Запропоновано алгоритм рішення матриці на ходу. Досліджено степінь прискорення базового методу квадратичного решета на основі вирішення матриці на ходу. Показано, що модифікований алгоритм збільшив кількість вдалих розкладань. Тобто, було зменшено кількість випадків, коли базовий квадратичного решета (при стандартному інтервалі просіюванні та розміру факторної бази) не зміг сформувати матрицю для отримання рішення

Ключові слова: факторизація, квадратичне решето, B-гладкі, вирішення матриці на ходу, факторна база

Предложен метод решения матрицы на ходу. Исследована степень ускорения базового метода квадратичного решета на основе решения матрицы на ходу. Показано, что модифицированный алгоритм увеличил количество удачных разложений. То есть, уменьшено количество случаев, когда базовый алгоритм квадратичного решета (при стандартном интервале просеивания и размере факторной базы) не смог сформировать матрицу для получения решения

Ключевые слова: факторизация, квадратичное решето, B-гладкие, решение матрицы на ходу, факторная база

UDC 511:003.26.09
DOI: 10.15587/1729-4061.2018.127596

ACCELERATION ANALYSIS OF THE QUADRATIC SIEVE METHOD BASED ON THE ONLINE MATRIX SOLVING

S. Vynnychuk
Doctor of Technical Sciences, Senior Researcher*
E-mail: vynnychuk@i.ua
V. Misko
Postgraduate student*
E-mail: vitalii.misko@gmail.com
*Department of automation of design of power plants
Pukhov Institute for Modelling in Energy Engineering National Academy of Sciences of Ukraine
Henerala Naumova str., 15, Kyiv, Ukraine, 03164

1. Introduction

Integer factorization is one of the oldest problems in mathematics. However, major breakthroughs have occurred over the past 30 years, especially after the introduction of public-key cryptography, and in particular, the RSA cryptosystem. We can say that if factorization is solved effectively, the RSA cryptosystem will be extremely vulnerable. That is why the RSA Security company funded a factorization contest called "RSA challenge". It is interesting that latest introductions of factorization algorithms are closely related to the RSA challenge.

The authors carried out the studies [1, 2] of methods of cryptographic analysis of the RSA algorithm. However, [3] shows that known examples of compromise of the RSA algorithm work only for specific implementations, and usually are not more effective than the factorization problem. In 1994, factorization of the RSA-129 number was performed by means of the quadratic sieve algorithm (QS) [4].

This fact was perceived as a great surprise, since it was believed that the RSA-129 number is very difficult to factorize.

The quadratic sieve method (QS) is inferior to the general number field sieve method. However, for numbers up to 100 decimal digits, it is still the best [5, 6].

Modification of the quadratic sieve algorithm will allow reducing the running time of the algorithm and increasing the limit value of the factorized number for which the algorithm of the quadratic sieve method is the best.

Therefore, the study of new ways to reduce its computing complexity is relevant.

2. Literature review and problem statement

At the moment, there are several factorization methods and their modifications, basic of which have been considered in [7]. These methods are characterized by exponential and subexponential computing complexity.

In the quadratic sieve method, for the factorized N number, integers x are tried to be found such that

$$y(x)=x^2-N \tag{1}$$

can be decomposed into small prime factors – factor base elements, i. e., the numbers p=2 and other smallest primes p, for which N is the quadratic residue modulo p. Such values of y are called B-smooth [8].

The number L^a of the factor base elements in the basic version of the QS method is recommended [2, 4, 6] to be equal to

$$L^a=\left(e^{\sqrt{\ln(N)\ln\ln(N)}}\right)^{\sqrt{2}/4}=L(N)^{\sqrt{2}/4}=L^{\sqrt{2}/4}, \tag{2}$$

where the maximum prime number in the factor base B is called smoothness boundary.

The purpose of the algorithm is to find a set of B-smooth numbers, on the basis of which [9] it is possible to obtain the value of X such that

$$Y^2(X)\equiv X^2(\mathrm{mod}\,N), \tag{3}$$

where $Y(X)$ is the product of a number $y(x)$, determined as in (1).

