DOI: 10.18698/0236-3933-2015-6-82-90

УДК 511.333;511.337

# РЕАЛИЗАЦИЯ ИНДЕКСНЫХ АЛГОРИТМОВ ПОИСКА ПРОСТЫХ ЧИСЕЛ С ПОМОЩЬЮ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ

**В.А. Минаев**[1], **М.П. Сычев**[1], **С.А. Никонов**[1], **Д.В. Никеров**[2]

[1]МГТУ им. Н.Э. Баумана, Москва, Российская Федерация
e-mail: m1va@yandex.ru; runc@bmstu.ru; nikonov.simon@yandex.ru

[2]Российский новый университет, Москва, Российская Федерация
e-mail: dnik@bk.ru

*Для реализации новых подходов к обеспечению безопасности информационных систем предложен индексный алгоритм поиска простых чисел с использованием параллельных вычислений. Проведена оценка быстродействия алгоритма в зависимости от мощности используемой вычислительной системы.*

***Ключевые слова*:** простые числа, составные числа, кольцевая факторизация, индексный алгоритм, параллельные вычисления.

# IMPLEMENTATION OF INDEX PRIME NUMBERS SEARCH ALGORITHMS USING PARALLEL COMPUTING

**V.A. Minaev**[1], **M.P. Sychev**[1], **S.A. Nikonov**[1], **D.V. Nikerov**[2]

[1]Bauman Moscow State Technical University, Moscow, Russian Federation
e-mail: m1va@yandex.ru; runc@bmstu.ru; nikonov.simon@yandex.ru

[2]Russian New University, Moscow, Russian Federation
e-mail: dnik@bk.ru

*The article considers implementation of new approaches ensuring the information systems security. For this purpose, index prime number search algorithms using parallel computing is developed. The assessment of the algorithm performance depending on computer system power is made.*

***Keywords*:** prime numbers, composite numbers, wheel factorization, index algorithm, parallel computing.

Реализация целого ряда алгоритмов обеспечения информационной безопасности, шифрования/дешифрования информации связана с процедурой поиска и использования простых чисел. Для ускорения указанной процедуры авторами в целях нахождения полного множества простых чисел на определенном отрезке натурального ряда предложен метод, использующий индексы составных чисел, или индексный алгоритм [1–3].

Этот метод базируется на результатах работы [4]. Суть его состоит в том, что вычисляются индексы составных чисел, указывающие их место в натуральном ряду, а затем для нахождения самих простых чисел выполняется поиск составных чисел с использованием разработанных авторами индексных алгоритмов.

В настоящей работе описана суть индексного алгоритма, а также выполнена оценка его быстродействия при параллельном просеивании простых чисел в диапазоне до $10^{12}$ на разных количествах ядер вычислительной системы.

Модификации алгоритма, приведенные в настоящей работе, заключаются в использовании битовых массивов, различных алгоритмов для простых чисел разных размеров, предпросева для малых простых чисел, блочного подхода, параллельных вычислений.

При описании метода использованы терминология и обозначения, приведенные в работах [1–3].

Порядок индексного алгоритма, а также порядок таблицы кольцевой факторизации — это 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 во многих других, в индексном алгоритме поиска простых чисел используется известный метод кольцевой факторизации. Он позволяет не рассматривать значительную часть заведомо составных чисел, а из остальных чисел сформировать таблицу, в каждой колонке которой находятся как простые, так и составные числа.

Например, при использовании кольцевой факторизации второго порядка отсеивается 66,66. . . % всех составных чисел, четвертого порядка — более 77 %. Применение кольцевой факторизации подробно описано в работах [1–5]. В работе [2] для поиска простых чисел применен подход, использующий свойство симметрии кольцевой факторизации (табл. 1) и существенно экономящий вычислительные ресурсы при реализации индексных алгоритмов. В работе [3] выполнено сравнение быстродействия индексного алгоритма с решетом Аткина, как одним из самых эффективных методов поиска простых чисел. В результате обнаружено, что индексный алгоритм на исследованном отрезке натурального ряда работает в 12 раз быстрее. Отметим, что индексный алгоритм разработан авторами в РФ и в настоящее время публикаций о подобного рода подходах в зарубежной печати нет.

