Math-Net.Ru

Общероссийский математический портал

Ю. С. Медведева, Быстрая нумерация слов, порождаемых грамматиками Дика, *Матем. заметки*, 2014, том 96, выпуск 1, 51–69

DOI: <https://doi.org/10.4213/mzm10398>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением <http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 54.196.121.252

26 апреля 2023 г., 16:59:35

Быстрая нумерация слов, порождаемых грамматиками Дика

Ю. С. Медведева

Задача нумерации и денумерации слов, порождаемых грамматиками Дика, возникает при работе трансляторов для языков высокого уровня и в целом ряде других приложений. В данной работе предлагается алгоритм быстрой нумерации и денумерации слов языков Дика, сложность которого на один символ нумеруемого слова при использовании алгоритма умножения и деления Шёнхаге–Штрассена равна $O(\log^3 n \log \log n)$ битовых операций. Применение ранее известных методов имеет сложность $O(n)$ на один символ нумеруемого слова. Построение предложенного алгоритма основано на методе Б. Я. Рябко.

Библиография: 11 названий.

DOI: 10.4213/mzm10398

1. Введение. Задача нумерации слов из некоторого упорядоченного множества W состоит в следующем: каждому слову из W ставится в соответствие его номер, т.е. единственное число из диапазона $[0; |W| - 1]$. При денумерации решается обратная задача: по числу из диапазона $[0; |W| - 1]$ находится соответствующее ему слово из W . Обычно на W рассматривается лексикографический порядок, а номера представляются в двоичной системе.

Задачи нумерации различных видов комбинаторных объектов привлекали внимание многих авторов. Среди этих задач можно отметить задачу нумерации слов с заданным количеством нулей и единиц и задачу нумерации слов с ограничением на количество подряд идущих одинаковых символов [1], [2].

В общем виде задача нумерации и денумерации рассматривалась в работе Ковера [3], где был предложен общий метод для любого заданного множества слов. В работе Рябко [4] был впервые предложен быстрый общий алгоритм нумерации слов заданного множества, который для множеств многих комбинаторных объектов имеет скорость экспоненциально большую, чем метод Ковера.

В данной работе рассматривается задача нумерации и денумерации слов языков Дика [5] или “правильных” последовательностей скобок. Словами языка Дика над $2m$ буквами 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 трех типов. Всего таких слов 18 (см. таблицу). Сопоставим им номера, записанные в двоичном виде, длины $\lceil \log_2 18 \rceil = 5$. В первый столбец запишем все такие слова, а во второй запишем их номера в двоичном виде.

Таблица

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 $2m$ длины $2n$, 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.

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 [6]–[8].

Алгоритм нумерации слов длины $2n$ языков Дика над алфавитом $2m$, основанный на методе Ковера [3], 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 $2n$ языков Дика над алфавитом $2m$, предлагаемый в данной работе и основанный на подходе из работы [4], имеет сложность $O(\log n M(n \log n)/n)$ битовых операций на один символ нумеруемого слова, где $M(k)$ время умножения или деления слов длины k . Если использовать метод Шёнхаге–Штрассена [9], сложность которого $O(k \log k \log \log k)$ при умножении или делении слов длины k , то сложность рассматриваемого метода $O(\log^3 n \log \log n)$ на один символ нумеруемого слова. Если использовать метод Фюрера [10], сложность которого $O(k \log k 2^{O(\log^* k)})$ при умножении или делении слов длины k , то сложность рассматриваемого метода $O(\log^3 n 2^{O(\log^* n)})$ на один символ нумеруемого слова.

2. Быстрый алгоритм нумерации слов языков Дика. Обозначим D_{2n}^{2m} множество слов языка Дика над алфавитом $2m$ длины $2n$. Нужно найти номер слова w из множества D_{2n}^{2m} среди всех слов этого множества.

Опишем алгоритм нумерации на примере поиска номера слова $w = () [] ([])$ среди слов множества D_8^4 , т.е. среди всех слов длины 8 правильных расстановок скобок двух типов.

Заменяем открывающие скобки всех видов на нули, а закрывающие скобки всех видов на единицы. Легко видеть, что полученное слово w' будет словом языка Дика над алфавитом $\{0, 1\}$, т.е. будет соответствовать правильной расстановке скобок одного вида. В данном случае получаем слово $w' = 01010011$. Обозначим S_{2n} множество всех слов из n нулей и n единиц, соответствующих правильным расстановкам

скобок. Мощность S_{2n} равна n -му числу Каталана [11],

$$|S_{2n}| = C_n = C_{2n}^n - C_{2n}^{n-1}.$$

Для описания метода мы рассмотрим вспомогательное множество A_m^n слов длины n алфавита $A_m = \{a_0, a_1, \dots, a_{m-1}\}$. Теперь сопоставим нумерованному слову множества D_{2n}^{2m} слово w'' , принадлежащее множеству A_m^n . Для этого заменим все открывающие скобки первого типа символом a_0 , все открывающие скобки второго типа символом a_1 и т.д., а закрывающие исключим. В данном случае слово $() [] ([)]$ будет сопоставлено слову $w'' = a_0 a_1 a_0 a_1$. Можно видеть, что порядок типов закрывающих скобок однозначно определяется порядком типов открывающих.

Таким образом, каждое слово множества D_{2n}^{2m} можно однозначно сопоставить паре слов (w', w'') , одно из множества S_{2n} , другое из множества A_m^n . И обратно, каждая пара слов, одно из множества S_{2n} , другое из множества A_m^n , однозначно определяет соответствующее ей слово множества D_{2n}^{2m} . Таким образом,

$$|D_{2n}^{2m}| = |A_m^n| \cdot |S_{2n}| = m^n \cdot (C_{2n}^n - C_{2n}^{n-1}).$$

В рассматриваемом примере:

$$|D_8^4| = |A_2^4| \cdot |S_8| = 2^4 \cdot (C_8^4 - C_8^3) = 2^4 \cdot (70 - 56) = 224.$$

Для описания алгоритма мы будем использовать следующее упорядочивание слов множества D_{2n}^{2m} : сначала будем упорядочивать их согласно лексикографическому порядку соответствующих им слов множества A_m^n , затем согласно лексикографическому порядку соответствующих им слов множества S_{2n} .

Обозначим номер слова w среди множества D_{2n}^{2m} , упорядоченного таким образом, через $N(w)$, номер слова w' среди множества S_{2n} , упорядоченного лексикографически, через $N'(w')$, номер слова w'' среди множества A_m^n , также упорядоченного лексикографически, через $N''(w'')$. Тогда легко видеть, что

$$N(w) = N''(w'') \cdot |S_{2n}| + N'(w'). \quad (2.1)$$

Описание нашего метода будет удобно начать с описания нахождения $N'(w')$ с помощью метода Ковера из [3]. Согласно [3] номер слова среди множества S_{2n} , упорядоченного лексикографически, можно найти по формуле

$$N'(x_1 x_2 \dots x_{2n}) = \sum_{i=1}^{2n} \sum_{\chi < x_i} N_{S_{2n}}(x_1 x_2 \dots x_{i-1} \chi), \quad (2.2)$$

где $N_{S_{2n}}(x_1 x_2 \dots x_{i-1} \chi)$ – количество слов из S_{2n} , начинающихся с $x_1 x_2 \dots x_{i-1} \chi$.

Покажем на примере $N_{S_8}(01)$, чему равно такое количество, т.е. сколько слов из множества слов, имеющих длину 8 из нулей и единиц, соответствующих правильным расстановкам скобок, начинаются на 01. Словами множества S_8 , начинающимися на 01, будут являться слова, состоящие из четырех нулей и четырех единиц, которые начинаются на 01, кроме тех, которые не соответствуют правильным расстановкам скобок. Количество всех слов, состоящих из четырех нулей и четырех единиц, начинающихся на 01, легко найти; оно равно количеству всех слов, состоящих из трех нулей и трех единиц, т.е. $C_6^3 = 20$.