The algorithm of the QS method works in two stages: the stage of forming a set of at least L^a+2 B-smooth numbers, on the basis of which one can obtain equal squares modulo N, and the data processing stage, where all collected information is placed in a matrix, the processing of which results in a solution [10–13].

At the first stage, the sieving interval is selected, the factor base is constructed and the sieving procedure is implemented. The most time-consuming part of the quadratic sieve algorithm is the sieving process when looking for B-smooth numbers based on the selection of x values in (1). In the general case (according to [14]), the size of the sieving interval can be obtained by the formula:

$$L^b = \left(e^{\sqrt{\ln(n)\ln\ln(n)}}\right)^{3\sqrt{2}/4} = L(n)^{3\sqrt{2}/4} = L^{3\sqrt{2}/4}. \quad (4)$$

The factor base size is one of the key parameters that determine the efficiency of the sieving algorithm. Too large factor base requires the search for a large number of B-smooth numbers, which increases the total execution time of the algorithm [1, 5, 13]. When the size is less than necessary, it will not be possible to find enough B-smooth numbers. The ratio (2) is the recommended number of the factor base elements, obtained on the basis of numerical experiments. Then the algorithm of the QS method searches B-smooth numbers in the quantity not less than $L^a + 2$. If enough B-smooth numbers are not found on the sieving interval, it is possible to increase both the size of the factor base and the sieving interval, which leads to a significant increase in the algorithm execution time.

For the quadratic sieve algorithm, a number of modifications related to the acceleration of the sieving process and solution of the matrix have been proposed.

To increase the number of possible B-smooth numbers, [12] proposes to memorize $y(x)=y_1(x)\cdot y_2(x)$ such that $y_1(x)$ is a smooth number and $B<y_2(x)<B^2$. In the presence of two such numbers y with the same y_2, their product becomes B-smooth. In [15], it is suggested to check whether $y_2(x)$ is an integer square. There is no need to look for a pair for such numbers as in the previous modification, they are called conditionally B-smooth and referred to a set of B-smooth.

In [5], the method of paralleling of the sieving process, known as MPQS (multiple polynomial quadratic sieve), has been proposed. It has been noted that the step of matrix solution cannot be parallelized, so steps have been taken to accelerate it [13].

In [16], it has been described that the number of units in the power matrix is much smaller than the number of zeros. For large numbers of 10^{100} or more, the ratio of the number of zeros to the number of units only grows. Most of the memory allocated for storing the matrix is used to store zeros. Therefore, instead of storing a two-dimensional matrix, it is proposed to store only units digit positions.

In all provided publications, it was considered that the stage of matrix solving requires a mandatory finding of the quantity of B-smooth numbers, not less than $L^a + 2$. In [17], a modification of the quadratic sieve algorithm has been proposed, in which, based on the current analysis of B-smooth numbers, the highest sequence number of the factor base element $p(\mathrm{i})$ is determined for each i-th B-smooth number, for which the exponent in the decomposition of B-smooth will be odd. If during obtaining a set of B-smooth it turns out that $L_{\max}$+2 B-smooth numbers are found, for which $p(\mathrm{i})\le L_{\max}$, then a matrix with the number of the factor base elements of $L_{\max}\le L^a$ can be formed. That is, both the size of the sieving interval and the size of the matrix can be reduced.

In the modified algorithm presented in [17], it is possible to achieve a decrease in the size of the sieving interval and the matrix only when in the set of $L_{\max}$+2 B-smooth numbers all odd powers of factors are assigned to the sequence numbers of the factor base elements, which do not exceed $L_{\max}\le L^a$. However, there are cases when the solution of the factorization problem is possible with a much smaller number of B-smooth numbers, where the factor base elements used in them can be placed randomly, not only among the smallest values. The option of $y(x)$ value, which is an integer square, is possible. Then the factorization problem is solved by the Fermat's method [1, 6, 7]. In other cases, methods of identification of such a subset of the factor base elements were not found in the scientific literature, as well as means of early identification of a set of B-smooth, for which the vectors, formed on the basis of odd exponents of the factor base elements, generate a linearly dependent subsystem. In this study, for the early identification of such a set of B-smooth numbers, it is proposed to use the online matrix diagonalization, when diagonalization continues with each occurrence of a new B-smooth and ends upon receipt of a zero vector.

