Ссылка на статью:
// Ученые записки УлГУ. Сер. Математика и информационные технологии. УлГУ. Электрон. журн. 2018, № 2, с. 52-64.

Поступила: 20.10.2018
Окончательный вариант: 17.11.2018

УДК 004.421

Кодирование текста с использованием специальной матрицы

Смагин А.А.[1,*]

[*] smaginaa1@mail.ru
[1]УлГУ, Ульяновск, Россия

Рассматриваются методы кодирования текста, позволяющие уменьшить объем памяти для его хранения, приводятся оценки эффективности и примеры демонстрации методов.

Ключевые слова: *матрица, кодирование, декодирование, композиция, биграмма*.

Введение

Как указывалось ранее, координатное кодирование опирается на матричное преобразование входных данных, которые вначале приводятся к значениям координат (индексов) матрицы, а затем по последним определяется позиция в матрице и осуществляется выборка кода исходного текста.

В настоящем подразделе рассматриваются матрицы целых чисел, которые могут быть использованы для кодирования целых чисел как из диапазона значений, входящих в матрицу, так и чисел, взятых за пределами этого диапазона. В этом случае, кодирование выполняет функцию более компактного представления целых чисел и, в общем случае, больших массивов. Кроме того, учитывая что буквы текста также отождествляются с числами определенной длины, предполагаемый метод матричного кодирования позволит уменьшить затраты памяти на хранение больших текстов. Под смешанными числами здесь понимаются целые положительные и отрицательные числа.

Метод матричного кодирования с помощью целых чисел предполагает построение сетки матрицы A прямоугольной формы размером 10x10, т.е. координаты включают целые числа от 0 до 9 и могут указывать 100 целых чисел (

Таблица 1).

Таблица 1

		1	2	3	4	5	6	7	8	9
0		2	4	7	11	16	22	29	37	46
1		3	5	8	12	17	23	30	38	47
2		5	6	9	13	18	24	31	39	48
3		8	9	10	14	19	25	32	40	49
4	1	12	13	14	15	20	26	33	41	50
5	16	17	18	19	20	21	27	34	42	51
6	2	23	24	25	26	27	28	35	43	52
7	9	30	31	32	33	34	35	36	44	53
8	7	38	39	40	41	42	43	44	45	54
9	6	47	48	49	50	51	52	53	54	55

Сам процесс организации матрицы A основывается на свойстве симметрии и разбиении исходной матрицы на две треугольные: нижнетреугольную N_L и верхнетреугольную N_R матрицы (Рис. 1).

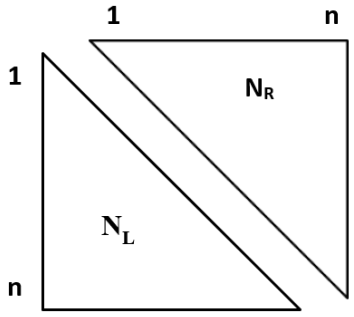

Рис. 1. Симметрия нижнетреугольной и верхнетреугольной матриц

Нижнетреугольная матрица N_L заполняется целыми числами

$$N_L = \{x_k\}, k = 1 \div 45, x \in Z,$$

причем диагональ D также состоит из целых положительных чисел

$$D = \{d_m\}, m = 1 \div 10, d \in Z.$$

Верхнетреугольная матрица N_R состоит из целых отрицательных чисел

$$N_R = \{-x_q\}, q = 1 \div 45, x \in Z.$$

Принцип заполнения матриц целыми числами заключается в использовании натурального ряда чисел и записи этих чисел по строкам до позиций диагональных чисел (нижнетреугольная матрица) с переходом на следующую нижнюю строку. [1]

Верхнетреугольная матрица N_R представляет собой зеркальное отражение матрицы N_L относительно диагонали и представлена теми же целыми числами, но с противоположным знаком (табл. 1).

Любой элемент матрицы А может быть определен как

$$x_{ij} = \begin{cases} j + 0{,}5 \cdot i(i+1) + 1, i \geq j, \\ -i - 0{,}5 \cdot j(j+1) - 1, i < j. \end{cases}$$

В силу симметрии матрицы относительно главной левой диагонали выполняется

$$a_{ij} = -a_{ji}$$

Такой прием организации матрицы А позволяет осуществлять кодирование целого числа от 0 до 99 в более компактном представлении с меньшим числом битов. При этом использование отрицательных чисел в верхнетреугольной матрице N_R при кодировании входных чисел сохраняет их положительные знаки. Сама симметрия матрицы позволяет уменьшить количество (ряд) чисел (сократить вдвое) с помощью которых осуществляется кодирование. [2]

Другими словами, исходный алфавит включает в себя множество чисел от 0 до 99 и его мощность $M_n = 100$, а алфавит матричного кодирования состоит из чисел от 1 до 55. Мощность алфавита матричного кодирования будет $M_m = 55$.