Слова, начинающиеся на 01, состоящие из четырех нулей и четырех единиц и не соответствующие правильным расстановкам скобок, это такие слова из четырех нулей и четырех единиц, для которых существует такое j , $2 < j \leq 8$, что количество единиц в последовательности $x_1 x_2 \dots x_j$ превышает количество нулей в этой последовательности. Существует взаимоднозначное соответствие таких слов и всех слов, состоящих из трех нулей и пяти единиц и начинающихся на 01. Это отображение осуществляется следующим образом. Для слова, не соответствующего правильной расстановке скобок, есть такие j , $2 < j \leq 8$, 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. Заменяем теперь в слове все символы после j -го на противоположные. Получаем слово из трех нулей и пяти единиц, начинающееся на 01. Так как это отображение взаимоднозначное, количество слов, начинающихся на 01, состоящих из четырех нулей и четырех единиц и не соответствующих правильным расстановкам скобок, равно количеству всех слов, начинающихся на 01 и состоящих из трех нулей и пяти единиц. Таких слов столько же, сколько слов, состоящих из двух нулей и четырех единиц, т.е. $C_6^2 = 15$. Таким образом,

$$N_{S_8}(01) = C_6^3 - C_6^2 = 5.$$

В общем виде

$$\begin{aligned} N_{S_{2n}}(x_1 x_2 \dots x_i) &= C_{2n-i}^{m-z} - C_{2n-i}^{m-z-1} \\ &= \frac{(2n-i)!}{(n-z)!(n-i+z)!} - \frac{(2n-i)!}{(n-z-1)!(n-i+z+1)!} \\ &= \frac{(2n-i)!(2z-i+1)}{(n-z)!(n-i+z+1)!}, \end{aligned} \quad (2.3)$$

где z – количество нулей в $x_1 x_2 \dots x_i$, при том, что $x_1 x_2 \dots x_i$ может быть началом слова, соответствующего правильной расстановке скобок длины $2n$. Если $x_1 x_2 \dots x_i$ не может быть началом слов, соответствующих правильной расстановке скобок длины $2n$, то, очевидно, $N_{S_{2n}}(x_1 \dots x_i) = 0$.

Мы видим, что $N_{S_{2n}}(x_1 \dots x_i)$ зависит только от $0 < i \leq 2n$, n и z – количества нулей в $x_1 x_2 \dots x_i$ ($0 < z \leq i$, $0 < z \leq n$). Таким образом, для заданного n существует не больше $3n^2/2$ различных $N_{S_{2n}}(x_1 \dots x_i)$. Кроме того, мы видим, что значения $N_{S_{2n}}(x_1 \dots x_i)$ не превосходят 2^{2n-i} , т.е. имеют длину не больше $2n-i$.

Будем использовать вспомогательную таблицу, которая каждой паре значений i , $0 < i < 2n$, и z , $0 \leq z \leq \min(i, n)$, сопоставляет значение

$$N_{S_{2n}}(x_1 \dots x_i) = \frac{(2n-i)!(2z-i+1)}{(n-z)!(n-i+z+1)!},$$

одинаковое для всех слов длины i с z нулями. Размер таблицы равен $O(n^3)$.

Для получения номера слова w' мы согласно (2.2) находим $N_{S_{2n}}(x_1 x_2 \dots x_{i-1} 0)$ для тех i , для которых $x_i = 1$, и складываем их. Эти значения для каждого такого i находятся следующим образом. Находится значение z , равное количеству нулей

в слове $x_1 \dots x_{i-1}0$. С помощью таблицы находится соответствующее паре i и z значение $N_{S_{2n}}(x_1 \dots x_i)$.

В данном примере:

$$\begin{aligned} N'(01010011) &= N_{S_8}(00) + N_{S_8}(0100) + N_{S_8}(0101000) \\ &\quad + N_{S_8}(01010010) = 9 + 3 + 0 + 0 = 12. \end{aligned} \quad (2.4)$$

Значения $N_{S_8}(00)$, $N_{S_8}(0100)$ при этом берутся из 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 по таблице находится значение $N_{S_8}(00)$, соответствующее паре (2, 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 $N_{S_8}(0100)$, соответствующее паре (4, 3) и равное $(4!3)/(1!4!) = 3$. Значения же $N_{S_8}(0101000)$ и $N_{S_8}(01010010)$ равны нулю, так как не существует слов множества S_8 , начинающихся на 0101000 или 01010010.

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

Перейдем теперь к описанию нахождения $N'(w')$ предлагаемым методом.

Определим величины $P(x_i|x_1 \dots x_{i-1})$, $q(x_i|x_1 \dots x_{i-1})$ при $0 < i \leq 2n$:

$$\begin{aligned} P(x_1) &= \frac{N_{S_{2n}}(x_1)}{|S_{2n}|}, & P(x_i|x_1 x_2 \dots x_{i-1}) &= \frac{N_{S_{2n}}(x_1 x_2 \dots x_i)}{N_{S_{2n}}(x_1 x_2 \dots x_{i-1})}, \\ q(x_1) &= \sum_{\chi < x_1} P(\chi), & q(x_i|x_1 \dots x_{i-1}) &= \sum_{\chi < x_i} P(\chi|x_1 \dots x_{i-1}). \end{aligned} \quad (2.5)$$

Можно видеть, что по (2.2)

$$N'(x_1 \dots x_{2n}) = |S_{2n}|(q(x_1) + q(x_2|x_1)P(x_1) + q(x_3|x_1 x_2)P(x_2|x_1)P(x_1) + \dots). \quad (2.6)$$

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

$$\begin{aligned} N'(x_1 \dots x_{2n}) &= |S_{2n}|((q(x_1) + q(x_2|x_1)P(x_1)) \\ &\quad + ((q(x_3|x_1 x_2) + q(x_4|x_1 \dots x_3)P(x_3|x_1 x_2))P(x_2|x_1)P(x_1)) + \dots). \end{aligned} \quad (2.7)$$

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

$$\begin{aligned} \rho_b^0 &= P(x_b|x_1 \dots x_{b-1}), & \lambda_b^0 &= q(x_b|x_1 \dots x_{b-1}), \\ \rho_b^a &= \rho_{2b-1}^{a-1} \rho_{2b}^{a-1}, & \lambda_b^a &= \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1} \lambda_{2b}^{a-1}. \end{aligned} \quad (2.8)$$

Тогда

$$\begin{aligned} \lambda^{\log(2n)} &= ((q(x_1) + q(x_2|x_1)P(x_1)) \\ &\quad + ((q(x_3|x_1 x_2) + q(x_4|x_1 \dots x_3)P(x_3|x_1 x_2)) \cdot P(x_2|x_1)P(x_1)) + \dots). \end{aligned}$$

Отсюда и (2.7)

$$N'(x_1 x_2 \dots x_{2n}) = \lambda_1^{\log(2n)} |S_{2n}| = \lambda_1^{\log(2n)} (C_{2n}^n - C_{2n}^{n-1}). \quad (2.9)$$

Можно видеть, что для $0 < i \leq 2n$ при $x_i = 0$

$$\begin{aligned} P(x_i | x_1 x_2 \dots x_{i-1}) &= \frac{(2n-i)!(2z-i+1)}{(n-z)!(n-i+z+1)!} \Bigg/ \frac{(2n-i+1)!(2z-i)}{(n-z+1)!(n-i+z+1)!} \\ &= \frac{(2z-i+1)(n-z+1)}{(2z-i)(2n-i+1)}, \end{aligned} \quad (2.10)$$

при $x_i = 1$

$$\begin{aligned} P(x_i | x_1 x_2 \dots x_{i-1}) &= \frac{(2n-i)!(2z-i+1)}{(n-z)!(n-i+z+1)!} \Bigg/ \frac{(2n-i+1)!(2z-i+2)}{(n-z)!(n-i+z+2)!} \\ &= \frac{(2z-i+1)(n-i+z+2)}{(2n-i+1)(2z-i+2)}. \end{aligned} \quad (2.11)$$

Перейдем теперь к тем вычислениям, которые будут служить иллюстрацией алгоритма. По формулам (2.10) и (2.11) находим значения

$$\begin{aligned} P(x_1) &= \rho_1^0, \quad P(x_2 | x_1) = \rho_2^0, \quad \dots, \quad P(x_8 | x_1 x_2 \dots x_7) = \rho_8^0, \\ q(x_1) &= \lambda_1^0, \quad q(x_2) = \lambda_2^0, \quad \dots, \quad q(x_8) = \lambda_8^0, \end{aligned}$$