3. The aim and objectives of the study

The studies were aimed at evaluating the efficiency of the modified quadratic sieve algorithm, which simultaneously implements the process of sieving and finding B-smooth numbers and the process of finding a zero vector in the diagonalization of the power vector matrix.

To achieve this aim, the following objectives were accomplished:

– to construct an algorithm for the online matrix solving in the quadratic sieve method;

– to analyze the influence of the online matrix solving on the speed and the result of factorization;

– to conduct a comparative estimation of complexity and time of implementation of the modified quadratic sieve method with the basic quadratic sieve and general number field sieve method.

4. Method of the online matrix solving of B-smooth numbers

In this study, we consider the problem of finding the sizes of the sieving interval and the factor base, where the factor base contains L^a elements, to be solved.

In the proposed algorithm that implements the online matrix solving, the additional vector $Vs[L^a+1]$ is used.

The search for a zero vector of the power matrix is presented below by the following steps:

1. Upon the occurrence of a new B-smooth number, the power vector *Vnew*, which corresponds to it, is introduced in the matrix.

2. We perform analysis of vector *Vnew*:

a. The position $k0$ of the first non-zero value of the vector *Vnew* and the position of the vector itself in the matrix kv are calculated.

b. If the zero value of the vector *Vnew* is absent, then the zero vector is found. Go to step 4.

3. The element of the vector *Vs* with the number that is equal to the first non-zero value of the added vector *Vnew–k*0 is checked.

a. If the element is empty, then the position of the added vector *Vnew–kv*: *Vs*[*k0*]=*kv* is introduced in this element. Go to step 1.

b. If this element is not empty, then a row that needs to be added to the vector *Vnew* is found using the value in this element. Go to step 2.

4. It is clarified whether the root value obtained is equal to *N*. If not, the factorization problem is solved, otherwise go to step 5.

5. Information about the *B*-smooth number with zero values of the transformed matrix elements is removed. Go to step 1.

5. Examples of application of the online matrix solving algorithm

Consider the efficiency of the proposed modification on the examples.

Example 1. We choose p=401 and q=103. These prime numbers form the number for factorization $p*q=N$=41303. We calculate the size of the factor base A=6 by the formula (2). Using the formula (4), we obtain the sieving interval M==203. The initial value $X0=xx=\sqrt{N}$ =204.

After sieving of $y(x)$ options through the factor base, we obtain *B*-smooth numbers. These numbers are shown in Table 1.

Table 1

Results of sieving of $y(x)$ options

Power vector of *B*-smooth numbers							*B*-smooth
Number sign	2	11	19	23	29	37	
1	1	1	0	0	2	0	–18502
1	**0**	**1**	**0**	**0**	**0**	**2**	**–15059**
1	**1**	**0**	**0**	**1**	**0**	**1**	**–1702**
0	1	0	2	0	0	0	722
0	0	0	0	2	1	0	15341
0	**1**	**1**	**0**	**1**	**0**	**1**	**18722**

For the basic quadratic sieve algorithm, these numbers are not enough to form a matrix and obtain a solution.

Table 2

Results of sieving of x options

X	Y	Factor base elements
865615	65147852	0 2 2 2 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
865533	–76806284	1 2 0 0 1 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 1
865481	–166819012	1 2 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 1

Note that the vectors in bold in Table 1 form a zero vector.

The method for the online matrix solving managed to find a zero vector with the current number of *B*-smooth numbers and factorize *N*.