Разработанная матрица обладает следующими свойствами:

1. числа по строкам образуются как

$$x_{i,j} = x_{i,j-1} + 1, \quad (1)$$

2. числа по столбцам образуются как

$$x_{i,j} = x_{i-1,j} + i, \quad (2)$$

3. числа по диагонали

$$x_{i,j} = x_{i-1,j-1} + j + 1.$$

В теории матриц перечисленные выше числа именуются как генераторы строк, столбцов, диагоналей. С их помощью осуществляется генерация матриц в памяти компьютера.

1. Алгоритмы

1.1.Алгоритм матричного кодирования.

В основу положено представление любого целого положительного числа в виде последовательности пар соседних десятичных цифр от 00 до 99. Любое целое

многоразрядное число N разбивается на пары, причем каждая пара может нести значение от 00 до 99, причем любое число, представимое парой, может быть разложено как

$$N = i \cdot 10 + j,$$

где $i, j \in \{0,1,2,\dots,9\}$ и i используется в качестве номера (координаты) первой строки матрицы, а j – в качестве номера столбца (вторая координата) этой же матрицы. [3]

Пример. Пусть исходное число N=100002. Разобьем его на пары:

$$\underset{3}{10}\ \underset{2}{00}\ \underset{1}{02}$$

Каждая пара кодируется с помощью матрицы A отдельно. Представим число N как суперкомпозицию пар $N = N_1 + N_2 + N_3$, где N_i – пара цифр. Тогда координаты пар:

$$N_1 = 02 = 0 \cdot 10 + 2 \rightarrow i_1 = 0, j_1 = 2;$$

$$N_2 = 00 = 0 \cdot 10 + 0 \rightarrow i_2 = 0, j_2 = 0;$$

$$N_3 = 10 = 1 \cdot 10 + 0 \rightarrow i_3 = 1, j_3 = 0;$$

Следует отметить, что разбиение исходного числа происходит с его младшего разряда, справа налево. Если количество цифр в исходном числе нечетно, то в начало числа добавляется ноль. Симметрия матрицы A относительно диагонали дает возможность кодировать числа, в которых встраиваются комбинации несколько подряд стоящих нулей.

В рассматриваемом примере получим матричные координаты трех пар: N_1, N_2, N_3. Выборка по этим координатам дает их матричные коды

$$N_1 \rightarrow (-4);\ N_2 \rightarrow 1;\ N_3 \rightarrow 2$$

Используя суперкомпозицию, получаем матричный код числа N=100002 => 2.1.(-4).

1.2.Алгоритм декодирования

В основу декодирования положена идея использования интервальной матрицы. Если рассмотреть массив матричных чисел как множество строк целых чисел, возрастающих от некоторых начальных значений $x_i^{\text{н}}$ до своих максимальных значений $x_i^{\text{к}}$, то можно каждую i-ую строку заключить в интервал $\{x_i^{\text{н}}, x_i^{\text{к}}\}$. Здесь в интервал включены его границы $x_i^{\text{н}}$,$x_i^{\text{к}}$ (фактически, это отрезок). [4]

Правой границей интервала являются значения диагональных элементов матрицы, которые содержат все максимальные значения для строк нижнетреугольной матрицы N_L. Из-за симметрии матрицы A рассмотрение верхнетреугольной матрицы N_R не требуется, т.к. ее любое матричное число можно перевести в число матрицы N_L по формуле$a_{ij} = -a_{ji}$.

Из нижнетреугольной матрицы можно получить треугольную матрицу интервалов (рис. 2).

Преобразуем треугольную матрицу интервалов в трехстолбцовую прямоугольную, как наиболее удобную для последующего декодирования. Для этого сохраним столбец координат по строкам матрицы A и используем его в качестве первого столбца новой матрицы интервалов I.