и

$$\begin{aligned} P(x_1) &= 1, \quad P(x_2 | x_1) = P(1|0) = \frac{5}{14}, \quad P(x_3 | x_1 x_2) = P(0|01) = \frac{6}{6}, \\ P(x_4 | x_1 x_2 x_3) &= P(1|010) = \frac{4}{10}, \quad P(x_5 | x_1 \dots x_4) = P(0|0101) = \frac{4}{4}, \\ P(x_6 | x_1 \dots x_5) &= P(0|01010) = \frac{3}{6}, \quad P(x_7 | x_1 \dots x_6) = P(1|010100) = \frac{6}{6}, \\ P(x_8 | x_1 \dots x_7) &= P(1|0101001) = \frac{2}{2}, \\ q(x_1) &= q(0) = 0, \quad q(x_2 | x_1) = q(1|0) = \frac{9}{14}, \quad q(x_3 | x_1 x_2) = q(0|01) = 0, \\ q(x_4 | x_1 x_2 x_3) &= q(1|010) = \frac{6}{10}, \quad q(x_5 | x_1 \dots x_4) = q(0|0101) = 0, \\ q(x_6 | x_1 \dots x_5) &= q(0|01010) = 0, \quad q(x_7 | x_1 \dots x_6) = q(1|010100) = 0, \\ q(x_8 | x_1 \dots x_7) &= q(1|0101001) = 0. \end{aligned} \quad (2.12)$$

Соответственно,

$$\begin{aligned} \rho_1^0 &= 1, \quad \rho_2^0 = \frac{5}{14}, \quad \rho_3^0 = 1, \quad \rho_4^0 = \frac{2}{5}, \quad \rho_5^0 = 1, \quad \rho_6^0 = \frac{1}{2}, \quad \rho_7^0 = 1, \quad \rho_8^0 = 1, \\ \lambda_1^0 &= 0, \quad \lambda_2^0 = \frac{9}{14}, \quad \lambda_3^0 = 0, \quad \lambda_4^0 = \frac{3}{5}, \quad \lambda_5^0 = 0, \quad \lambda_6^0 = 0, \quad \lambda_7^0 = 0, \quad \lambda_8^0 = 0. \end{aligned} \quad (2.13)$$

Затем по (2.8) получаем

$$\begin{aligned} \rho_1^1 &= \frac{5}{14}, & \rho_2^1 &= \frac{2}{5}, & \rho_3^1 &= \frac{1}{2}, & \rho_4^1 &= 1, & \lambda_1^1 &= \frac{9}{14}, & \lambda_2^0 &= \frac{3}{5}, & \lambda_3^0 &= 0, & \lambda_4^0 &= 0, \\ \rho_1^2 &= \frac{1}{7}, & \rho_2^2 &= \frac{1}{2}, & \lambda_1^2 &= \frac{6}{7}, & \lambda_2^2 &= 0, \\ \rho_1^3 &= \frac{1}{14}, & \lambda_1^3 &= \frac{6}{7}. \end{aligned} \quad (2.14)$$

По (2.9)

$$N'(01010011) = \lambda_1^3 \cdot |S_8| = 6/7 \cdot (C_8^4 - C_8^3) = 12.$$

Таким образом, мы получили $N'(w')$, номер слова 01010011 среди множества S_8 .

Поиск номера слова w'' среди множества A_m^n – это перевод числа из m -ичной системы в двоичную, буквы алфавита A_m цифрами m -ичной системы записи: α_0 это цифра, обозначающая нуль в этой системе, α_1 – цифра, обозначающая единицу, и т.д. Для нахождения двоичного представления m -ичного слова w'' применяем быстрый алгоритм перевода из m -ичной системы в двоичную из [4], основанный на принципе “разделяй и властвуй”. Обозначим двоичное представление m -ичного слова $x_1 \dots x_i$ через $(x_1 \dots x_i)_2$. На первом шаге находятся двоичные представления цифр, из которых состоит w'' : $(\alpha_0)_2 = 0$, $(\alpha_1)_2 = 1$, $(\alpha_2)_2 = 10$ и т.д. Зная двоичные представления двух m -ичных слов длины i $x_1x_2 \dots x_i$ и $x_{i+1}x_{i+2} \dots x_{2i}$, мы можем найти двоичное представление m -ичного слова $x_1x_2 \dots x_{2i}$ длины $2i$, составленного из первых двух, по формуле

$$(x_1x_2 \dots x_{2i})_2 = (x_1x_2 \dots x_i)_2 \cdot m^i + (x_{i+1}x_{i+2} \dots x_{2i})_2.$$

Используя эту формулу, находим по n двоичным представлениям букв, из которых состоит w'' , двоичные представления $n/2$ подслов длины 2, из которых состоит w'' , затем по ним по этой же формуле двоичные представления $n/4$ подслов длины 4, из которых состоит w'' , и продолжаем эти вычисления, пока не находим двоичное представление w'' , т.е. $N''(w'')$.

Очевидно, что для нашего примера такого перевода не требуется, так как система, соответствующая алфавиту A_2 , уже двоичная. $N''(\alpha_0\alpha_1\alpha_0\alpha_1) = (0101)_2 = 5$. Находим номер слова $()[] ([])$ по (2.1):

$$N(()[] ([])) = N''(\alpha_0\alpha_1\alpha_0\alpha_1) \cdot 14 + N'(01010011) = 5 \cdot 14 + 12 = 82.$$

Свойства данного метода описывает

ТЕОРЕМА 1. *Объем памяти, требуемый для кодирования слова длины $2n$ языка Дика над алфавитом $2t$ равен $O(n \log n)$ бит. Скорость кодирования слова длины $2n$ языка Дика над алфавитом $2t$, т.е. время, требуемое для кодирования одной буквы, равна $O(\log n M(n \log n)/n)$ битовых операций, где $M(k)$ время умножения двух слов длины k .*

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

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

Время вычисления $N(w)$ состоит из времени вычисления $N'(w')$, времени вычисления $N''(w'')$ и времени вычисления $N(w)$ по полученным $N'(w')$ и $N''(w'')$.

Найдем время вычисления $N'(w')$. Оно состоит из времени вычисления ρ_b^a и λ_b^a для $0 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$ и времени вычисления произведения $\lambda^{\log(2n)} |S_{2n}|$. Для вычисления ρ_b^0 , λ_b^0 , $1 \leq b \leq 2n$ требуется вычислить $2n$ значений $P(x_b | x_1 \dots x_{b-1})$ и $2n$ значений $P(\chi | x_1 \dots x_{b-1})$, $\chi < x_b$. Для вычисления каждого из этих значений по формуле (2.10) или (2.11) требуется совершить одно умножение чисел длины $\log(2n)$ для определения числителя и одно умножение чисел длины $\log(2n)$ для определения знаменателя.

Обозначим $M(n)$ время умножения двух чисел длины n .

Время вычисления ρ_b^0 и λ_b^0 при $1 \leq b \leq 2n$ равно $8nM(\log(2n))$.

Из (2.10) и (2.11) видно, что числители и знаменатели дробей $P(x_b | x_1 \dots x_{b-1})$, (и, очевидно, дробей $P(\chi | x_1 \dots x_{b-1})$, $\chi < x_b$) при $1 \leq b \leq 2n$ не превышают $4n^2$. Для записи числителя, так же как и для записи знаменателя этих дробей, требуется максимум $2 \log n + 2$ бит. Следовательно, по (2.8), для записи дробей λ_b^0 и ρ_b^0 при $1 \leq b \leq 2n$ достаточно $2 \log n + 2$ для числителя и столько же для знаменателя. Вычисление величин ρ_b^1 или λ_b^1 в соответствии с (2.8) при $1 \leq b \leq n$ потребует соответственно двух или трех операций умножения чисел, длина которых не превосходит $2 \log n + 2$ бит, а общее число операций умножения для вычисления всех λ_b^1 , ρ_b^1 при $1 \leq b \leq n$ равно $5n$. При вычислении λ_b^1 используется обычное равенство $a/b + c/d = (ad + bc)/(bd)$, требующее три умножения. В результате будут получены дроби, у которых требуется не более $2(2 \log n + 2)$ бит для записи числителя и столько же для знаменателя. Аналогично, для вычисления ρ_b^2 и λ_b^2 при $1 \leq b \leq n/2$ требуется $5n/2$ операций умножения над числами длины $2 \cdot 2 \log n + 4$ бит и так далее, для вычисления ρ_b^a и λ_b^a при $1 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$ требуется $5n/2^{a-1}$ операций умножения над числами длины $2^{a-1}(2 \log n + 2)$ бит.

