\alpha+1}$, $\rho_j'^{\alpha+1}$, $\lambda_j''^{\alpha+1}$, $\rho_j''^{\alpha+1}$ по формулам (32), (33), (34).

□

Лемма 3. Для всех $0 \le i \le n$ по паре оценок $\lambda^+(0,\, i)$, $\lambda^-(0,\, i)$ таких, что

$$\begin{aligned} \lambda^+(0,\, i) - \lambda^-(0,\, i) &< \frac{1}{q_{max}^2}, \\ \lambda^+(0,\, i) &\ge \lambda_i^0, \\ \lambda^-(0,\, i) &< \lambda_i^0 + \rho_i^0, \end{aligned} \tag{55}$$

и префиксу $x_1 \dots x_{i-1}$ можно, с помощью процедуры 2, найти букву x_i и значения λ_i^0, ρ_i^0 или такую пару букв x'_i и x''_i, что одна из них равна x_i и соответствующие им значения $\lambda_i'^0$, $\lambda_i''^0$, $\rho_i'^0$, $\rho_i''^0$, причём для $i = n$ при выполнении дополнительного условия

$$\lambda^-(0,\, i) \le \lambda_i^0, \tag{56}$$

букву x_n можно найти однозначно.

Доказательство. Пусть для некоторого i, $0 \le i \le n$, даны оценки $\lambda^+(0,\, i)$, $\lambda^-(0,\, i)$ такие, что выполняется условие (55) префикс $x_1 \dots x_{i-1}$.

Применяя процедуру 2 для $b = i$, находим все χ, удовлетворяющие условиям (35). Из (55) следует, что для x_i выполняются условия (35).

Покажем, что таких найденных χ может быть не более двух, что неочевидно в случае, если нумеруются слова над алфавитом, в котором более двух букв. Пусть таких χ более двух. Обозначим их в порядке возрастания χ_1, χ_2, χ_3. Тогда для χ_1 $\lambda_i^0 + \rho_i^0 = q(\chi_1|x_1 \dots x_{i-1}) + P(\chi_1|\dots x_{i-1})$, что равно по (4) $q(\chi_2|x_1 \dots x_{i-1})$. Для χ_3 $\lambda_i^0 = q(\chi_3|x_1 \dots x_{i-1})$, что равно, по (4), $q(\chi_2|x_1 \dots x_{i-1}) + P(\chi_2|\dots x_{i-1})$. Таким образом, одновременно выполняются неравенства

$$\begin{aligned} \lambda^-(0,\, i) &< q(\chi_2|x_1 \dots x_{i-1}), \\ \lambda^+(0,\, i) &\ge q(\chi_2|x_1 \dots x_{i-1}) + P(\chi_2|\dots x_{i-1}). \end{aligned}$$

Отсюда

$$\lambda^+(0,\, i) - \lambda^-(0,\, i) > P(\chi_2|x_1 \dots x_{i-1}).$$

По (24) $P(\chi_2|x_1 \dots x_{i-1}) \ge 1/q_{max}$. Т. е.

$$\lambda^+(0,\, i) - \lambda^-(0,\, i) > 1/q_{max}.$$

Получили противоречие с условиями (55). Отсюда следует, что может быть не более двух χ, удовлетворяющих условиям (35). Значит, мы можем, применяя процедуру 2 для $b = i$, найти букву x_i или такую пару букв x_i' и x_i'', что одна из них равна x_i.

Аналогично доказывается, что если даны оценки со свойствами (55), (56), то, применяя процедуру 2, можно найти букву x_i.

Очевидно, что в случае, если найдено x_i, с помощью процедуры 2 находятся λ_i^0 и ρ_i^0 по формулам (7); в случае, если найдены x_i' и x_i'', с помощью процедуры 2 находятся $\lambda_i'^0$, $\rho_i'^0$, $\lambda_i''^0$, $\rho_i''^0$, по формулам (36). □

Из леммы 1, следует что оценки $\lambda^+(1, \log n)$, $\lambda^-(1, \log n)$, полученные на первом шаге выполнения алгоритма по формулам (25), обладают свойствами (37), (38) для $i = 1$, $\alpha = \log n$. Отсюда по леммам 2, 3 описанный алгоритм денумерации позволяет находить слово $v = x_1 \dots x_n$ из множества S по данному номеру $code_S(v)$. □

Опишем свойства алгоритма денумерации слов множества S.