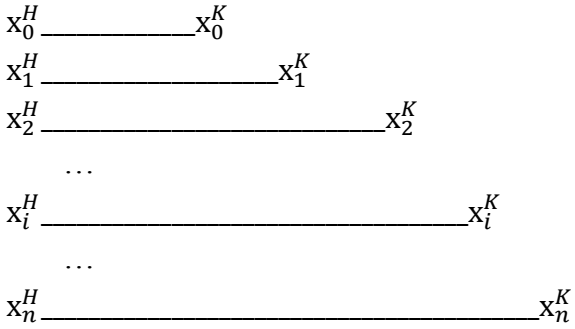

Рис. 2. Схема треугольной матрицы интервалов

Второй столбец в матрице I будет представлять левые границы интервалов $(x_i^{\text{н}})$ и в его качестве используем первый столбец матрицы A. Третий столбец будет представлен диагональю матрицы A и является правой $(x_i^{\text{к}})$ границей всех строчных интервалов. Полученная матрица интервалов представлена в таблице 2.

Процедура декодирования с помощью матрицы интервалов осуществляется следующим образом. Закодированное с помощью матрицы A число M построчно сравнивается со всеми интервалами

$$x_i^{\text{н}} \le M \le x_i^{\text{к}}, 0 \le i \le 9$$

Таблица 2

j/ i	**0**	**1**
0	1	1
1	2	3
2	4	6
3	7	10
4	11	15
5	16	21
6	22	28
7	29	36
8	37	45
9	46	55

И если оно попадает в один из них, то номер этой строки запоминается, т.е. определено значение первой координаты i. Для получения второй координаты j используется формула

$$j = M - \frac{i\cdot(i+1)}{2} - 1 \qquad (3)$$

Сформировав координаты i, j можно окончательно восстановить число N по формуле $N = i \cdot 10 + j$.

В случае отрицательного значения M необходимо взять число M по модулю и выполнить точно те же действия, что и в случае положительного M, однако для восстановления числа будем использоваться другая формула

$$N = -(j \cdot 10 + i). \quad (4)$$

Рассмотрим алгоритм декодирования на прежнем примере. В результате кодирования числа $N = 100002$ получен код $M = 2.1.(-4)$.

Выделим коды пар справа налево. $N_1^M = -4;\ N_2^M = 1;\ N_3^M = 2.$

Код пары: $N_1^M = (-4) \ \rightarrow$ строка с $i_1 = 2;$

$$j_1 = |N_1^M| - \frac{2\cdot(2+1)}{2} - 1 = 4 - 3 - 1 = 0.$$

для $i = 2, j = 0,$

$$N_1 = 0 \cdot 10 + 2 = 02.$$

Код пары: $N_2^M = 1 \ \rightarrow$ строка с $i_2 = 0;$

$$j_2 = 1 - \frac{0\cdot 1}{2} - 1 = 0.$$

для $i_2 = 0, j_2 = 0$

$$N_2 = 0 \cdot 10 + 0 = 00.$$

Код пары: $N_3^M = 2 \ \rightarrow$ строка $i = 1,$

$$j_3 = 2 - \frac{1\cdot 2}{2} - 1 = 0.$$

для $i_3 = 1, j_3 = 0$

$$N_3 = 1 \cdot 10 + 0 = 10.$$

Используя композицию, получаем исходное число

$$N = N_3 * N_2 * N_1 = 100002.$$

Важное достоинство матричного кодирования состоит в том, что требуется всего лишь одно обращение к матрице. Декодирование также упрощено, его время определяется $2i$ сравнениями кода с границами интервалами, расчётами значений j по (3.3) и для N по (3.4). Таким образом, предлагаемое матричное кодирование обладает такими важными свойствами, как экономичность записи целых чисел и высокой скоростью выполнения операций по их сжатию/восстановлению. [5]

Для кодирования текста структура предложенной матрицы сохранится, однако заголовки строк, столбцов матрицы необходимо изменить в соответствии со следующей идеей.

Известные таблицы частот биграмм для русского языка, которые имеют матричную форму и на пересечении столбца (имя первой буквы) и строки (имя второй буквы) записано значение частоты их встречаемости в тексте. Если переставить столбцы и строки этой таблицы частот таким образом, чтобы в левом верхнем углу матрицы кодирования, где представлены числа с малыми значениями, расположились биграммы с наибольшей частотой, то они будут кодироваться наиболее короткими кодами. Такая матрица будет иметь размер 32×32, а её элементами будут числа, получаемые по (3.1) и (3.2).