Следовательно, общее время вычислений λ_b^a и ρ_b^a при $1 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$ выражается следующим образом:

$$5nM(2 \log n + 2) + \dots + \frac{5n}{2^{a-1}} M(2^{a-1}(2 \log n + 2)) + \dots + 5M(n(2 \log n + 2)). \quad (2.15)$$

Обозначим $M^*(n)$ время умножения двух чисел длины n , деленное на длину этих чисел: $M^*(n) = M(n)/n$. Тогда по (2.15) общее время вычислений λ_b^a и ρ_b^a :

$$10n \log n M^*(2 \log n) + 10n \log n M^*(4 \log n) + \dots + 10n \log n M^*(2n \log n). \quad (2.16)$$

В этой сумме $\log(2n)$ слагаемых, из них каждое не больше $10n \log n M^*(2n \log n)$.

Таким образом, время вычисления дробей λ_b^a и ρ_b^a равно

$$O((\log n + 1)10n \log n M^*(n 2 \log n)) = O\left(n \log^2 n \frac{M(n 2 \log n)}{n(2 \log n)}\right) = O(\log n M(n 2 \log n)). \quad (2.17)$$

Время вычисления произведения $\lambda^{\log(2n)} |S_{2n}|$ состоит из времени вычисления произведения числителя $\lambda^{\log(2n)}$ и $|S_{2n}|$ и времени вычисления частного полученного числа и знаменателя $\lambda^{\log(2n)}$. Количество знаков, нужных для записи числителя $\lambda^{\log(2n)}$,

не превышает $2n \cdot 2 \log n$. Количество знаков, нужных для записи $|S_{2n}|$, не превышает $2n$, так как количество двоичных последовательностей длины $2n$, соответствующих правильным расстановкам скобок ($|S_{2n}|$), меньше, чем количество всех двоичных последовательностей длины $2n$. Таким образом, время умножения числителя $\lambda^{\log(2n)}$ и $|S_{2n}|$ равно $M(2n \cdot 2 \log n)$. Длина полученного числа не превышает $4n(2 \log n)$. Длина знаменателя $\lambda^{\log(2n)}$ не превышает $2n \cdot 2 \log n$. Так как время деления двух чисел длины a равно времени умножения двух чисел длины a , по [7] время деления полученного числа и знаменателя $\lambda^{\log(2n)}$ равно $M(4n \cdot 2 \log n)$.

Таким образом, время вычисления $N'(w')$ равно сумме времени вычисления ρ_b^0 и λ_b^0 при $1 \leq b \leq 2n$, т.е. $8nM(\log(2n))$, времени вычисления λ_b^a и ρ_b^a при $1 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$, т.е. $O(\log n M(n \cdot 2 \log n))$ и времени вычисления $\lambda^{\log(2n)}|S_{2n}|$, т.е. $M(2n \cdot 2 \log n) + M(4n \cdot 2 \log n)$:

$$\begin{aligned} & 8nM(\log(2n)) + O(\log n M(n \cdot 2 \log n)) + M(2n \cdot 2 \log n) + M(4n \cdot 2 \log n) \\ &= 8n \log(2n) M^*(\log(2n)) + O(\log n n(2 \log n) M^*(n \cdot 2 \log n)) + O(M(4n \cdot 2 \log n)) \\ &= O(8n \log(2n) M^*(\log(2n)) + \log n n(2 \log n) M^*(n \cdot 2 \log n) + M(4n \cdot 2 \log n)) \\ &= O(n \log n(2 \log n) M^*(4n \cdot 2 \log n)) = O(\log n M(n \log n)). \end{aligned} \quad (2.18)$$

Определим теперь время вычисления $N''(w'')$. Для вычисления каждого из значений $(x_1 x_2)_2, \dots, (x_{n-1} x_n)_2$ требуется совершить по одному умножению чисел m и x_i при i , равных $1, 3, \dots, n-1$, и по одному сложению полученных произведений и чисел x_{i+1} . Количество знаков, требуемое для записи числа m , как и для записи x_i , не превышает $\log m$. Поэтому время их умножения $M(\log m) = O(1)$. Длина полученного произведения не превышает $2 \log m$. Длина x_{i+1} не превышает $\log m$. Поэтому время их сложения не превышает $2 \log m = O(1)$. Таким образом, для вычисления всех $(x_1 x_2)_2, \dots, (x_{n-1} x_n)_2$ требуется совершить n операций, занимающих время $O(1)$. Общее время $O(n)$. Аналогично, для вычислений $(x_1 x_2 x_3 x_4)_2, \dots, (x_{n-3} x_{n-2} x_{n-1} x_n)_2$ требуется совершить $n/4$ операций, занимающих время $M(2 \log m) = O(1)$, и $n/4$ операций, занимающих время $4 \log m = O(1)$. Общее время $O(n)$, и т.д.; для вычислений $(x_1 \dots x_{2^k})_2, \dots, (x_{n-2^{k-1}} \dots x_n)_2$ при $1 \leq k \leq \log n$ требуется совершить $n/2^k$ операций, занимающих время $M(2^{k-1} \log m)$, и $n/2^k$ операций, занимающих время $2^k \log m$. Общее время

$$\frac{n}{2^k} \cdot M(2^{k-1} \log m) + 2^k \log m.$$

Общее время вычисления $N''(w'')$:

$$\begin{aligned} & \frac{n}{2} (M(\log m) + 2 \log m) + \frac{n}{4} (M(2 \log m) + 4 \log m) + \dots + \left(M\left(\frac{n}{2} \log m\right) + n \log m \right) \\ &= \frac{n}{2} \cdot (\log m M^*(\log m) + 2 \log m) + \frac{n}{4} \cdot (2 \log m M^*(2 \log m) + 4 \log m) + \dots \\ & \quad \dots + \left(\frac{n}{2} \log m M^*\left(\frac{n}{2} \log m\right) + n \log m \right). \end{aligned} \quad (2.19)$$

Всего в этом выражении $\log n$ слагаемых, каждое из которых не превышает

$$\frac{n}{2} \log m M^*\left(\frac{n}{2} \log m\right) + n \log m,$$

таким образом время, требуемое для вычисления $N''(w'')$, равно

$$\log n \left(\frac{n}{2} \log m M^* \left(\frac{n}{2} \log m \right) + n \log m \right) = O(\log n M(n)). \quad (2.20)$$

Определим время, требуемое для вычисления $N(w)$ по $N'(w')$ и $N''(w'')$ по (2.1). Количество знаков, требуемое для записи $N''(w'')$, не превосходит $n \log m$. Количество знаков, требуемое для записи числа $|S_{2n}|$, не превосходит $2n$. Для умножения этих чисел требуется время $M(n \log m) = O(M(n))$. Количество знаков, требуемое для записи полученного числа, не превосходит $n \log m$. Количество знаков, требуемое для записи $N'(w')$, не превосходит $2n$. Для сложения полученного числа и $N'(w')$ требуется время $n \log m = O(n)$. Таким образом, общее время, требуемое для вычисления $N(w)$ по $N'(w')$ и $N''(w'')$, равно

$$O(M(n)) + O(n) = O(M(n)). \quad (2.21)$$

По (2.18), (2.20), (2.21) общее время определения номера w :

$$O(\log n M(n \log n)) + O(\log n M(n)) + O(M(n)) = O(\log n M(n \log n)).$$

Сложность вычисления номера слова w на один его символ $O(\log n / n M(n \log n))$.

Оценим необходимый для осуществления нумерации объем памяти. Для определения $N'(w')$ при вычислении величин λ_b^a и ρ_b^a при $1 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$ используются только величины λ_b^{a-1} и ρ_b^{a-1} , $1 < a \leq \log n$, $1 \leq b \leq 2n/2^a$. Поэтому для нумерации достаточно иметь память для хранения наборов λ_b^a , ρ_b^a , $1 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$, и λ_b^{a+1} , ρ_b^{a+1} , $1 < a \leq \log n$, $1 \leq b \leq 2n/2^a$. Длина каждой дроби λ_b^a и ρ_b^a не превосходит $2^{a+1}(2 \log n + 2)$. Отсюда объем памяти, требуемый для определения номера слова w' , не превосходит $O(n \log n)$.

Для определения $N''(w'')$ при вычислении $(x_1 \dots x_{2^k})_2, \dots, (x_{n-2^{k+1}} \dots x_n)_2$, $1 \leq k \leq \log n$, используются только величины $(x_1 \dots x_{2^{k-1}})_2, \dots, (x_{n-2^{k-1}+1} \dots x_n)_2$. Поэтому для нумерации достаточно иметь память для хранения наборов

$$(x_1 \dots x_{2^k})_2, \dots, (x_{n-2^{k+1}} \dots x_n)_2$$

и $(x_1 \dots x_{2^{i-1}})_2, \dots, (x_{n-2^{k-1}+1} \dots x_n)_2$, $1 \leq k \leq \log n$. Длина каждого числа

$$(x_1 \dots x_{2^k})_2, \dots, (x_{n-2^{k+1}} \dots x_n)_2$$

равна $2^k \log m$; всего таких чисел $n/2^k$. Длина каждого числа

$$(x_1 \dots x_{2^{k-1}})_2, \dots, (x_{n-2^{k-1}+1} \dots x_n)_2$$

равна $2^{k-1} \log m$; всего таких чисел $n/2^{k-1}$. Отсюда объем памяти, требуемый для вычисления $N''(w'')$, не превосходит $O(n \log m) = O(n)$.