В табл. 1 использованы принятые ранее обозначения [1–4]: $p_1 = 2$, $p_2 = 3, \ldots, p_n$ — записанные по возрастанию 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 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 табл. 1 приведена последовательность индексов $0, 1, 2, \ldots, k_{\max}$, в остальных столбцах — отображения этой последовательности во множества вида $\{p_n\# \cdot k - p_{n+s}\}, \ldots, \{p_n\# \times \times k - p_{n+r}\}, \ldots, \{p_n\# \cdot k - p_{n+1}\}$, $\{p_n\# \cdot k - 1\}, \{p_n\# \cdot k + 1\}$, $\{p_n\# \times \times k + p_{n+1}\}, \ldots, \{p_n\# \cdot k + p_{n+r}\}, \ldots, \{p_n\# \cdot k + p_{n+s}\}$, содержащие как простые, так и составные числа, а также единицу.

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 | $p_n\#\cdot 1 + 1$ | $p_n\#\cdot 1 + p_{n+1}$ | … | $p_n\#\cdot 1 + p_{n+r}$ | … | $p_n\#\cdot 1 + p_{n+s}$ |
| … | … | … | … | … | … | … | … | … | … | … | … | … |
| $p_n\#\cdot k_{\max} - p_{n+s}$ | … | $p_n\#\cdot k_{\max} - p_{n+r}$ | … | $p_n\#\cdot k_{\max} - p_{n+1}$ | $p_n\#\cdot k_{\max} - 1$ | $k_{\max}$ | $p_n\#\cdot k_{\max} + 1$ | $p_n\#\cdot k_{\max} + p_{n+1}$ | … | $p_n\#\cdot k_{\max} + p_{n+r}$ | … | $p_n\#\cdot k_{\max} + p_{n+s}$ |

**Правило знаков произвольного порядка при построении таблицы кольцевой факторизации.** Для определения индексов простых чисел в каждом столбце таблицы кольцевой факторизации второго порядка в работе [4] предложены соотношения для составных чисел из столбцов таблицы кольцевой факторизации, называемые “правилом знаков”. В работе [2] эти соотношения обобщены на произвольный порядок:

$$
\begin{aligned}
c^{+p_{n+r_1}\cdot p_{n+r_2}}_{q^{-p_{n+r_1}}_{p_n\#\cdot i}} &= q^{-p_{n+r_1}}_{p_n\#\cdot i} \cdot q^{-p_{n+r_2}}_{p_n\#\cdot m} = (p_n\# \cdot i - p_{n+r_1}) \cdot (p_n\# \cdot m - p_{n+r_2});\\
c^{+p_{n+r_1}\cdot p_{n+r_2}}_{q^{+p_{n_1}}_{p_n\#\cdot i}} &= q^{+p_{n+r_1}}_{p_n\#\cdot i} \cdot q^{+p_{n+r_2}}_{p_n\#\cdot m} = (p_n\# \cdot i + p_{n+r_1}) \cdot (p_n\# \cdot m + p_{n+r_2});\\
c^{-p_{n+r_1}\cdot p_{n+r_2}}_{q^{-p_{n+r_1}}_{p_n\#\cdot i}} &= q^{-p_{n+r_1}}_{p_n\#\cdot i} \cdot q^{+p_{n+r_2}}_{p_n\#\cdot m} = (p_n\# \cdot i - p_{n+r_1}) \cdot (p_n\# \cdot m + p_{n+r_2});\\
c^{-p_{n+r_1}\cdot p_{n+r_2}}_{q^{+p_{n+r_1}}_{p_n\#\cdot i}} &= q^{+p_{n+r_1}}_{p_n\#\cdot i} \cdot q^{-p_{n+r_2}}_{p_n\#\cdot m} = (p_n\# \cdot i + p_{n+r_1}) \cdot (p_n\# \cdot m - p_{n+r_2}).
\end{aligned}
\tag{1}
$$

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 методу 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 и расположены в табл. 1.