Теорема 4. *1) Сложность денумерации, т. е. время, требуемое для нахождения одной буквы денумеруемого слова, равна*

$$O(\log n M(n \log q_{max})/n), \tag{57}$$

где $M(n)$ — время умножения или деления двух слов длины n.

2) Объём памяти, требуемый для денумерации слова множества S длины n, равен

$$O((n \log q_{max}) \log n). \tag{58}$$

Следствие 1. При использовании алгоритма быстрого умножения Шёнхаге – Штрассена, для которого $M(n) = O(n \log n \log\log n)$, скорость нумерации равна $O(\log n \log q_{max} \log(n \log q_{max}) \log\log(n \log q_{max}))$.

Следствие 2. При использовании алгоритма быстрого умножения Фюрера, для которого $M(n) = O(n \log n 2^{O(\log^* n)})$, скорость нумерации равна $O(\log n \log(n \log q_{max}) 2^{O(\log^*(n \log q_{max}))})$.

Д о к а з а т е л ь с т в о . Одной операции умножения при вычислении λ_b^a в (7) соответствует одна операция деления, причём при вычислении оценок $\lambda^+(a, b)$, $\lambda^-(a, b)$ длина числителя и знаменателя делимого не превосходит $2^a \log q_{max}$ бит, а делителя – $2^a \log q_{max}$ бит.

Длина делимого и делителя пропорциональна длине сомножителей, используемых при вычислении λ_b^a. Следовательно, и время вычисления $\lambda^+(a, b)$, $\lambda^-(a, b)$ будет пропорционально времени вычисления λ_b^a. Кроме операций деления, при декодировании вычисляются те же величины λ_b^a и ρ_b^a или аналогичные им пары величин $\lambda_b'^a$, $\lambda_b''^a$ и $\rho_b'^a$, $\rho_b''^a$, которые используются при кодировании и для этого используется столько же времени. Следовательно, время, затрачиваемое на выполнение операций, необходимых на вычисление оценок $\lambda^+(a, b)$, $\lambda^-(a, b)$, $0 \le a \le \log n$, $1 \le b \le n/2^a$ пропорционально времени кодирования $O(\log n M(n \log q_{max})/n)$.

Из описания алгоритма видно, что декодирование левой половины $x_1 \dots x_{n/2}$ и правой половины $x_{n/2+1} \dots x_n$ слова проводится независимо и может осуществляться при использовании одной и той же памяти. Рассуждая так же, как при анализе кодирования, легко получить, что объёмы памяти, используемой при кодировании и декодировании, асимптотически совпадают. □

Опишем свойства денумерации слов языков Дика из скобок одного типа.

Теорема 5. *1)Скорость кодирования слова длины n языка Дика над алфавитом мощности 2, т. е. время, требуемое для кодирования одной буквы, равна $O(\log n M(n \log n)/n)$ битовых операций, где $M(n)$ – время умножения двух слов длины n.*

2)Объём памяти, требуемый для кодирования слова длины n языка Дика над алфавитом мощности 2, равен $O(n \log n)$ бит.

Следствие 1. При использовании алгоритма быстрого умножения Шёнхаге – Штрассена, имеющего сложность $M(n) = n \log n \log\log n$, скорость денумерации равна $O(\log^3 n \log\log n)$.

Следствие 2. При использовании алгоритма быстрого умножения Фюрера, имеющего сложность $M(n) = n \log n 2^{O(\log^* n)}$, скорость денумерации равна $O(\log^3 n 2^{O(\log^* n)})$.

Доказательство. Следует из теоремы 4 и (20). □

4. Нумерация и денумерация двоичных слов длины n, содержащих $n/2$ единиц

Обозначим через B_n множество двоичных слов длины n (n – чётное), в которых содержится ровно $n/2$ единиц. Например, для $n = 4$ множество B_4 будет состоять из следующих слов: 0011, 0101, 0110, 1001, 1010, 1100.

Описанные в частях 2, 3 алгоритмы нумерации и денумерации слов множества S можно применить для нумерации слов множества B_n.