Общий объем памяти для вычисления $N(w)$ равен $O(n \log n) + O(n) = O(n \log n)$. Теорема доказана.

3. Алгоритм денумерации. Опишем алгоритм декодирования. Сделаем это на примере поиска слова из множества D_8^4 по номеру $N = 82$.

По номеру N находятся $N'(w')$ и $N''(w'')$: $N'(w') = N \bmod |S|$, $N''(w'') = \lfloor N/|S| \rfloor$.

В нашем примере $N'(w') = 82 \bmod 14 = 12$, $N''(w'') = \lfloor 82/14 \rfloor = 5$.

Найдем по $N'(w')$ слово w' из множества S_{2n} .

Для описания алгоритма введем вспомогательные функции для верхней и нижней оценки λ_b^a . Пусть p/q – рациональное число, представленное как пара целых положительных чисел p, q , $p \leq q$, и пусть $t > 1$ – целое число. Положим $l = \lfloor \log q \rfloor$. Пусть $(q_l q_{l-1} \dots q_0)$ и $(p_l p_{l-1} \dots p_0)$ – двоичные представления чисел q и p . Тогда определим $\phi_t^+(p/q)$ и $\phi_t^-(p/q)$ следующим образом:

$$\phi_t^+\left(\frac{p}{q}\right) = \frac{\sum_{i=l-t}^l p_i 2^i + 2^{l-t}}{\sum_{i=l-t}^l q_i 2^i}, \quad \phi_t^-\left(\frac{p}{q}\right) = \frac{\sum_{i=l-t}^l p_i 2^i}{\sum_{i=l-t}^l q_i 2^i + 2^{l-t}}. \quad (3.1)$$

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

Например,

$$\phi_3^+\left(\frac{9}{17}\right) = \frac{5}{8}, \quad \phi_3^-\left(\frac{9}{17}\right) = \frac{4}{9}.$$

Пусть $q_{\max}$ равно максимальному знаменателю чисел

$$\frac{N_{S_{2n}}(x_1 \dots x_{i+1})}{N_{S_{2n}}(x_1 \dots x_i)}, \quad x_1 \dots x_{2n} \in S_{2n}, \quad 1 \leq i \leq 2n - 1.$$

Из (2.10) и (2.11) следует, что

$$q_{\max} = 4n^2. \quad (3.2)$$

Из (2.8) получаем, что знаменатели рациональных дробей λ_b^a и ρ_b^a не превосходят $q_{\max}^{2^a}$ при всех $1 \leq b \leq 2n/2^a$ и, следовательно,

$$\rho_b^a \geq \frac{1}{q_{\max}^{2^a}}. \quad (3.3)$$

Первый шаг поиска слова w' из множества S_{2n} по данному номеру $N'(w')$ заключается в вычислении оценок $\lambda^+(\log(2n), 1)$, $\lambda^-(\log(2n), 1)$ по формулам

$$\begin{aligned} \lambda^+(\log(2n), 1) &= \phi_{8n \lceil \log q_{\max} \rceil + 4}^+\left(\frac{N'(w')}{|S_{2n}|}\right), \\ \lambda^-(\log(2n), 1) &= \phi_{8n \lceil \log q_{\max} \rceil + 4}^-\left(\frac{N'(w')}{|S_{2n}|}\right). \end{aligned} \quad (3.4)$$

Опишем рекурсивную процедуру получения по известным префиксу $x_1 \dots x_{2^a(b-1)}$ и оценкам $\lambda^+(a, b)$, $\lambda^-(a, b)$ ($0 < a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$) 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 . Причем, если $b = 2n/2^a$, то выполняя эту процедуру, можно однозначно определить подслово $x_{2n-2^a+1} \dots x_{2n}$. (В этом случае находить λ_b^a , ρ_b^a уже не требуется, так как, имея известный префикс $x_1 \dots x_{2n-2^a}$ и определив подслово $x_{2n-2^a+1} \dots x_{2n}$, мы найдем искомое слово $x_1 \dots x_{2n}$.)

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

$$\begin{aligned}\lambda^+(a-1, 2b-1) &= \phi_{2^{a+1} \lceil \log q_{\max} \rceil + 4}^+(\lambda^+(a, b)), \\ \lambda^-(a-1, 2b-1) &= \phi_{2^{a+1} \lceil \log q_{\max} \rceil + 4}^-(\lambda^-(a, b)).\end{aligned}\quad (3.5)$$

Если $a-1 > 0$, то выполняем рекурсивную процедуру 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 $x''_{2^{a-1}(2b-2)+1} \dots x''_{2^{a-1}(2b-1)}$, а соответствующие им предположительные λ_{2b-1}^{a-1} , ρ_{2b-1}^{a-1} обозначим через $\lambda_{2b-1}'^{a-1}$, $\lambda_{2b-1}''^{a-1}$, $\rho_{2b-1}'^{a-1}$ и $\rho_{2b-1}''^{a-1}$.

Если $a-1 = 0$, то находим такие буквы χ , $\chi \in \{0, 1\}$, для которых одновременно выполнялись бы условия

$$\begin{aligned}P(\chi|x_1 \dots x_{2b-2}) &> 0, \\ \lambda^+(0, 2b-1) &\geq q(\chi|x_1 \dots x_{2b-2}), \\ \lambda^-(0, 2b-1) &< q(\chi|x_1 \dots x_{2b-2}) + P(\chi|x_1 \dots x_{2b-2}).\end{aligned}\quad (3.6)$$

Если существует одно такое χ , то $x_{2b-1} = \chi$. Если существует два таких χ , считаем x'_{2b-1} равным меньшему такому χ , т.е. нулю, x''_{2b-1} большему такому χ , т.е. единице. В первом случае вычисляем значения λ_{2b-1}^0 и ρ_{2b-1}^0 по формулам (2.8), во втором случае значения λ_{2b-1}^0 , ρ_{2b-1}^0 , $\lambda_{2b-1}''^0$ и $\rho_{2b-1}''^0$ по следующим формулам:

$$\begin{aligned}\lambda_{2b-1}^0 &= q(x'_{2b-1}|x_1 \dots x_{2b-2}), \\ \lambda_{2b-1}''^0 &= q(x''_{2b-1}|x_1 \dots x_{2b-2}), \\ \rho_{2b-1}^0 &= P(x'_{2b-1}|x_1 \dots x_{2b-2}), \\ \rho_{2b-1}''^0 &= P(x''_{2b-1}|x_1 \dots x_{2b-2}).\end{aligned}\quad (3.7)$$

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 $\lambda_{2b-1}'^{a-1}$, $\rho_{2b-1}'^{a-1}$, $\lambda_{2b-1}''^{a-1}$ и $\rho_{2b-1}''^{a-1}$. Во втором случае проверяем, выполняются ли следующие неравенства:

$$\begin{aligned}\lambda_{2b-1}^{a-1} + \rho_{2b-1}'^{a-1} &\leq \lambda^-(a, b), \\ \lambda_{2b-1}''^{a-1} &> \lambda^+(a, b).\end{aligned}\quad (3.8)$$

Если выполняется первое неравенство, то

$$x_1 \dots x_{2^{a-1}(2b-2)} = x''_1 \dots x''_{2^{a-1}(2b-2)}, \quad \lambda_{2b-1}^{a-1} = \lambda_{2b-1}''^{a-1}, \quad \rho_{2b-1}^{a-1} = \rho_{2b-1}'^{a-1}.$$

Если выполняется второе неравенство, то

$$x_1 \dots x_{2^{a-1}(2b-2)} = x'_1 \dots x'_{2^{a-1}(2b-2)}, \quad \lambda_{2b-1}^{a-1} = \lambda_{2b-1}'^{a-1}, \quad \rho_{2b-1}^{a-1} = \rho_{2b-1}''^{a-1}.$$