**Индексный алгоритм произвольного порядка для отсева составных чисел из таблицы кольцевой факторизации.** Отсев оставшихся после кольцевой факторизации составных чисел из множеств $\{q_{p_n\#\cdot k}^{-p_{n+s}}\},\ldots,\{q_{p_n\#\cdot k}^{-p_{n+r}}\},\ldots,\{q_{p_n\#\cdot k}^{-p_{n+1}}\},\{q_{p_n\#\cdot k}^{-1}\},\{q_{p_n\#\cdot k}^{+1}\},\{q_{p_n\#\cdot k}^{+p_{n+1}}\},\ldots,\{q_{p_n\#\cdot k}^{+p_{n+r}}\},\ldots,\{q_{p_n\#\cdot k}^{+p_{n+s}}\}$, где $k=0,1,2,\ldots k_{\max}$. Кратные числу $q\in\{Q\}=\{p_n\#\cdot i-p_{n+s}\}\cup\ldots\cup\{p_n\#\cdot i-p_{n+r}\}\cup\ldots\cup\{p_n\#\cdot i-p_{n+1}\}\cup\{p_n\#\cdot i-1\}\cup\{p_n\#\cdot i+1\}\cup\{p_n\#\cdot i+p_{n+1}\}\cup\ldots\cup\{p_n\#\cdot i+p_{n+r}\}\cup\ldots\cup\{p_n\#\cdot i+p_{n+s}\}$, где $i=0,1,2,\ldots,i_{\max}$ (для максимально возможного $i=i_{\max}$ должно быть выполнено условие $\left(q_{p_n\#\cdot i_{\max}}^{-p_{n+s}}\right)^2\le N_{\max}$, $r=1,2,3,\ldots,s$, составные числа можно исключить, вычислив их индексы. Для этого каждому $q\in\{Q\}$ соответствует свой паттерн

$$t_q^j, q\in\{Q\}, \tag{2}$$

т.е. схема размещения чисел, кратных $q$, 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 чисел, кратных $q=q_{30\cdot0}^{+7}=7$, при выполнении индексного алгоритма третьего порядка. Напомним, что третий порядок индексного алгоритма означает, что нужно взять три 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 числа, кратные 7, нужен паттерн (2) для числа 7:

$$t_q^j=\{3,2,0,-3,3,0,-2,-3\},\ \text{для}\ j=-4,-3,-2,-1,1,2,3,4, q=7. \tag{3}$$

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

Обратим внимание на то, что паттерн $t_q^j$ 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 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

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 |
| … | … | … | … | … | … | … | … | … |

подходящим $p$, то в нем останутся только простые числа. Более того, достаточно рассмотреть только простые $p \in \{Q\}$, поскольку составные будут высеивать те числа, которые уже отсеяны простыми до них. Отметим, что число паттернов равно числу элементов множества $\{Q\}$, которое зависит от значения $N_{\max}$. Следовательно, на отрезках равного размера с различным $N_{\max}$ число паттернов будет больше у отрезка с большим $N_{\max}$. Перейдем к описанию некоторых программных модификаций индексного алгоритма.

**Модификация индексного алгоритма для параллельных вычислений.** Поскольку алгоритм построен таким образом, что может независимо обрабатывать заданный ему отрезок натурального ряда, то задачу поиска всех простых чисел на определенном большом отрезке можно разбить на подзадачи поиска на частях этого отрезка. Такой подход позволяет при программной реализации алгоритма использовать сразу две модификации: блочный подход и параллельные вычисления.

При использовании параллельных вычислений потоки могут явно взаимодействовать между собой через разделяемую память и/или через передачу сообщений. В нашем случае можно использовать оба типа взаимодействий, т.е. вести обмен данными между потоками на узле посредством общей памяти, а между узлами — с помощью передачи сообщений. Такой способ программирования более сложен, но

наиболее эффективен с точки зрения использования аппаратных ресурсов каждого узла многопроцессорной системы.

При написании программы авторы воспользовались открытым стандартом OpenMP для реализации взаимодействия потоков, работающих на одном узле. Реализация взаимодействий между узлами свелась к назначению каждому узлу своего отрезка натурального ряда и сбору данных, поступающих после обработки.

**Другие подходы к модификации индексного алгоритма.**

1. *Предпросев*: 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 одним из них занимаемая память не превышала размеров кэша процессора 1-го уровня.

3. *Битовые массивы*: в таблице кольцевой факторизации третьего порядка ровно 8 ячеек, и очень удобно одну строку кодировать одним байтом, 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.

