УДК 511.333; 511.337

БЫСТРОДЕЙСТВИЕ ИНДЕКСНЫХ АЛГОРИТМОВ ПРИ ПОИСКЕ ПРОСТЫХ ЧИСЕЛ ПО СРАВНЕНИЮ С РЕШЕТОМ АТКИНА

В. А. Минаев*, М. П. Сычев*, С. А. Никонов*, Д. В. Никеров**

**Московский государственный технический университет им. Н. Э. Баумана*
***Российский новый университет*

Поступила в редакцию 01.02.2016 г.

Аннотация. В статье описывается модификация индексного алгоритма поиска простых чисел и проведено сравнительное исследование его быстродействия по отношению к алгоритму с наименьшей асимптотической сложностью – решету Аткина.
Ключевые слова: простые числа, кольцевая факторизация, индексный алгоритм, решето Аткина.
Annotation. This article describes modification of the Primes index searching algorithm and comparative analysis of its performance with respect to the algorithm with minimal asymptotic complexity – sieve of Atkin.
Keywords: primes, wheel factorization, index algorithm, sieve of Atkin.

1. ВВЕДЕНИЕ

Для решения задачи нахождения полного множества простых чисел на определенном отрезке натурального ряда авторами обоснован и разработан индексный алгоритм их поиска [1, 2]. Метод базируется на результатах работы [3]. В настоящей статье описывается модификация индексного алгоритма и проведено сравнительное исследование его быстродействия по отношению к алгоритму с наименьшей асимптотической сложностью – решету Аткина [4].

Модификации алгоритма, описанные в статье, заключаются в использовании: битовых массивов; сочетания различных индексных алгоритмов для поиска простых чисел разных размеров; предпросева для малых простых чисел; блочного подхода; параллельных вычислений.

При описании метода использована терминология и обозначения, введённые в работах [1–3]. Так, под *порядком индексного алгоритма и порядком таблицы кольцевой факторизации* подразумевается количество первых простых чисел, использованных при получении приориала для формирования соответствующей кольцевой факторизации.

Переменная k – *индекс* числа на строке кольцевой факторизации с соответствующим номером, введена для формирования таблицы кольцевой факторизации.

Паттерн – повторяющаяся схема размещения составных чисел в таблице кольцевой факторизации, кратных одному и тому же числу.

Как и многие другие, индексный алгоритм для предварительного просеивания использует метод кольцевой факторизации [1]. Он позволяет не рассматривать значительную часть заведомо составных чисел, а из остальных формирует таблицу, в каждой колонке которой находятся как простые, так и составные числа. Например, при использовании второго порядка кольцевой факторизации отсеивается 66,66…% всех составных чисел, для четвертого порядка – больше 77% [5].

В работе [2] предложен подход, использующий свойство симметрии кольцевой факторизации (табл. 1), которое дает возможность более наглядно представлять суть индексного алгоритма, а также существенно экономить

вычислительные ресурсы при поиске простых. В табл. 1 использованы общепринятые обозначения: $p_1 = 2,\ p_2 = 3,\ldots,\ p_n$ – записанные в порядке возрастания простые числа; $p_n\#$ – приориал (произведение всех простых чисел, меньших либо равных p_n); $p_{n+1},\ldots,\ p_{n+r},\ldots,\ p_{n+s}$ – не участвовавшие в получении $p_n\#$ простые числа и их различные произведения, меньшие $p_n\#/2$ и записанные по возрастанию; s – количество простых чисел и их произведений на интервале $(p_n; p_n\#/2)$; $r = 1, 2, 3,\ldots, s$; j – номер столбца кольцевой факторизации.

В *центральном столбце* таблицы приведена последовательность индексов $0, 1, 2,\ldots, k_{\max}$, в остальных столбцах – отображения этой последовательности во множества вида

$$\{p_n\#\cdot k - p_{n+s}\},\ldots\ \{p_n\#\cdot k - p_{n+r}\},\ldots$$
$$\{p_n\#\cdot k - p_{n+1}\},\ \{p_n\#\cdot k - 1\},\ \{p_n\#\cdot k + 1\},$$
$$\{p_n\#\cdot k + p_{n+1}\},\ldots\ \{p_n\#\cdot k + p_{n+r}\},\ldots$$
$$\{p_n\#\cdot k + p_{n+s}\},$$

содержащие как простые, так и составные числа, а также единицу.