Значение $|B_n|$ вычисляется исходя из определения множества B_n по формуле

$$|B_n| = C_n^{n/2}. \tag{59}$$

Из определения следует, что

$$N_{B_n}(x_1 x_2 \dots x_i) = C_{n-i}^{n/2-m}, \tag{60}$$

где m — количество единиц в префиксе $x_1 x_2 \dots x_i$.

Можно видеть из (60) и определения (4), что для множества B_n значения $P(x_i|x_1 \dots x_{i-1})$ $(0 < i \le n)$ вычисляются по формулам:

$$\begin{gathered} P(x_i|x_1 \dots x_{i-1}) = \frac{(n-i)!}{(n/2-i+m)!(n/2-m)!} : \\ : \frac{(n-i+1)!}{(n/2-i+1+m)!(n/2-m)!} = \frac{n/2-i+1+m}{n-i+1}, \end{gathered} \tag{61}$$

при $x_i = 0$;

$$\begin{gathered} P(x_i|x_1 \dots x_{i-1}) = \frac{(n-i)!}{(n/2-i+m)!(n/2-m)!} : \\ : \frac{(n-i+1)!}{(n/2-i+m)!(n/2-m+1)!} = \frac{n/2-m+1}{n-i+1}, \end{gathered} \tag{62}$$

при $x_i = 1$.

Применяя формулы (59), (61) и (62) и алгоритмы нумерации и денумерации слов множества S, описанные в частях 2, 3, получаем алгоритмы нумерации и денумерации слов множества B_n.

Можно видеть по определению q_{max} и из (61) и (62), что для множества B_n

$$q_{max} \le n. \tag{63}$$

Свойства алгоритмов кодирования и декодирования описывает следующая теорема.

Теорема 6. *Объём памяти, требуемый для кодирования двоичного слова длины* n *с количеством единиц* $n/2$*, равен* $O(n \log n)$ *бит. Скорость кодирования слова длины* n *с заданным количеством единиц, т. е. время, требуемое для кодирования одной буквы, равна* $O(\log n M(n \log n)/n)$ *битовых операций, где* $M(n \log n)$ *– время умножения двух слов длины* $n \log n$.

Следствие 1. При использовании алгоритма быстрого умножения Шёнхаге – Штрассена, для которого $M(n) = n \log n \log\log n$*, скорость нумерации равна* $O(\log^3 n \log\log n)$.

Следствие 2. При использовании алгоритма быстрого умножения Фюрера, для которого $M(n) = n \log n 2^{O(\log^* n)}$*, скорость нумерации равна* $O(\log^3 n 2^{O(\log^* n)})$.

Объём памяти, требуемый для денумерации слова длины n *с количеством единиц* $n/2$*, равен* $O(n \log^2 n)$.

Сложность денумерации, т. е. время, требуемое для нахождения одной буквы денумеруемого слова, равна $O(\log n M(n \log n)/n)$*, где* $M(n \log n)$ *— время умножения или деления двух слов длины* $n \log n$.

Следствие 1. При использовании алгоритма быстрого умножения Шёнхаге – Штрассена, имеющего сложность $M(n) = n \log n \log\log n$*, скорость денумерации равна* $O(\log^3 n \log\log n)$.

Следствие 2. При использовании алгоритма быстрого умножения Фюрера, имеющего сложность $M(n) = n \log n 2^{O(\log^* n)}$*, скорость денумерации равна* $O(\log^3 n 2^{O(\log^* n)})$.

Доказательство. Свойства следуют из теорем 1 и 4 и из (63). □

Литература

1. *Beenker G. F. M., Immink K. A. S.* A generalized method for encoding and decoding run-length-limited binary sequences // IEEE Transactions on Information Theory. 1983. V. 29, № 3. P. 751–754.

2. *Cover T. M.* Enumerative source encoding // IEEE Transactions on Information Theory. 1973. V. IT-19, №. 1. P. 73–77.

3. *Datta S. and McLaughlin S. W.* An enumerative method for runlength-limited codes: permutation codes // IEEE Transactions on Information Theory. 1999. V. IT-45, № 6. P. 2199–2204.