Example 2. Let p=7624217, q=98269, N=749224180373, the factor base size: L^a=29, sieving interval: L^b=24052, initial value $X0=xx=\sqrt{N}$ =865578.

The factor base elements: 2, 7, 11, 19, 23, 31, 37, 41, 43, 61, 67, 71, 101, 127, 131, 157, 163, 167, 173, 179, 181, 191, 193, 211, 223, 227, 229, 241, 263.

The basic quadratic sieve method needed to check 5535 x options to get 30 *B*-smooth and factorize the number.

As can be seen from Table 2, the modified method has checked 194 x options, having obtained only 3 *B*-smooth.

This turned out to be enough to obtain a zero vector.

The modified algorithm obtained a solution *thirty* times faster than the basic algorithm.

Table 3 gives examples of the obtained acceleration factors of the modified algorithm with an approximate step of 20. Acceleration is calculated by the ratio of the number of *X* sieved by the basic method to the number of *X* sieved by the modified method.

The method of the online matrix solving in some cases allows forming a zero vector of already obtained vectors for B-smooth numbers even if the basic method failed to obtain enough B-smooth to form a matrix. So for one of the numerical experiments, 200 prime numbers in the interval from 23663 to 152065567 with a floating step were selected and ten thousand N for factorization were generated. The basic algorithm failed factorization in 686 cases of 10000. The modified algorithm reduced this figure to 503 cases. The time of factorization of all successful cases also decreased from 3941 seconds to 3301 seconds, which is 16 percent.

Factorization of one number of 10^{22} in size by the basic quadratic sieve method may take up to 2 minutes, so the studies for a large number of much larger numbers were not conducted because this requires substantial computing resources. For the efficiency analysis, control ranges for N close to $\log_{10} N^{12+(k+2)}$, where k=0, 1,.., 5 were chosen.

Table 3

Comparative analysis of the modified quadratic sieve method with the basic quadratic sieve method.

P	Q	Number of X sieved by the modified method	Number of X sieved by the basic method	L^a	Number of B-smooth obtained by the modified method	Number of B-smooth obtained by the basic method	Acceleration, times
152065567	21386557	7739	7739	54	59	59	1
24359	75511	62	655	18	7	19	10.5
170537	47087	37	769	20	4	21	20.8
44351	60343	33	1125	18	3	19	34.0
111121	41381	20	1377	19	3	20	68.8
853529	333367	226	19025	27	3	22	84.1
51749	32609	3	305	18	2	19	101.6

For each k, 10000 options of N were formed. p and q were chosen according to the formula

$$p = i*10^{\log_{10} N \pm j} + 200*f,$$

where

$$i = 1,2,...,9; \quad j = 0,...,5; \quad f = 0,1,...,200.$$

On the basis of the obtained results, their extrapolation on the numbers of the order up to 10^{130} was performed.

The efficiency analysis was conducted on several grounds:

1. The number of sieved X for the basic and modified methods.

The relative reduction of the total number of sieved X (in percentage) is given in Table 4.

Table 4

Percentage of acceleration of the modified quadratic sieve method relative to the basic quadratic sieve method, according to the factorized number size

lgN	%
14	13.06974138
16	12.20758206
18	10.23803938
20	9.047552192
22	8.337249169

Since the procedure of sieving the test values of x is the most time-consuming, the estimation of the number of sieved x is an important component that characterizes the effectiveness of the algorithm, and such estimation is accurate. At the same time, it is impossible to take into account the time for the matrix re-solving in case of wrong solutions. Therefore, the estimation by the number of sieved x is not complete.

2. The total execution time of the factorization task. The calculated percentage of reduction of the total execution time of the factorization program is given in Table 5.