1. ПРАВИЛО ЗНАКОВ ПРОИЗВОЛЬНОГО ПОРЯДКА ПРИ ПОСТРОЕНИИ ТАБЛИЦЫ КОЛЬЦЕВОЙ ФАКТОРИЗАЦИИ

Для определения индексов простых чисел в каждом столбце таблицы кольцевой факторизации второго порядка в работе [1] предложены соотношения для составных чисел из столбцов таблицы кольцевой факторизации, называемые «правилом знаков». В работе [2] эти соотношения обобщены на произвольный порядок:

$$\begin{aligned}
c^{+p_{n+\eta}\cdot p_{n+r_2}}_{q^{-p_{n+\eta}}_{p_n\#\cdot i}} &= q^{-p_{n+\eta}}_{p_n\#\cdot i}\cdot q^{-p_{n+r2}}_{p_n\#\cdot m} = \\
&= (p_n\#\cdot i - p_{n+r_1})\cdot(p_n\#\cdot m - p_{n+r_2}) \\
c^{+p_{n+\eta}\cdot p_{n+r_2}}_{q^{+p_{n\eta}}_{p_n\#\cdot i}} &= q^{+p_{n+\eta}}_{p_n\#\cdot i}\cdot q^{+p_{n+r_2}}_{p_n\#\cdot m} = \\
&= (p_n\#\cdot i + p_{n+r_1})\cdot(p_n\#\cdot m + p_{n+r_2}) \\
c^{-p_{n+\eta}\cdot p_{n+r_2}}_{q^{-p_{n+\eta}}_{p_n\#\cdot i}} &= q^{-p_{n+\eta}}_{p_n\#\cdot i}\cdot q^{+p_{n+r_2}}_{p_n\#\cdot m} = \\
&= (p_n\#\cdot i - p_{n+r_1})\cdot(p_n\#\cdot m + p_{n+r_2}) \\
c^{-p_{n+\eta}\cdot p_{n+r_2}}_{q^{+p_{n+\eta}}_{p_n\#\cdot i}} &= q^{+p_{n+\eta}}_{p_n\#\cdot i}\cdot q^{-p_{n+r_2}}_{p_n\#\cdot m} = \\
&= (p_n\#\cdot i + p_{n+r_1})\cdot(p_n\#\cdot m - p_{n+r_2})
\end{aligned} \qquad (1)$$

Чтобы получить все формулы составных чисел, нужно записать выражения (1) со всеми сочетаниями значений r_1 и r_2 из ряда $1, 2, 3,\ldots s$; кроме этого, согласно правилу кольцевой факторизации нужно в ряде случаев вместо p_{n+r_1} и/или p_{n+r_2} ставить единицу. В работе [2] определен принцип, основанный на соотношениях (1), согласно которому составные числа расположены в приведенной таблице.

2. ИНДЕКСНЫЙ АЛГОРИТМ ПРОИЗВОЛЬНОГО ПОРЯДКА ОТСЕВА СОСТАВНЫХ ЧИСЕЛ ИЗ ТАБЛИЦЫ КОЛЬЦЕВОЙ ФАКТОРИЗАЦИИ

Отсев оставшихся после кольцевой факторизации составных чисел из множеств $\{q^{-p_{n+s}}_{p_n\#\cdot k}\},\ldots\{q^{-p_{n+r}}_{p_n\#\cdot k}\},\ldots\{q^{-p_{n+1}}_{p_n\#\cdot k}\},\ \{q^{-1}_{p_n\#\cdot k}\},\ \{q^{+1}_{p_n\#\cdot k}\},$ $\{q^{+p_{n+1}}_{p_n\#\cdot k}\},\ldots\{q^{+p_{n+r}}_{p_n\#\cdot k}\},\ldots\{q^{+p_{n+s}}_{p_n\#\cdot k}\}$, где $k = 0, 1, 2,\ldots k_{\max}$. Кратные числу $q \in \{Q\} = \{p_n\#\cdot i - p_{n+s}\}\cup \ldots\cup \{p_n\#\cdot i - p_{n+r}\}\cup \ldots\cup \{p_n\#\cdot i - p_{n+1}\}\cup \{p_n\#\cdot i - 1\}\cup \{p_n\#\cdot i + 1\}\cup \{p_n\#\cdot i + p_{n+1}\}\cup$