1.3.Алгоритм кодирования текста

Кодирование текста происходит следующим образом:

1. текст разбивается на биграммы;
2. в качестве номера строки кодирующей матрицы используется первая буква биграммы;
3. в качестве номера столбца используется вторая буква.

Выбранный элемент матрицы и представляет сжатый код биграммы.

Если таблицу частот формировать динамически, т. е. вычислять вероятности появления биграмм и затем их сортировать в порядке убывания, то появляется возможность дополнительного уменьшения общей длины исходного текста. При этом соответствующим образом и образуется сама матрица кодирования A. Достоинством динамического подхода является адаптация к кодируемому тексту — средневзвешенная длина кода становится короче. [6]

При оценке эффективности метода для кодирования текстов возникает естественный вопрос: является ли он оптимальным? Для того чтобы ответить на этот вопрос, найдём среднюю информацию на один элементарный символ, и сравним её с максимально возможной информацией, которая равна одной двоичной единице.

Для этого определим среднюю информацию, содержащуюся в одной биграмме передаваемого текста используя формулу энтропии (по К. Шеннону)

$$H(\sigma) = -\sum_{i=1}^{1024} P_i \cdot \log P_i$$

для методов на основе статического и динамического подходов, которые составили соответственно 7,63 и 7,58 бит/биграмму.

Среднее количество символов на одну биграмму для метода на основе статического подхода:

$n_{\text{ср}}$= 8,264

Среднее количество символов на одну биграмму для метода на основе динамического подхода:

$n_{\text{ср}}$= 8,109

Деля энтропию $H(\sigma)$ на $n_{\text{ср}}$, получаем информацию на одну биграмму — для метода на основе статического подхода:

$$I_{16} = \frac{H}{n_{\text{ср}}} = 0.923(\text{двоичных единиц}),$$

для метода на основе динамического подхода:

$$I_{16} = \frac{H}{n_{\text{ср}}} = 0.934(\text{двоичных единиц}).$$

Таким образом, полученные оценки весьма близки к своему верхнему пределу 1, а разработанный матричный код весьма близок к оптимальному.

Данные для анализа получены при кодировании среднестатистического текста объёмом 61,6 тысячи символов.

Время кодирования определяется как $T = t_{рч} + t_{в} + t_{вф}$, где $t_{рч}$– время разбиения числа на пары; $t_{в}$ – время выделения одноразрядных 10-х чисел; $t_{вф}$ – время вычисления по формуле.

Предложенный алгоритм отличает высокая скорость работы как при упаковке, так и при распаковке, достаточно скромные требования к памяти и простая программная реализация. Еще одним важным преимуществом данного метода кодирования является тот факт, что хранить матрицу совсем необязательно.

Хорошее сжатие обеспечивается на интервале [31750-32750]. Числа здесь имеют длину матричного кода в среднем на 1,89 бита меньше по сравнению с традиционным двоичным кодом.

2. Кодирование биграмм текста с использованием модифицированной матрицы B

Модифицированная матрица B отличается от матрицы A значениями матричных чисел в одноименных координатах. Элементы матрицы вычисляются по формуле:

$$a_{ij} = \begin{cases} j + 0.5i(i+1), i \geq j \\ -i - 0.5j(j+1), i < j \end{cases}$$

Матрица B имеет вид, представленный в таблице 3.

Таблица 3

i \ j	0	1	2	3	4	...
0	0	-1	-3	-6	-10	...
1	1	2	-4	-7	-11	...
2	3	4	5	-8	-12	...
3	6	7	8	9	-13	...
4	10	11	12	13	14	...
5	15	16	17	18	19	...
...	...	...	...	...	...	...

Матрица симметрична относительно левой главной диагонали и выполняется равенство $a_{ij} = –a_{ji}$.

Рассмотрим кодирующие свойства матрицы B на текстовом сообщении и будем, как и в предыдущем случае, осуществлять разбиение слов текста на пары символов и использовать их как координаты в матрице B.

Если сообщение нечётной длины, то к нему в конец добавляется пустой символ, то есть символ, про который заранее известно, что он не будет использован в сообщениях. Таким образом, сообщение приобретёт чётную длину и его можно будет закодировать данным методом. При декодировании пустой символ нужно будет убрать.

Например, закодируем слово «строка» (рис. 3).