Table 5

Percentage of acceleration of the modified quadratic sieve method relative to the basic quadratic sieve method, according to the factorized number size

lgN	%
14	15.42857143
16	12.47148289
18	11.89327278
20	10.65345846
22	8.647172602

It should be noted that the estimation of calculation time is not accurate. One and the same task can be performed for different (close) time associated with the running of the operating system task scheduler. Therefore, the data presented in Table 5, obtained on the basis of one calculation for each method are approximate. Estimation of the error of such data was not carried out, since each of the calculations required much time, which makes it impossible to conduct a statistically significant number of experiments.

According to the results from Table 4, 5, the functions were formed by the least squares method,

$$T = \frac{194.39}{\lg N} - 0.49 \qquad (5)$$

and

$$T = \frac{134.57}{\lg N} + 4.43$$

respectively.

Fig. 1 shows the graph of these functions.

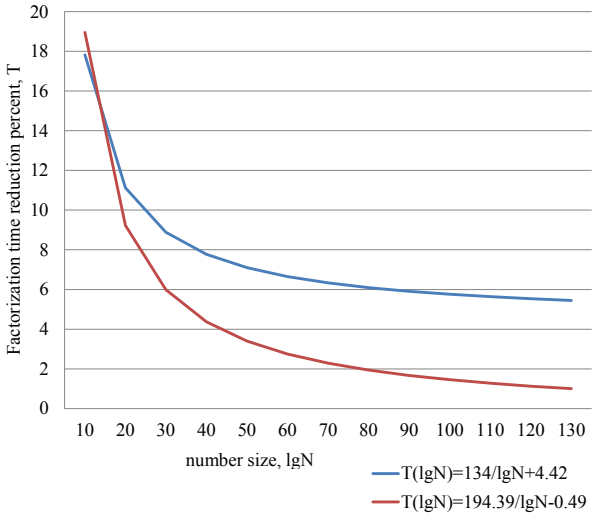

Fig. 1. Acceleration graph of the online matrix solving method

For further estimation, we will use the formula (5), since the comparison of execution time has errors, though takes into account the time to find p and q. Therefore, by the formula (5) we obtain that for numbers of 10^{100} in size, the modified quadratic sieve algorithm based on the online matrix solving has an acceleration of about 5.76 percent, and for numbers of 10^{130} in size – 5.45 percent.

6. Comparative estimation of complexity of the modified method of quadratic sieve with general number field sieve

To compare the relative efficiency of the QS and GNFS methods, we assume that

$$T^*_{QS}(N) = \exp\left((\ln N \cdot \ln\ln N)^{1/2}\right),$$

$$T^*_{GNFS}(N) = \exp\left((64/9)^{1/3}(\ln N)^{1/3}\cdot(\ln\ln N)^{2/3}\right),$$

$$T^*_{QS}(N) = K*T^*_{GNFS}(N),$$

where the coefficient K is determined from the condition that the QS method is better than GNFS for the numbers 10^{110}, but for $N>10^{129}$ GNFS will be better, and for some $10^{110}<N<10^{129}$ the relations are the same

$$\frac{T^*_{QS}(N)}{T^*_{GNFS}(N)} = \frac{T_{QS}(N)}{T_{GNFS}(N)}.$$

We calculate $T^*_{QS}(N)$, $T^*_{GNFS}(N)$ and

$$T_K(N) = T^*_{GNFS}(N)/T^*_{QS}(N),$$

the value of which will allow characterizing the coefficient K [18, 19].