4. *Führer M.* Faster integer multiplication // Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing (STOC 2007). San Diego, California, USA. 2007. P. 57–66.

5. *Gadouleau M. and Yan Z.* Constant-rank codes and their connection to constant-dimension codes // IEEE Transactions on Information Theory. 2010. V. IT-56. P. 3207–3216.

6. *Kautz W.* Fibonacci codes for synchronization control // IEEE Transactions on Information Theory. 1965. V. 11, № 2. P. 284–292.

7. *Knuth D. E.* Efficient balanced codes // IEEE Transactions on Information Theory. 1986. V. IT-32. P. 51–53.

8. *Koetter R. and Kshcischang F. R.* Coding for errors and erasures in random network coding // IEEE Transactions on Information Theory. 2008. V. 54, № 8. P. 3579–3591.

9. *Lint J. H., van and Wilson R. M.* A Course in Combinatorics. Cambridge University Press, 2001 (second edition).

10. *Milenkovic O. and Vasic B.* Permutation $(d;k)$-codes: efficient enumerative coding and phrase length distribution shaping // IEEE Transactions on Information Theory. 2000. V. IT-46, № 7. P. 2671–2675.

11. *Schönhage A. and Strassen V.* Schnelle Multiplikation grosser Zahlen // Computing. 1971. V. 7. P. 281–292.

12. *Silberstein N. and Etzion T.* Enumerative coding for Grassmannian space // IEEE Transactions on Information Theory. 2011. V. 57, № 1. P. 365–374.

13. *Silberstein N. and Etzion T.* Error-correcting codes in projective space via rank-metric codes and Ferrers diagrams // IEEE Transactions on Information Theory. 2009. V. IT-55. P. 2909–2919.

14. *Skachek V.* Recursive code construction for random networks // IEEE Transactions on Information Theory. 2010. V. IT-56. P. 1378–1382.

15. *Tang D. T., Bahl L. R.* Block codes for a class of constrained noiseless channels // Information and Control. 1970. V. 17, № 5. P. 436–461.

16. *Weber, J. H., Schouhamer Immink.* Knuth's balanced codes revisited // IEEE Transactions on Information Theory. 2010. V. 56. P. 1673–1679.

17. *Ахо А., Лам М., Сети Р., Ульман Дж.* Компиляторы. Принципы, технологии и инструментарий. М.: Вильямс, 2008.

18. *Кричевский Р. Е.* Сжатие и поиск информации. М.: Радио и связь, 1989.

19. *Медведева Ю. С.* Быстрая нумерация элементов грассманиана // Вычислительные технологии. 2013. Т. 18, № 3. С. 22–33.

20. *Медведева Ю. С., Рябко Б. Я.* Быстрый алгоритм нумерации слов с заданными ограничениями на длины серий единиц. // Проблемы передачи информ. 2010. Т. 46, № 4. С. 130–139.

21. *Рейнольд Э., Нивергельт Ю., Део Н.* Комбинаторные алгоритмы. Теория и практика. М.: Мир, 1980.

22. *Рябко Б. Я.* Быстрая нумерация комбинаторных объектов. // Дискретная математика. 1998. Т. 10. № 2. С. 101–119.

23. *Шень А.* Программирование: теоремы и задачи. М.: МЦНМО, 2004.

Статья поступила в редакцию 03.10.2013;
переработанный вариант – 01.11.2013.

Медведева Юлия Сергеевна
аспирант ИВТ СО РАН (630090 Россия, Новосибирск, пр. ак. Лаврентьева, 6), тел. (383) 330-61-50, e-mail: brainwashed@yandex.ru.

Fast algorithm of enumerative encoding for main problems of information theory

Yu. Medvedeva

We propose the algorithm of fast enumeration for main problems of coding theory. These problems are: 1) encoding of binary words with given number of ones and its special case when number of ones equals number of zeroes; 2) encoding of run-length-constrained words. This problem is of interest to magnetic recording and some other fields; 3) encoding of the Grassmannian space elements and encoding of the Dyck language words. We apply the modification of method of fast enumeration of combinatorial objects proposed by B. Ryabko to stated problems. Our algorithm has less computational complexity than other known algorithms do.

Keywords: encoding, enumerative encoding, information theory.