Для повышения эффективности кодирования с помощью матрицы B используем код Хаффмана: –19+8+1, который дает суммарную длину кода 14 бит. Знаки чисел выполняют роль разделителей [7].

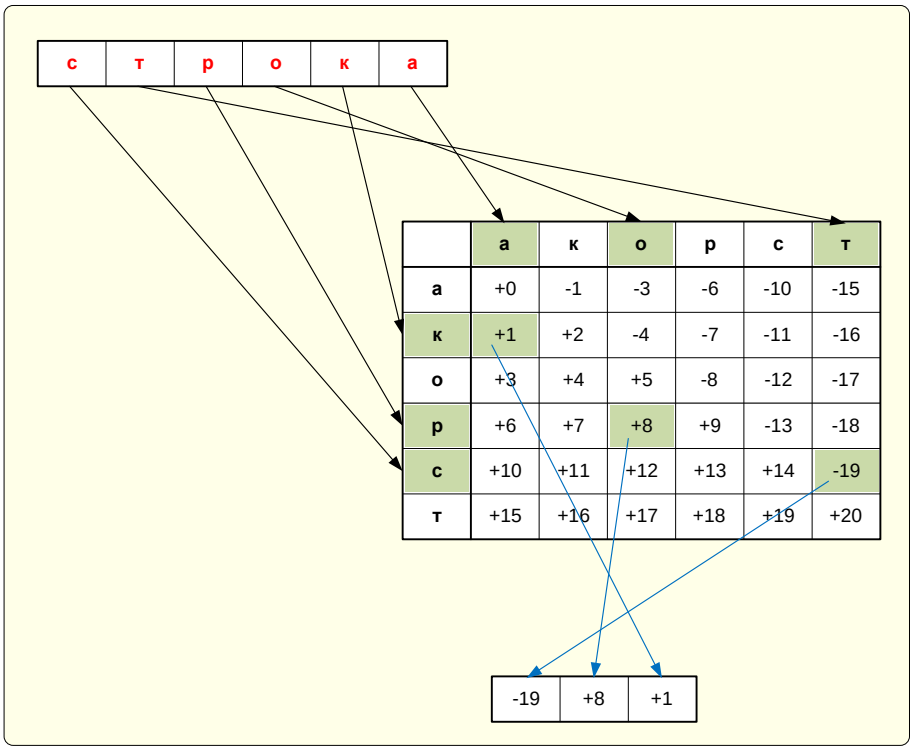

Рис. 3. Схема кодирования слова «строка»

Порядок символов в кодовой матрице берётся такой, в котором они расположены в алфавите. При этом сочетания последних букв в алфавите будут кодироваться наиболее длинными кодовыми комбинациями. Если такие сочетания встречаются в сообщении часто, то код получается слишком длинным. Поэтому возникает необходимость оптимизации алгоритма кодирования, используя частотный словарь. Здесь он будет выглядеть в виде частотной матрицы. Эту частотную матрицу нужно отсортировать по строкам и столбцам по убыванию частот, а полученный порядок символов использовать в частотной матрице. Проведем еще раз кодирование слова «строка».

Частотная матрица:

	а	к	о	р	с	т
А	0	0	0	0	0	0

к	0.33	0	0	0	0	0
о	0	0	0	0	0	0
р	0	0	0.33	0	0	0
с	0	0	0	0	0	0.33
т	0	0	0	0	0	0

Отсортированная частотная матрица:

	а	**о**	**т**	**к**	**р**	**с**
к	0.33	0	0	0	0	0
р	0	0.33	0	0	0	0
с	0	0	0.33	0	0	0
а	0	0	0	0	0	0
о	0	0	0	0	0	0
т	0	0	0	0	0	0

Кодовая матрица:

	а	**о**	**т**	**к**	**р**	**с**
к	+ 0	– 1	– 3	– 6	– 10	– 15
р	+ 1	+ 2	– 4	– 7	– 11	– 16
с	+ 3	+ 4	+ 5	– 8	– 12	– 17
а	+ 6	+ 7	+ 8	+ 9	– 13	– 18
о	+ 10	+ 11	+ 12	+ 13	+ 14	– 19
т	+ 15	+ 16	+ 17	+ 18	+ 19	+ 20

Тогда текстовое слово «строка» закодируется как:

«строка» → +5+2+0 → 01001100111. Длина кода – 11 бит.

Проведём сравнение двух методов кодирования на основе модифицированной матрицы.