Таблица 1

Результаты симметричной кольцевой факторизации для $p_n\#$ в табличном виде

$q^{+p_{n+s}}_{p_n\#\cdot k}$	…	$q^{+p_{n+r}}_{p_n\#\cdot k}$	…	$q^{+p_{n+1}}_{p_n\#\cdot k}$	$q^{-1}_{p_n\#\cdot k}$	k	$q^{+1}_{p_n\#\cdot k}$	$q^{+p_{n+1}}_{p_n\#\cdot k}$	…	$q^{+p_{n+r}}_{p_n\#\cdot k}$	…	$q^{+p_{n+s}}_{p_n\#\cdot k}$
$-(s+1)$	…	$-(r+1)$	…	-2	-1	j	1	2	…	$(r+1)$	…	$(s+1)$
						0	1	p_{n+1}	…	p_{n+r}	…	p_{n+s}
$p_n\#\cdot 1 - p_{n+s}$	…	$p_n\#\cdot 1 - p_{n+r}$	…	$p_n\#\cdot 1 - p_{n+1}$	$p_n\#\cdot 1 - 1$	1	$p_n\#\cdot 1 + 1$	$p_n\#\cdot 1 + p_{n+1}$	…	$p_n\#\cdot 1 + p_{n+r}$	…	$p_n\#\cdot 1 + p_{n+s}$
…	…	…	…	…	…	…	…	…	…	…	…	…
$p_n\#\cdot k_{\max} - p_{n+s}$	…	$p_n\#\cdot k_{\max} - p_{n+r}$	…	$p_n\#\cdot k_{\max} - p_{n+1}$	$p_n\#\cdot k_{\max} - 1$	$k_{\max}$	$p_n\#\cdot k_{\max} + 1$	$p_n\#\cdot k_{\max} + p_{n+1}$	…	$p_n\#\cdot k_{\max} + p_{n+r}$	…	$p_n\#\cdot k_{\max} + p_{n+s}$

$\ldots\cup\ \{p_n\#\cdot i+p_{n+r}\}\cup\ldots\cup\ \{p_n\#\cdot i+p_{n+s}\}$, где $i=0,\ 1,\ 2,\ldots\ i_{\max}$ (для максимально возможного $i=i_{\max}$ должно быть выполнено условие $\left(q^{-p_{n+s}}_{p_n\#\cdot i_{\max}}\right)^2\le N_{\max}$), $r=1,\ 2,\ 3,\ldots\ s$, составные числа можно исключить, вычислив их индексы. Для этого каждому $q\in\{Q\}$ соответствует свой паттерн

$$t^j_q, q\in\{Q\}, \qquad (2)$$

то есть *схема размещения* кратных q чисел в строках с индексами $\left\lceil\frac{q}{2}\right\rceil,\ldots,\left\lfloor\frac{3q}{2}\right\rfloor$ табл. 1. Этот *паттерн повторяется* в строках с индексами $m\cdot q+\left\lceil\frac{q}{2}\right\rceil,\ldots\ m\cdot q+\left\lfloor\frac{3q}{2}\right\rfloor$, где $m=0,\ 1,\ 2,\ldots$, что показано в работе [2].

Рассмотрим в качестве примера отсеивание кратных $q=q^{+7}_{30\cdot 0}=7$ чисел при выполнении индексного алгоритма третьего порядка. Напомним, что третий порядок индексного алгоритма означает, что нужно взять 3 первых простых числа и перемножить их: $2\times3\times5=30$, после чего составить таблицу симметричной кольцевой факторизации, содержащую все простые числа и состоящую из членов прогрессий $\{30k-13\}=\{30(k-1)+17\}$, $\{30k-11\}=\{30(k-1)+19\}$, $\{30k-7\}=\{30(k-1)+23\}$, $\{30k-1\}=\{30(k-1)+29\}$, $\{30k+1\}$, $\{30k+7\}$, $\{30k+11\}$, $\{30k+13\}$. Для того чтобы отсеять все кратные 7 числа, нужно в диапазоне индексов [4; 10] определить паттерн (2) для числа 7:

$$t^j_q=\{3,2,0,-3,3,0,-2,-3\},$$
$$\text{для } j=-4,-3,-2,-1,1,2,3,4;\ q=7. \qquad (3)$$

Далее следует во множестве диапазонов индексов {[11; 17], [18; 24], …} исключить соответственно паттерну (3) кратные 7 числа. Описанный процесс показан в табл. 2, полученной из табл. 1 при $n=3$.

Обратим внимание на то, что паттерн t^j_q обладает в данном случае свойством центральной симметрии относительно q. Это свойство является общим для любого паттерна как следствие построения таблицы симметричной кольцевой факторизации. Более подробно примеры *использования паттернов* рассмотрены в работе [2].