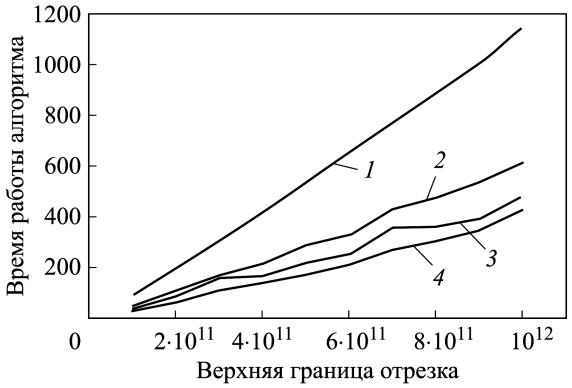
4. *Использование специализированных алгоритмов* для чисел разных порядков: больше размера сегмента (встречаются один раз или не встречаются вообще), порядка размера сегмента (встречаются несколько раз), и все остальные, т.е. меньшие, чем размер сегмента.

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 проведено его исследование на быстродействие на вычислительном кластере в МГТУ им. Н.Э. Баумана, располагающем 25 узлами Intel Xeon 5120 1.86 GHz с 4 ядрами каждый.

На рисунке приведены результаты о времени работы алгоритма для отрезков натурального ряда от единицы до значения на оси абсцисс при различных числах потоков на одном узле. На рисунке видно, что для исследованной части натурального ряда до $10^{12}$ при увеличении числа ядер время работы индексного алгоритма сокращается соответственно на 46 % для двух, на 58 % — для трех и на 63 % — для четырех ядер.

Результаты следующие: для вычисления всех простых чисел до $10^{15}$ потребовалось 504 ч работы в пересчете на один узел, до $2 \cdot 10^{15}$ — 1032 ч, а до $3 \cdot 10^{15}$ — 1584 ч. При вычислениях на всех доступных 25 узлах вычислительных ресурсов потребовалось соответственно 20, 42 и 64 ч реального времени.

**Выводы.** Индексный алгоритм в его текущей реализации представляет собой принципиально новый подход к поиску простых чисел.




**Сравнение времен работы индексного алгоритма поиска простых чисел в диапазоне натурального ряда до $10^{12}$ на различных количествах ядер**

Его быстродействие существенно улучшается при использовании параллельных вычислений. В настоящей статье дана количественная характеристика таких вычислений, а именно, на увеличение числа узлов алгоритм реагирует линейным увеличением производительности, на увеличение числа потоков в узле алгоритм реагирует нелинейно, но, тем не менее, также значительно. Это позволяет предположить, что при переходе к длинной арифметике и вычислениям на графических картах производительность алгоритмов вычисления простых чисел, а следовательно, скорость факторизации составных чисел возрастет, что даст возможность ставить и решать более сложные задачи обеспечения информационной безопасности [6–8], в частности, связанные с алгоритмами шифрования/дешифрования информации.

## ЛИТЕРАТУРА


1. *Высокопроизводительный* алгоритм генерации простых чисел в произвольном диапазоне с применением кольцевой факторизации / В.А. Минаев, Н.П. Васильев, В.В. Лукьянов, С.А. Никонов, Д.В. Никеров // Спецтехника и связь. 2013. № 5. С. 49–57.
2. *Минаев В.А.*, *Никонов С.А.*, *Никеров Д.В.* Симметричные формы индексных алгоритмов вычисления простых чисел // Спецтехника и связь. 2014. № 5. С. 40–48.
3. *Минаев В.А.*, *Никонов С.А.*, *Никеров Д.В.* Сравнение быстродействия модифицированного индексного алгоритма с решетом Аткина при поиске простых чисел // Спецтехника и связь. 2015. № 2. С. 38–41.
4. *Минаев В.А.* Простые числа: новый взгляд на закономерности формирования. М.: Логос, 2011. 80 с.
5. *Pritchard P.* Linear prime-number sieves: A family tree // Science of Computer Programming. 1987. No. 9. P. 17–35.
6. *Минаев В.А.*, *Саблин В.Н.*, *Фисун А.П.* Теоретические основы информатики и информационная безопасность. М.: Радио и связь, 2000. 468 с.
7. *Минаев В.А.*, *Скрыль С.В.* Основы информационной безопасности. Воронеж: Изд-во Воронежского института МВД РФ, 2001. 464 с.
8. *Курушин В.Д.*, *Минаев В.А.* Компьютерные преступления и информационная безопасность. М.: Новый юрист, 1998. 256 с.