Table 6

Comparative data for the QS and GNFS methods

lg$N$	$T^*_{QS}(N)$	$T^*_{GNFS}(N)$	$T_K(N)=T^*_{GNFS}(N)/T^*_{QS}(N)$
110	1.8209e+016	3.4122e+016	0.53365
111	2.2252e+016	3.9907e+016	0.5576
112	2.7172e+016	4.6635e+016	0.58266
113	3.3154e+016	5.4452e+016	0.60887
114	4.0422e+016	6.3527e+016	0.63629
115	4.9245e+016	7.4056e+016	0.66498
116	5.995e+016	8.6261e+016	0.69498
117	7.2927e+016	1.004e+017	0.72637
118	8.8648e+016	1.1677e+017	0.7592
119	1.0768e+017	1.357e+017	0.79355
120	1.3071e+017	1.5758e+017	0.82947
121	1.5854e+017	1.8285e+017	0.86706
122	1.9218e+017	2.1203e+017	0.90637
123	2.3278e+017	2.4568e+017	0.9475
124	2.8177e+017	2.8447e+017	0.99052
125	3.4085e+017	3.2915e+017	1.0355
126	4.1203e+017	3.8059e+017	1.0826
127	4.9776e+017	4.3977e+017	1.1319
128	6.0093e+017	5.0781e+017	1.1834
129	7.2501e+017	5.8598e+017	1.2373
130	8.7416e+017	6.7574e+017	1.2936

From Table 6, it follows that $T_K(N)$ grows with the increasing number of decimal digits N. However, the interesting fact is that with increasing the number of decimal digits N per unit, $T_K(N)$ increases 1.045 times for lg$N$=113 and reaches the value of 1.0459 for lg$N$=160, with a smooth monotonous growth. That is, regardless of the boundary value lg$N$, at which the QS and GNFS methods have the same computing complexity, any option of the improved QS method, for which the boundary value lg$N$ is increased by one, requires its computing complexity to be reduced at least 1.045 times.

For the performed calculations, $o(1)=1$. was chosen. It is possible to assert with full confidence that the dynamics for $o(1)\neq 1$ will be similar to that given above.

7. Discussion of the results of the study of the efficiency of the modified quadratic sieve method

The speed of the quadratic sieve method depends on such heuristic values as the size of the factor base and the sieving interval.

It is shown that for the selected 10,000 numbers of 10^{13} in size, the modified algorithm managed to reduce the number of failed factorizations from 686 cases to 503 relative to the basic quadratic sieve algorithm. This became possible due to the fact that in the modified algorithm there is no need to obtain L^a+2 B-smooth numbers prior to diagonalization of the matrix, as in the case of the basic method. A zero vector in a number of cases can be obtained much earlier, as illustrated in the examples given.

Among other important characteristics of this method, it should be noted that when used, the same operations as in the basic quadratic sieve method are performed, only their order is changed. That is, in the worst cases when the required number of B-smooth is L^a+2, the computing complexity of the modified method will be the same as in the basic one.

But it can be significantly reduced if the set of B-smooth numbers, for which the power matrix vectors form a linearly dependent system, are found quickly.

The peculiarities of the proposed modification include the fact that while simultaneously searching for B-smooth and diagonalizing the matrix, problems with the required amount of computer memory may arise, since it is known that when factorizing the RSA-129 number for solving the matrix, a supercomputer was used, which was not required to obtain B-smooth numbers.

In the analysis of the relationships between the computing complexity of the quadratic sieve and general number field sieve (GNFS) methods, it was found that an increase in the factorized number N by one decimal digit decreases the computing complexity of GNFS compared with QS 1.045 times (by 4.5 %) for numbers 10^{125}–10^{130} and this value varies quite slowly. Therefore, we can assume that any modifications to the quadratic sieve method, which, with the growth of N, reduce its computing complexity a number of times, asymptotically close to a constant, will not be competitive with GNFS with sufficiently large N.

The estimation of acceleration of the modified method for numbers up to 10^{130}, which is approximately 5.45 percent, shows that the proposed modified method allowed reducing the computing complexity of the basic QS method, but the value of the factorized numbers N, for which the method would be the best, increased only by a decimal digit.

Further improvements to the quadratic sieve method, which would provide a much more significant reduction in its computing complexity, should be related to approaches aimed at reducing the sieving interval and the size of the factor base, which in relative terms should be the greater, the higher N.

8. Conclusions

1. The algorithm for the online matrix solving, which accelerates the basic quadratic sieve method was developed. In some cases, 10, 100 and more times acceleration is possible. The average reduction of the computing complexity of the modified method for numbers up to 10^{130}, according to the estimates obtained, is 5.45 percent. This effect is associated with the possibility of obtaining a zero vector in some cases much earlier than L^a+2 B-smooth numbers are found, which is provided in the algorithm of the basic method and illustrated in the examples given.