Таким образом, если в заданном отрезке $[N_{\min}, N_{\max}]$ с помощью соответствующих паттернов (2) отсеять составные числа, кратные всем подходящим q, то в нем останутся только простые числа. Более того, достаточно рассмотреть только простые, поскольку составные будут высеивать те числа, которые уже отсеяны простыми до них. Отметим, что количество паттернов равно количеству элементов множества $\{Q\}$, которое зависит от значения $N_{\max}$. Следовательно, на отрезках равного размера с различным $N_{\max}$ количество паттернов будет больше у отрезка с большим $N_{\max}$. Перейдем к описанию некоторых программных модификаций индексного алгоритма.

3. ИСПОЛЬЗОВАНИЕ БИТОВЫХ МАССИВОВ ДЛЯ ЭКОНОМИИ ВЫЧИСЛИТЕЛЬНОЙ ПАМЯТИ

Для хранения объемной информации при поиске простых чисел нужно выбрать оптимальную структуру данных. Поскольку минимизация количества используемой памяти является приоритетным условием при реализации программы поиска простых, использование массива данных типа *bool* не является оптимальным вариантом. Это связано с тем, что переменная типа *bool* занимает в памяти относительно большое пространство – 1 байт. Поэтому возникает необходимость минимизации затрат памяти. Рассмотрим оригинальный способ построения битовых массивов.

Таблица кольцевой факторизации третьего порядка состоит из строк по 8 ячеек. Каждая из ячеек должна содержать информацию о том, простое число ей соответствует, или оно уже отмечено как составное. Поскольку в таблице ровно 8 ячеек, очень удобно одну строку кодировать одним байтом. То есть изначально таблице соответствует массив данных типа *unsigned char* или *uint8* заполненный байтами со значением 11111111 (1 обозначает, что число простое), а для того, чтобы отметить какое-либо число в строке как составное, нужно эту строку умножить на байт, состоящий из единиц и нуля на соответствующем месте.

Таблица 2

Пример адресации составных чисел при $q = q_{30\cdot0}^{+7} = 7$

$30k-13=30(k-1)+17$	$30k-11=30(k-1)+19$	$30k-7=30(k-1)+23$	$30k-1=30(k-1)+29$	k	$30k+1$	$30k+7$	$30k+11$	$30k+13$
				0	1	7	11	13
17	19	23	29	1	31	37	41	43
47	49	53	59	2	61	67	71	73
77	79	83	89	3	91	97	101	103
107	109	113	119	4	121	127	131	133
137	139	143	149	5	151	157	161	163
167	169	173	179	6	181	187	191	193
197	199	203	209	7	211	217	221	223
227	229	233	239	8	241	247	251	253
257	259	263	269	9	271	277	281	283
287	289	293	299	10	301	307	311	313
317	319	323	329	11	331	337	341	343
347	349	353	359	12	361	367	371	373
377	379	383	389	13	391	397	401	403
407	409	413	419	14	421	427	431	433
437	439	443	449	15	451	457	461	463
467	469	473	479	16	481	487	491	493
497	499	503	509	17	511	517	521	523
527	529	533	539	18	541	547	551	553
557	559	563	569	19	571	577	581	583
587	589	593	599	20	601	607	611	613
617	619	623	629	21	631	637	641	643
647	649	653	659	22	661	667	671	673
677	679	683	689	23	691	697	701	703
707	709	713	719	24	721	727	731	733
…	…	…	…	…	…	…	…	…

4. ИСПОЛЬЗОВАНИЕ СПЕЦИАЛИЗИРОВАННЫХ АЛГОРИТМОВ ДЛЯ МАЛЫХ, СРЕДНИХ И БОЛЬШИХ ПРОСТЫХ ЧИСЕЛ

Отсев составных чисел, кратных простым числам внутри текущего сегмента, осуществляется по-разному для чисел разного размера. Различаются следующие виды чисел. Большие – числа, которые встречаются в сегменте 1 раз или не встречаются вообще, если само число больше сегмента. Средние – встречаются несколько раз, то есть само число порядка размера сегмента. Малые числа – все остальные, то есть меньшие, чем размер сегмента.

Малые числа просеиваются с помощью индексного алгоритма третьего порядка, а средние и большие – с помощью индексного алгоритма 4 порядка, то есть размером 210. Это связано с тем, что кратные малым простым составные числа встречаются очень часто и, соответственно, их нужно очень часто отсеивать, а быстрее всего это делать с использованием описанного битового массива.