Если в результате сравнений (3.8) 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

$$\begin{aligned}\lambda^+(a-1, 2b) &= \phi_{2^{a+1} \lceil \log q_{\max} \rceil + 4}^+ \left(\frac{\lambda^+(a, b) - \lambda_{2b-1}^{a-1}}{\rho_{2b-1}^{a-1}} \right), \\ \lambda^-(a-1, 2b) &= \phi_{2^{a+1} \lceil \log q_{\max} \rceil + 4}^- \left(\frac{\lambda^-(a, b) - \lambda_{2b-1}^{a-1}}{\rho_{2b-1}^{a-1}} \right),\end{aligned}\quad (3.9)$$

Применяем к ним рекурсивную процедуру, если $a-1 > 0$, 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 $b = 2n/2^a$, только подслово $x_{2^{a-1}(2b-1)+1} \dots x_{2^{a-1}2b}$) или пару подслов, такую, что одно слово 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 $a-1 = 0$, выполняем над полученными оценками действия, аналогичные действиям над оценками λ_{2b-1}^0 , ρ_{2b-1}^0 ; в результате находим x_{2b} , λ_{2b}^0 , ρ_{2b}^0 или пары букв x'_{2b} и x''_{2b} , ($x'_{2b} < x''_{2b}$) и пар значений λ_{2b}^0 , ρ_{2b}^0 , $\lambda_{2b}^{\prime 0}$, $\rho_{2b}^{\prime 0}$. Если же $b = 2n/2^a$, т.е. $a = 0$, $b = 2n$ и мы нашли буквы x'_{2n} и x''_{2n} , то это означает, что $x_{2n} = x''_{2n}$.

Если ни одно из неравенств (3.8) не выполняется, то если $b = 2n/2^a$, 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 , $2^{a-1}(2b-1) + 1 \leq i \leq 2^{a-1}2b$, такие, что

$$\begin{aligned}q(x_i | x_1 \dots x_{i-1}) &= 0, \\ P(x_i | x_1 \dots x_{i-1}) &> 0.\end{aligned}\quad (3.10)$$

Если ни одно из неравенств (3.8) не выполняется, при этом $b \neq 2n/2^a$, то находим последовательно буквы x'_i , $2^{a-1}(2b-1) + 1 \leq i \leq 2^{a-1}2b$, такие, что

$$\begin{aligned}q(x'_i | x_1 \dots x_{2^a(b-1)} x'_{2^a(b-1)+1} \dots x'_{i-1}) + P(x'_i | x_1 \dots x_{2^a(b-1)} x'_{2^a(b-1)+1} \dots x'_{i-1}) &= 1, \\ P(x'_i | x_1 \dots x_{2^a(b-1)} x'_{2^a(b-1)+1} \dots x'_{i-1}) &> 0,\end{aligned}\quad (3.11)$$

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

$$\begin{aligned}q(x''_i | x_1 \dots x_{2^a(b-1)} x''_{2^a(b-1)+1} \dots x''_{i-1}) &= 0, \\ P(x''_i | x_1 \dots x_{2^a(b-1)} x''_{2^a(b-1)+1} \dots x''_{i-1}) &> 0.\end{aligned}\quad (3.12)$$

Считаем рекурсивно по формулам, аналогичным (2.8), пару λ_{2b}^{a-1} , ρ_{2b}^{a-1} (в формулах используется префикс $x_1 \dots x_{2^a(b-1)} x'_{2^a(b-1)+1} \dots x'_{2^a b}$) и пару чисел $\lambda_{2b}^{\prime a-1}$, $\rho_{2b}^{\prime a-1}$ (в формулах используется префикс $x_1 \dots x_{2^a(b-1)} x''_{2^a(b-1)+1} \dots x''_{2^a b}$).

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 $x_{2^a(b-1)+1} \dots x_{2^a b}$. Если $b \neq 2n/2^a$, вычисляем значения λ_b^a и ρ_b^a по формулам

$$\lambda_b^a = \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1} \cdot \lambda_{2b}^{a-1}, \quad \rho_b^a = \rho_{2b-1}^{a-1} \cdot \rho_{2b}^{a-1}. \quad (3.13)$$

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 $x'_{2^a(b-1)+1} \dots x'_{2^a b}$ считаем равным конкатенации

$$x_{2^{a-1}(2b-2)+1} \dots x_{2^{a-1}(2b-1)} \quad \text{и} \quad x'_{2^{a-1}(2b-1)+1} \dots x'_{2^{a-1}2b},$$

слово $x''_{2^a(b-1)+1} \dots x''_{2^ab}$ считаем равным конкатенации

$$x_{2^{a-1}(2b-2)+1} \dots x_{2^{a-1}(2b-1)} \quad \text{и} \quad x''_{2^{a-1}(2b-1)+1} \dots x''_{2^{a-1}2b}.$$

Вычисляем значения λ_b^a , ρ_b^a , $\lambda_b^{\prime a}$, $\rho_b^{\prime a}$ по формулам

$$\begin{aligned} \lambda_b^a &= \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1} \cdot \lambda_{2b}^{a-1}, & \rho_b^a &= \rho_{2b-1}^{a-1} \cdot \rho_{2b}^{a-1}, \\ \lambda_b^{\prime a} &= \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1} \cdot \lambda_{2b}^{\prime a-1}, & \rho_b^{\prime a} &= \rho_{2b-1}^{a-1} \cdot \rho_{2b}^{\prime a-1}. \end{aligned} \quad (3.14)$$

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

$$x'_{2^{a-1}(2b-2)+1} \dots x'_{2^{a-1}(2b-1)} \quad \text{и} \quad x'_{2^{a-1}(2b-1)+1} \dots x'_{2^{a-1}2b},$$

а слово $x''_{2^a(b-1)+1} \dots x''_{2^ab}$ – равным конкатенации

$$x''_{2^{a-1}(2b-2)+1} \dots x''_{2^{a-1}(2b-1)} \quad \text{и} \quad x''_{2^{a-1}(2b-1)+1} \dots x''_{2^{a-1}2b}.$$

Вычисляем значения λ_b^a , ρ_b^a , $\lambda_b^{\prime a}$, $\rho_b^{\prime a}$ по формулам

$$\begin{aligned} \lambda_b^a &= \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1} \cdot \lambda_{2b}^{a-1}, & \rho_b^a &= \rho_{2b-1}^{a-1} \cdot \rho_{2b}^{a-1}, \\ \lambda_b^{\prime a} &= \lambda_{2b-1}^{a-1} + \rho_{2b-1}^{a-1} \cdot \lambda_{2b}^{\prime a-1}, & \rho_b^{\prime a} &= \rho_{2b-1}^{a-1} \cdot \rho_{2b}^{\prime a-1}. \end{aligned} \quad (3.15)$$

Этим завершается описание рекурсивной процедуры.

Применяя описанную процедуру для $a = \log(2n)$, $b = 1$, находим по оценкам $\lambda^+(\log(2n), 1)$, $\lambda^-(\log(2n), 1)$ слово $x_1 \dots x_{2n}$. (При этом считаем, что нам известен префикс нулевой длины.)

Рассмотрим пример. Найдем слово w' длины $n = 8$ по данному номеру $N'(w') = 12$. Находим $q_{\max} = 4n^2 = 64$. Находим $\lambda^+(3, 1)$, $\lambda^-(3, 1)$ по формулам (3.4):

$$\lambda^+(3, 1) = \phi_{196}^+ \frac{12}{14} = \frac{12 \cdot 2^{193} + 1}{14 \cdot 2^{193}}, \quad \lambda^-(3, 1) = \frac{12 \cdot 2^{193}}{14 \cdot 2^{193} + 1}.$$

Находим $\lambda^+(2, 1)$, $\lambda^-(2, 1)$, а затем $\lambda^+(1, 1)$, $\lambda^-(1, 1)$, $\lambda^+(0, 1)$, $\lambda^-(0, 1)$ по формулам (3.5):

$$\begin{aligned} \lambda^+(2, 1) &= \phi_{100}^+ \frac{12 \cdot 2^{193} + 1}{14 \cdot 2^{193}} = \frac{12 \cdot 2^{97} + 1}{14 \cdot 2^{97}}, & \lambda^-(2, 1) &= \frac{12 \cdot 2^{97}}{14 \cdot 2^{97} + 1}, \\ \lambda^+(1, 1) &= \phi_{52}^+ \frac{12 \cdot 2^{97} + 1}{14 \cdot 2^{97}} = \frac{12 \cdot 2^{49} + 1}{14 \cdot 2^{49}}, & \lambda^-(1, 1) &= \frac{12 \cdot 2^{49}}{14 \cdot 2^{49} + 1}, \\ \lambda^+(0, 1) &= \phi_{28}^+ \frac{12 \cdot 2^{49} + 1}{14 \cdot 2^{49}} = \frac{12 \cdot 2^{25} + 1}{14 \cdot 2^{25}}, & \lambda^-(0, 1) &= \frac{12 \cdot 2^{25}}{14 \cdot 2^{25} + 1}. \end{aligned}$$

Находим χ такие, для которых выполняются неравенства (3.6), т.е.

$$\lambda^+(0, 1) \geq q(\chi), \quad \lambda^-(0, 1) < q(\chi) + P(\chi), \quad P(\chi) > 0.$$

Единственное χ , которое удовлетворяет этим неравенствам, это нуль; можно видеть, что

$$\lambda^+(0, 1) > q(0) = 0, \quad \lambda^-(0, 1) < q(0) + P(0) = 1, \quad P(0) = 1 > 0,$$

в то же время для единицы эти неравенства не выполняются, так как $P(1) = 0$. Поэтому $x_1 = 0$.