2. On the basis of numerical experiments, it is shown that the online matrix solving allows the factorization of the number in some cases where the basic quadratic sieve algorithm (standard sieving interval and size of the factor base) failed to form a matrix for obtaining a solution. Namely, for the selected 10,000 numbers of 10^{13} in size, the modified algorithm managed to reduce the number of failed factorizations from 686 cases to 503 relative to the basic quadratic sieve algorithm.

References

1. Yan S. Y. Primality Testing and Integer Factorization in Public-Key Cryptography. Springer, 2009. 372 p. doi: 10.1007/978-0-387-77268-4
2. Analiz kanalov uyazvimosti sistemy RSA / Gorbenko I. D., Dolgov V. I., Potiy A. V., Fedorchenko V. N. // Bezopasnost' informacii. 1995. Issue 2. P. 22–26.
3. Brown D. Breaking RSA May Be As Difficult As Factoring // Cryptology ePrint Archive. 2005. URL: https://eprint.iacr.org/2005/380
4. Pomerance C. A Tale of Two Sieves // The Notices of the Amer. Math. Soc. 1996. Vol. 43, Issue 23. P. 1473–1485.
5. Landquist E. The Quadratic Sieve Factoring Algorithm // MATH 488: Cryptographic Algorithms. 2001. URL: http://www.cs.virginia.edu/crab/QFS_Simple.pdf
6. Ishmuhametov Sh. T. Metody faktorizacii natural'nyh chisel. Kazan': Kazan. un., 2011. 190 p.
7. Vasilenko O. N. Teoretiko – chislovye algoritmy v kriptografii. Moscow: MCNMO, 2003. 328 p.
8. Shnaer B. Prikladnaya kriptografiya. Moscow: Dialektika, 2003. 610 p.
9. Buhler J. P. Algorithmic Number Theory. Lattices, Number Fields, Curves and Cryptography: Mathematical Sciences Research Institute Publications. Cambridge University Press, 2008. 664 p.
10. Pomerance C. The quadratic sieve factoring algorithm // Lecture Notes in Computer Science. 1985. P. 169–182. doi: 10.1007/3-540-39757-4_17
11. Hoffstein J., Pipher J., Silverman J. The Quadratic Sieve Factoring Algorithm // An Introduction to Mathematical Cryptography. New York, 2001. 538 p.
12. Crandall R., Pomerance C. Prime Numbers. A Computational Perspective. New York, 2005. 597 p.
13. Pomerance C. Smooth numbers and the quadratic sieve // Algorithmic Number Theory. 2008. P. 69–81.
14. Pomerance C. Analysis and comparison of some integer factoring algorithms // In Computational Methods in Number Theory. Vol. 154. Amsterdam, 1982. P. 89–139.
15. Misko V. Pryskorennia metodu kvadratychnoho resheta na osnovi vykorystannia umovno B-hladkykh chysel // Systemni doslidzhennia ta informatsiyni tekhnolohiyi. 2018. Issue 1.
16. MSDN Archive. Factoring large numbers with quadratic sieve // MSDN. 2006. URL: https://blogs.msdn.microsoft.com/devdev/2006/06/19/factoring-large-numbers-with-quadratic-sieve
17. Vynnychuk S., Misko V. Pryskorennia metodu kvadratychnoho resheta na osnovi vykorystanni rozshyrenoi faktornoi bazy ta formuvannia dostatnoi kilkosti B-hladkykh chysel // Information technology and security. 2017.
18. Pomerance C. The number field sieve // Proceedings of Symposia in Applied Mathematics. 1994. P. 465–480. doi: 10.1090/psapm/048/1314884
19. Stevenhagen P. The number field sieve / J. P. Buhler, P. Stevenhagen (Eds.) // Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography. Cambridge, 2008.