5. ДРУГИЕ ПОДХОДЫ К МОДИФИКАЦИИ ИНДЕКСНОГО АЛГОРИТМА

1. *Предварительный просев*: выполняется для заранее известных малых простых чисел.

2. *Блочный подход*: для значительного ускорения работы алгоритма происходит разбивка заданного отрезка натурального ряда на сегменты таким образом, чтобы при работе над каждым из них занимаемая память не превышала размеров кэша процессора первого уровня.

3. *Параллельные вычисления*: каждое ядро процессора можно загрузить как задачей просеивания своего множества сегментов, так и распределить всю задачу на множество процессоров в вычислительном кластере.

ОБСУЖДЕНИЕ РЕЗУЛЬТАТОВ

Изложенный алгоритм реализован на языке программирования C/C++ с использованием открытого исходного кода проекта http://primesieve.org , в котором авторы использовали оптимизированное решето Эратосфена. Далее алгоритм модифицирован до индексного и на этой основе проведено исследование его быстродействия в сравнении с оригинальной реализацией решета Аткина, размещенной по адресу http://cr.yp.to/primegen.html [6].

На рис. 1 представлены затраты времени на работу алгоритмов для отрезков натурального ряда от единицы до значений на оси абсцисс. Из рис. 1 видно, что для исследованной части натурального ряда индексный алгоритм работает значимо быстрее, чем решето Аткина.

ЗАКЛЮЧЕНИЕ

Индексный алгоритм в его текущей реализации представляет принципиально новый и перспективный подход к поиску простых чисел. Его быстродействие значительно выше по сравнению с решетом Аткина, характеризующимся среди алгоритмов поиска простых наименьшей асимптотической сложностью. Практическое применение модифицированного индексного алгоритма позволяет увеличить производительность алгоритмов вычисления простых чисел при переходе к длинной арифметике, а, следовательно, скорость факторизации составных чисел, что напрямую связано с обеспечением информационной безопасности систем и процессов [7–11], эффективностью алгоритмов шифрования/дешифрования информации.

СПИСОК ЛИТЕРАТУРЫ

1. *Минаев В. А., Васильев Н. П., Лукьянов В. В., Никонов С. А., Никеров Д. В.* Высокопроизводительный алгоритм генерации простых чисел в произвольном диапазоне с применением кольцевой факторизации // Спецтехника и связь. – М. : РосНОУ, 2013. – № 5. – С. 49–57.