Были представлены два метода кодирования. Второй метод является модификацией первого. Если проанализировать эти методы, то можно выявить области их эффективного применения.

Как видно из примеров, второй метод даёт меньший по длине код, чем первый. Но при этом нужно вместе с кодом передавать необходимую для декодирования информацию о кодовой матрице. При кодировании коротких сообщений размер этой информации может быть слишком велик по сравнению с самим кодом. При кодировании коротких сообщений и потоковых данных предпочтительнее использовать первый метод. Второй метод больше подходит для кодирования сообщений средней и большой длины.

Сравним с классическим методом кодирования Хаффмана.

По результатам исследования в большинстве сообщений метод кодирования Хаффмана даёт больший коэффициент сжатия, чем рассмотренные матричные методы. Преимущество этих методов в скорости.

Алгоритм кодирования Хаффмана состоит из двух основных шагов:

1. Заполнение частотного словаря (цикл с числом итераций равным длине сообщения).
2. Кодирование сообщения (цикл с числом итераций равным числу различных символов в сообщении).

Сообщение может содержать много символов. Рассмотренные матричные методы используют 12 символов: {1, 2, …, 9, 0, +, –}. Последующее кодирование в бинарный код методом Хаффмана может быть упрощено путём замены второго цикла на сопоставление символам заранее известных кодовых комбинаций.

Рассмотрим кодирование вторым методом слова «строка».

1. Матричное кодирование.
   «строка» → +5+2+0
2. Заполнение частотного словаря (в отсортированном виде).

	0.500
	0.166
	0.166
	0.166

3. Кодирование сообщения.

   Нужно заранее (всего один раз) вычислить кодовые комбинации для 12 символов. Здесь нужно лишь сопоставить символам найденные кодовые комбинации.

	0
	10
	110
	111

Получаем новый код слова «строка» → 010110111. Длина кода 9 бит.

К сжатому тексту с помощью кодирования пар можно применить один из статистических алгоритмов сжатия, например, арифметическое кодирование. При арифметическом кодировании текст представляется в виде двоичной дроби из полуинтервала [0; 1). Перед началом кодирования содержащий её интервал - [0: 1). Каждый новый символ сужает текущий интервал в соответствии со своей вероятностью, определяемой моделью. Более вероятные символы добавляют, таким образом, меньше битов к результату, чем менее вероятные. Арифметическое кодирование позволяет кодировать символы даже долями бита и, кроме того, осуществляет сжатие за один проход по источнику.

Заключение

Эффективность описанного выше метода с переходом на одиннадцатибитовую кодировку для текстовых файлов составила 3,5—3,6 бит/символ, что является довольно неплохим результатом, сравнимым со сжатием, которое даёт, например, LZW—модификация наиболее известного словарного алгоритма Лемпела-Зива. Однако такое сжатие осуществляется за 4 прохода, а расход памяти при использовании простых структур данных составляет около 500 Кб. К преимуществам же данного алгоритма можно отнести то, что распаковка сжатого текста осуществляется за один проход и требует очень мало памяти.

Список литературы

1. Андерсон Дж.А. *Дискретная математика и комбинаторика.* Пер. с англ..-М.: Издательский дом «Вильямс», 2003.
2. Берлекэмп Э. *Алгебраическая теория кодирования*. М.: Мир, 1971.
3. Грэхем Р., Кнут Д., Паташник О. *Конкретная математика. Основание информатики.* Пер. с англ. М.: Мир, 1998.
4. Смагин А.А. *Модели разбиений.* Ульяновск: УлГУ, 2013.
5. Смагин А.А.,Терентьева Ю.Б. Математическая модель счетчика построенная на основе кода Линча-Девисона // *Известия вузов. Приборостроение,* 2000, т.43, № 3, с.28-32.
6. Смагин А.А., Булаев А.А. Исследование весовых матриц целых десятичных чисел // *Ученые записки Ульяновского государственного университета. Сер. Математика и*

информационные технологии. Вып. 1(5) / Под ред. проф. А. А. Смагина. Ульяновск: УлГУ, 2013. С. 176-183.

7. Смагин А.А., Булаев А.А. Определение порядковых номеров чисел, образующих строки весовых треугольных матриц // *Ученые записки Ульяновского государственного университета. Сер. Математика и информационные технологии. Вып. 1(5)* / Под ред. проф. А. А. Смагина. Ульяновск: УлГУ, 2013. С. 169-176.