Находим λ_1^0 и ρ_1^0 по формулам (2.8):

$$\lambda_1^0 = q(0) = 0, \quad \rho_1^0 = P(0) = 1.$$

Находим $\lambda^+(0, 2)$, $\lambda^-(0, 2)$ по формулам (3.9):

$$\lambda^+(0, 2) = \phi_{28}^+ \frac{(12 \cdot 2^{49} + 1)/(14 \cdot 2^{49}) - 0}{1} = \frac{12 \cdot 2^{25} + 1}{14 \cdot 2^{25}}, \quad \lambda^-(0, 2) = \frac{12 \cdot 2^{25}}{14 \cdot 2^{25} + 1}.$$

Находим χ такие, для которых выполняются неравенства (3.6), т.е.

$$\lambda^+(0, 2) \geq q(\chi|0), \quad \lambda^-(0, 2) < q(\chi|0) + P(\chi|0), \quad P(\chi|0) > 0.$$

Неравенства выполняются для $\chi = 1$, так как

$$\lambda^+(0, 2) > q(1|0) = \frac{5}{14}, \quad \lambda^-(0, 2) < q(1|0) + P(1|0) = 1, \quad P(1|0) = \frac{5}{14} > 0;$$

неравенства не выполняются для $\chi = 0$, так как

$$\lambda^-(0, 2) > q(0|0) + P(0|0) = \frac{9}{14}.$$

Поэтому $x_2 = 1$.

Находим λ_2^0 и ρ_2^0 , а затем λ_1^1 и ρ_1^1 по формулам (2.8):

$$\begin{aligned} \lambda_2^0 &= q(1|0) = \frac{9}{14}, & \rho_2^0 &= P(1|0) = \frac{5}{14}, \\ \lambda_1^1 &= \lambda_1^0 + \lambda_2^0 \cdot \rho_1^0 = \frac{9}{14}, & \rho_1^1 &= \rho_1^0 \cdot \rho_2^0 = \frac{5}{14}. \end{aligned}$$

Находим $\lambda^+(1, 2)$, $\lambda^-(1, 2)$ по формулам (3.9):

$$\lambda^+(1, 2) = \frac{3 \cdot 2^{50} + 1}{5 \cdot 2^{50}}, \quad \lambda^-(1, 2) = \frac{3 \cdot 2^{50}}{5 \cdot 2^{50} + 1}.$$

Находим $\lambda^+(0, 3)$, $\lambda^-(0, 3)$, по формулам (3.5):

$$\lambda^+(0, 3) = \frac{3 \cdot 2^{26} + 1}{5 \cdot 2^{26}}, \quad \lambda^-(0, 3) = \frac{3 \cdot 2^{26}}{5 \cdot 2^{26} + 1}.$$

Находим χ такие, для которых выполняются неравенства (3.6), т.е.

$$\lambda^+(0, 3) \geq q(\chi|01), \quad \lambda^-(0, 3) < q(\chi|01) + P(\chi|01), \quad P(\chi|01) > 0.$$

Неравенства выполняются для $\chi = 0$, так как

$$\lambda^+(0, 3) > q(0|01) = 0, \quad \lambda^-(0, 3) < q(0|01) + P(0|01) = 1, \quad P(0|01) = 1 > 0,$$

неравенства не выполняются для $\chi = 1$, так как $P(1|01) = 0$. Поэтому $x_3 = 0$.

Находим λ_3^0 и ρ_3^0 по формулам (2.8):

$$\lambda_3^0 = q(0|01) = 0, \quad \rho_3^0 = P(0|01) = 1.$$

Находим $\lambda^+(0, 4)$, $\lambda^-(0, 4)$ по формулам (3.9):

$$\lambda^+(0, 4) = \frac{3 \cdot 2^{26} + 1}{5 \cdot 2^{26}}, \quad \lambda^-(0, 4) = \frac{3 \cdot 2^{26}}{5 \cdot 2^{26} + 1}.$$

Находим χ такие, для которых выполняются неравенства (3.6), т.е.

$$\lambda^+(0, 4) \geq q(\chi|010), \quad \lambda^-(0, 4) < q(\chi|010) + P(\chi|010), \quad P(\chi|010) > 0.$$

Неравенства выполняются для $\chi = 0$, так как

$$\lambda^+(0, 4) > q(0|010) = 0, \quad \lambda^-(0, 4) < q(0|010) + P(0|010) = \frac{3}{5}, \quad P(0|010) = \frac{3}{5} > 0,$$

неравенства также выполняются для $\chi = 1$, так как

$$\lambda^+(0, 4) > q(1|010) = \frac{3}{5}, \quad \lambda^-(0, 4) < q(1|010) + P(1|010) = 1, \quad P(1|010) = \frac{2}{5} > 0.$$

Таким образом, $x'_4 = 0$, $x''_4 = 1$. Находим λ_4^0 , ρ_4^0 , $\lambda_4''^0$, $\rho_4''^0$ по формулам (3.7):

$$\lambda_4^0 = q(0|010) = 0, \quad \rho_4^0 = P(0|010) = \frac{3}{5}, \quad \lambda_4''^0 = q(1|010) = \frac{3}{5}, \quad \rho_4''^0 = P(1|010) = \frac{2}{5}.$$

Находим λ_2^1 , ρ_2^1 , $\lambda_2''^1$, $\rho_2''^1$, а затем λ_1^2 , ρ_1^2 , $\lambda_1''^2$, $\rho_1''^2$ по формулам (3.14):

$$\begin{aligned} \lambda_2^1 &= 0, \quad \rho_2^1 = \frac{3}{5}, \quad \lambda_2''^1 = \frac{3}{5}, \quad \rho_2''^1 = \frac{2}{5}, \\ \lambda_1^2 &= \frac{9}{14}, \quad \rho_1^2 = \frac{3}{14}, \quad \lambda_1''^2 = \frac{12}{14}, \quad \rho_1''^2 = \frac{2}{14}. \end{aligned}$$

Производим сравнения (3.8). Обнаруживаем, что одновременно выполняются оба неравенства: $\lambda_1''^2 + \rho_1^2 > \lambda^-(3, 1)$, $\lambda_1''^2 \leq \lambda^+(3, 1)$.

При этом выполняется равенство $1 = (2n)/2^3$, т.е. $b = 2n/2^a$. Поэтому буква $x_4 = x_4'' = 1$, буквы $x_5, \dots, x_8$ 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.