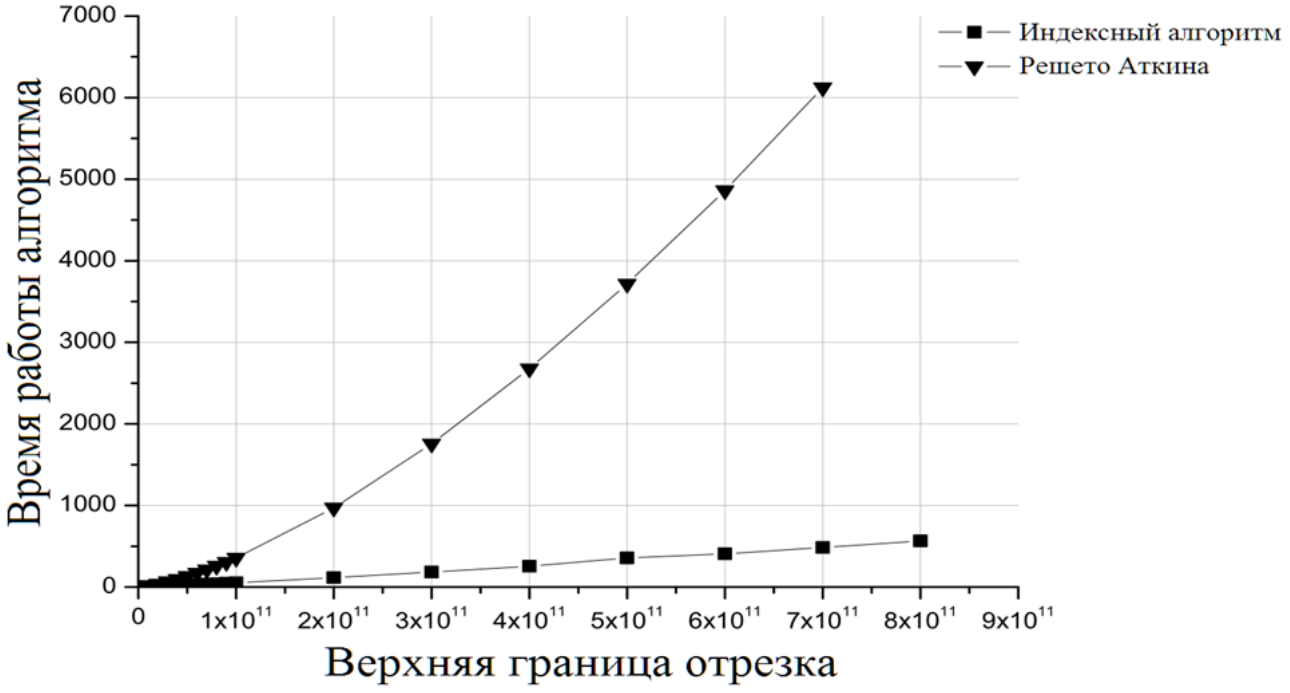

Рис. 1. Сравнение скоростей работы модифицированного индексного алгоритма и решета Аткина

2. *Минаев В. А., Никонов С. А., Никеров Д. В.* Симметричные формы индексных алгоритмов вычисления простых чисел // Спецтехника и связь – М. : РосНОУ, 2014. – № 5. – С. 40–48.

3. *Минаев В. А.* Простые числа: новый взгляд на закономерности формирования. – М. : Логос, 2011. – 80 с.

4. *Atkin A. O. L., Bernstein D. J.* Prime sieves using binary quadratic forms // Math. Comp. – 2003. – № 73 (246). – С. 1023–1030.

5. *Pritchard P.* Linear prime-number sieves: A family tree//Science of Computer Programming. 9 (1987), pp. 17–35.

6. *Bernstein D. J.* Primegen. – URL http://cr.yp.to/primegen.html. (Дата обращения: 19.03.2015).

7. *Минаев В. А., Саблин В. Н., Фисун А. П. Минаев В. А., Фисун А. П.* Теоретические основы информатики и информационная безопасность. – М. : Научно-техническое издательство «Радио и связь», 2000. – 468 с.

8. *Карпычев В. Ю., Минаев В. А.* Цена информационной безопасности // Системы безопасности. – 2003. – № 5. – С. 128–130.

9. *Минаев В. А., Фисун А. П., Касилов А. Г., Фисенко В. Е., Афанасьев В. В., Митяев В. В., Фисун Р. А., Джевага К. А., Кожухов С. А.* Развитие методологических основ информатики и информационной безопасности систем. Депонированная рукопись № 1165-В2004 07.07.2004.

10. *Минаев В. А., Скрыль С. В. и др.* Основы информационной безопасности. Под редакцией В. А. Минаева и С. В. Скрыль. – Воронеж : Издательство Воронежского института МВД России, 2001. – 464 с.

11. *Курушин В. Д., Минаев В. А.* Компьютерные преступления и информационная безопасность. – М. : Издательство «Новый юрист», 1998. – 256 с.

Минаев Владимир Александрович – д-р техн. наук, профессор, профессор МГТУ им. Н. Э. Баумана.
Тел.: (916) 294-92-90.
E-mail: m1va@yandex.ru

Сычев Михаил Павлович – д-р техн. наук, профессор, начальник Регионального учебно-научного центра «Безопасность» МГТУ им. Н. Э. Баумана.
Тел.: (495) 740-93-09.
E-mail: mpsichov@sm.bmstu.ru

Никонов Семен Андреевич – аспирант МГТУ им. Н. Э. Баумана.
Тел.: (916)723-25-38
E-mail: nikonov.simon@gmail.com

Никеров Дмитрий Викторович – преподаватель Российского нового университета.
Тел.: (916)666-76-02
E-mail: dnik@bk.ru

Minaev Vladimir Aleksandrovich – Doctor of Technical Sciences, Professor of the Bauman Moscow State Technical University.
Tel.: (916)294-92-90.
E-mail: m1va@yandex.ru

Sychev Mihail Pavlovich – Dr. of Technical Sciences, Professor, Head of the Regional educational and scientific center «Security» of the Bauman Moscow State Technical University.
Tel.: (495) 740-93-09
E-mail: mpsichov@sm.bmstu.ru

Nikonov Simon Andreevich – Graduate of the Bauman Moscow State Technical University.
Tel.: (916)723-25-38
E-mail: nikonov.simon@gmail.com

Nikerov Dmitriy Viktorovich – Teacher of the Russian New University.
Tel.: (916)666-76-02
E-mail: dnik@bk.ru