## REFERENCES


[1] Minaev V.A., Vasil'ev N.P., Luk'yanov V.V., Nikonov S.A., Nikerov D.V. High speed algorithm of the primes generation in random range using wheel factorization. *Spetstekhnika i svyaz'* [Specialized machinery and communication], 2013, no. 5, pp. 49–57 (in Russ.).

[2] Minaev V.A., Nikonov S.A., Nikerov D.V. Symmetrical forms of index algorithms for prime numbers computing. *Spetstekhnika i svyaz'* [Specialized machinery and communication], 2014, no. 5, pp. 40–48 (in Russ.).

[3] Minaev V.A., Nikonov S.A., Nikerov D.V. Modified index algorithm with respect to sieve of Atkin primes search performance comparison. *Spetstekhnika i svyaz'* [Specialized machinery and communication], 2015, no. 2, pp. 38–41 (in Russ.).

[4] Minaev V.A. Prostye chisla: novyy vzglyad na zakonomernosti formirovaniya [Prime Numbers: A New Perspective on the Regularities of Formation]. Moscow, Logos Publ., 2011. 80 p.

[5] Pritchard P. Linear prime-number sieves: A family tree. *Science of Computer Programming*, 1987, no. 9, pp. 17–35.

[6] Minaev V.A., Sablin V.N., Fisun A.P. Teoreticheskie osnovy informatiki i informatsionnaya bezopasnost' [Theoretical Bases of Computer Science and Information Security]. Moscow, Radio i svyaz' Publ., 2000. 468 p.

[7] Minaev V.A., Skryl' S.V. Osnovy informatsionnoy bezopasnosti [Fundamentals of Information Security]. Voronezh, Voronezhskiy inst. MVD RF Publ., 2001. 464 p.

[8] Kurushin V.D., Minaev V.A. Komp'yuternye prestupleniya i informatsionnaya bezopasnost' [Computer Crimes and Information Security]. Moscow, Novyy yurist Publ., 1998. 256 p.



Статья поступила в редакцию 8.06.2015



Минаев Владимир Александрович — д-р техн. наук, профессор МГТУ им. Н.Э. Баумана.
МГТУ им. Н.Э. Баумана, Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5.

Minaev V.A. — D.Sc. (Eng.), Professor, Bauman Moscow State Technical University.
Bauman Moscow State Technical University, 2-ya Baumanskaya ul. 5, Moscow, 105005 Russian Federation.

Сычев Михаил Павлович — д-р техн. наук, профессор МГТУ им. Н.Э. Баумана.
МГТУ им. Н.Э. Баумана, Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5.

Sychev M.P. — D.Sc. (Eng.), Professor, Bauman Moscow State Technical University.
Bauman Moscow State Technical University, 2-ya Baumanskaya ul. 5, Moscow, 105005 Russian Federation.

Никонов Семен Андреевич — аспирант Российского нового университета.
Российский новый университет, Российская Федерация, 105005, Москва, ул. Радио, д. 22.

Nikonov S.A. — Ph.D. student, Russian New University.
Russian New University, ul. Radio 22, Moscow, 105005 Russian Federation.

Никеров Дмитрий Викторович — аспирант Российского нового университета.
Российский новый университет, Российская Федерация, 105005, Москва, ул. Радио, д. 22.

Nikerov D.V. — Ph.D. student, Russian New University.
Russian New University, ul. Radio 22, Moscow, 105005 Russian Federation.

**Просьба ссылаться на эту статью следующим образом:**

Минаев В.А., Сычев М.П., Никонов С.А., Никеров Д.В. Реализация индексных алгоритмов поиска простых чисел с помощью параллельных вычислений // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. 2015. № 6. С. 82–90.

**Please cite this article in English as:**

Minaev V.A., Sychev M.P., Nikonov S.A., Nikerov D.V. Implementation of index prime number search algorithms using parallel computing. *Vestn. Mosk. Gos. Tekh. Univ. im. N.E. Baumana, Priborostr.* [Herald of the Bauman Moscow State Tech. Univ., Instrum. Eng.], 2015, no. 6, pp. 82–90.