Теперь переводим номер слова $N''(w'')$ из двоичной системы в m -ичную с символами алфавита A_m в качестве цифр системы. Алгоритм основан на том, что по

двоичному представлению числа m -ичной системы находятся двоичные представления его половин:

$$\begin{aligned} N(x_1 x_2 \dots x_{n/2}) &= \left\lfloor \frac{N(x_1 \dots x_n)}{(m^{n/2})} \right\rfloor, \\ N(x_{n/2+1} \dots x_n) &= N(x_1 \dots x_n) \bmod m^{n/2}, \\ N(x_1 x_2 \dots x_{n/4}) &= \left\lfloor \frac{N(x_1 \dots x_{n/2})}{(m^{n/4})} \right\rfloor, \\ N(x_{n/4+1} \dots x_{n/2}) &= N(x_1 \dots x_{n/2}) \bmod m^{n/4}, \\ N(x_{n/2+1} \dots x_{3n/4}) &= \left\lfloor \frac{N(x_{n/2+1} \dots x_n)}{(m^{n/4})} \right\rfloor, \\ N(x_{3n/4+1} \dots x_n) &= N(x_{n/2+1} \dots x_n) \bmod m^{n/4}, \end{aligned} \quad (3.16)$$

и т.д. Это действие повторяется $\log n$ раз, в результате чего мы получаем все знаки числа $w'' = (x_1 x_2 \dots x_n)_m$, которые заменяем соответствующими им буквам алфавита A_m . Если полученный результат имеет меньше n знаков, слева дописываются символы, соответствующие нулям. Получаем слово w'' . Так как в нашем примере $m = 2$, такого перевода специально осуществлять не надо. Получаем

$$N''(w'') = (101)_2, \quad w'' = \alpha_0 \alpha_1 \alpha_0 \alpha_1. \quad (3.17)$$

Теперь в слове w' заменяем нули на открывающие скобки m типов в порядке, заданном словом w'' . Единицы заменяем на закрывающие скобки m типов таким образом, чтобы полученное слово было правильной расстановкой скобок m типов. Этот способ однозначно определяется типами открывающих скобок. В нашем случае получаем $() [] ([])$. Денумеруемое слово w найдено.

Свойства алгоритма денумерации описывает следующая теорема.

ТЕОРЕМА 2. *Объем памяти, требуемый для денумерации слова длины $2n$, 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 3. *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 4. *При использовании алгоритма быстрого умножения Фюрера, имеющего сложность $M(n) = n \log n 2^{O(\log^* n)}$, скорость денумерации равна*

$$O(\log^3 n 2^{O(\log^* n)}).$$

При вычислении слова w' по его номеру декодирование левой половины $x_1, x_2, \dots, x_n$ и правой половины $x_{n+1}, \dots, x_{2n}$ проводятся независимо и могут осуществляться при использовании одной и той же памяти. То же верно и для вычисления слова w'' по его номеру. Легко получить, что объемы памяти, используемые при кодировании и декодировании слова w , асимптотически совпадают.

Общий объем памяти для вычисления слова w по его номеру равен $O(n \log n)$.

Время, требуемое для определения слова w по его номеру $N(w)$, состоит из времени определения $N'(w')$ и $N''(w'')$ по $N(w)$, из времени определения слова w' по $N'(w')$ и времени определения слова w'' по $N''(w'')$.

Оценим время, требуемое для вычисления $N'(w')$ и $N''(w'')$ по $N(w)$. Для вычисления $N''(w'')$ требуется совершить одну операцию деления над числами $N''(w'')$ и $|S_{2n}|$, т.е. над числами длин $n \log m$ и $2n$. Это требует времени $M(n \log m) = O(M(n))$. Для вычисления $N'(w') = N(w) - |S_{2n}| \cdot N''(w'')$ требуется совершить одну операцию умножения чисел длин $n \log m = O(n)$ и $2n = O(n)$ и одну операцию вычитания чисел длин $n(\log m) = O(n)$. Для этого требуется время $O(M(n)) + O(n) = O(M(n))$.

Оценим время, требуемое для вычисления слова w' по $N'(w')$. При вычислении $\lambda^+(a, b)$ и $\lambda^-(a, b)$ по формулам (3.9) требуется совершить две операции деления, причем длина числителя и знаменателя делимого не превосходит

$$2^{a+3} \lceil \log q_{\max} \rceil + 4 = 2^{a+3}(2 \log n + 2) + 4$$

бит, а делителя

$$2^{a+1} \lceil \log q_{\max} \rceil = 2^{a+1}(2 \log n + 2)$$

бит. Длины делимого и делителя пропорциональны длинам сомножителей, используемых при вычислении λ_b^a по (2.8). Следовательно, время вычисления $\lambda^+(a, b)$ и $\lambda^-(a, b)$ будет пропорционально времени вычисления λ_b^a при нумерации. Кроме того, при денумерации вычисляются таким же образом, как и при нумерации, значения величин λ_b^a и ρ_b^a , или пары равных им по длине значений $\lambda_b'^a, \lambda_b''^a, \rho_b'^a, \rho_b''^a$. Кроме того, для сравнения $\lambda_b'^a + \rho_b'^a$ с $\lambda^+(a+1, (b-1)/2)$ и $\lambda_b''^a$ с $\lambda^-(a+1, (b-1)/2)$ требуется сравнить числа, длины которых не превышают $2^{a+2} \lceil \log q_{\max} \rceil + 4$. Время сравнения линейно зависит от длины сравниваемых чисел, что асимптотически меньше времени их умножения и деления. Следовательно, время, затрачиваемое на выполнение операций, необходимых для вычисления оценок $\lambda^+(a, b)$, $\lambda^-(a, b)$, значений λ_b^a и ρ_b^a (или пар $\lambda_b'^a, \lambda_b''^a, \rho_b'^a, \rho_b''^a$) и сравнения $\lambda_b'^a + \rho_b'^a$ с $\lambda^+(a+1, (b-1)/2)$ и $\lambda_b''^a$ с $\lambda^-(a+1, (b-1)/2)$, $0 \leq a \leq \log(2n)$, $1 \leq b \leq 2n/2^a$ пропорционально времени кодирования w' :

$$O(\log n M(2n \log n)) = O(\log n M(n \log n)).$$

Оценим время, затрачиваемое на определение букв x_i или пар x'_i, x''_i , $1 \leq i \leq 2n$. Для нахождения одной буквы или пары букв требуется не более двух сравнений слов, длина каждого из которых не превосходит $2^2 \lceil \log q_{\max} \rceil + 4 = 8 \log n + 10$ бит. Отсюда получаем, что общее число операций над битовыми словами, затрачиваемых на этом этапе, есть $O(n \log n)$.

Таким образом, находим, что время, требуемое для нахождения слова w' по его номеру $N(w')$, равно:

$$O(\log n M(n \log n)) + O(n \log n) = O(\log n M(n \log n)).$$

Можно заметить, что время денумерации слова w'' пропорционально времени его нумерации при учете того, что время умножения $M(n)$ двух слов длины n равно времени деления двух слов длины n , т.е. время денумерации w'' равно $O(\log n M(n))$.

Таким образом, общее время, требуемое для нахождения слова w по его номеру, равно:

$$O(M(n)) + O(\log nM(n \log n)) + O(\log nM(n)) = O(\log nM(n \log n)).$$

Скорость вычисления слова w по его номеру, т.е. время, необходимое для вычисления одной буквы слова w по его номеру, равна: $O(\log nM(n \log n)/n)$. Теорема доказана.

СПИСОК ЦИТИРОВАННОЙ ЛИТЕРАТУРЫ

- [1] Т. J. Lunch, “Sequence time coding for data compression”, *Proc. IEEE*, **54**:10 (1966), 1490–1491.
- [2] Ю. С. Медведева, Б. Я. Рябко, “Быстрый алгоритм нумерации слов с заданными ограничениями на длины серий единиц”, *Пробл. передачи информ.*, **46**:4 (2010), 130–139.
- [3] Т. М. Cover, “Enumerative Source Encoding”, *IEEE Trans. Information Theory*, **IT-19**:1 (1973), 73–77.
- [4] Б. Я. Рябко, “Быстрая нумерация комбинаторных объектов”, *Дискрет. матем.*, **10**:2 (1998), 101–119.
- [5] А. Е. Пентус, М. Р. Пентус, *Теория формальных языков*, Учебное пособие, Изд-во ЦПИ при мех.-мат. фак-те МГУ, М., 2004.
- [6] Э. Рейнгольд, Ю. Нивергельт, Н. Део, *Комбинаторные алгоритмы. Теория и практика*, Мир, М., 1980.
- [7] А. В. Ахо, М. С. Лам, Р. Сети, Д. Д. Ульман, *Компиляторы. Принципы, технологии и инструментарий*, Вильямс, М., 2008.
- [8] Р. Е. Кричевский, *Сжатие и поиск информации*, Радио и связь, М., 1989.
- [9] A. Schönhage, V. Strassen, “Schnelle Multiplikation grosser Zahlen”, *Computing*, **7**:3-4 (1971), 281–292.
- [10] M. Fürer, “Faster integer multiplication”, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, ACM, New York, 2007, 57–66.
- [11] А. Шень, *Программирование: теоремы и задачи*, МЦНМО, М., 2004.

Ю. С. Медведева

Институт вычислительных технологий СО РАН,
г. Новосибирск
E-mail: mjulja@gmail.com

Поступило

15.08.2013

Исправленный вариант

25.12.2013