Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36) *ISSN 2522-9818 (print)*
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36) *ISSN 2524-2296 (online)*

UDC 004.415.2:004.89:621.548 DOI: https://doi.org/10.30837/2522-9818.2026.2.109

Vladimir Pevnev, Oles Yudin

CLASSIFICATION AND EXPERIMENTAL STUDIES OF THE FACTORIZATION ALGORITHMS EFFICIENCY

The subject of the study is factorization algorithms, namely experimental testing of the efficiency of modern algorithms for integer factorization, identifying patterns between factorization algorithms and the size of the numbers being factorized in terms of the time required to perform the factorization operation. **The purpose of the article is to** analyze the performance of factorization algorithms using various composite numbers, in particular medium-sized numbers (~ 20 digits), which allows evaluating their efficiency and execution time. In the course of the research, the following **tasks** must be performed: classify modern factorization algorithms, experimentally verify the efficiency of modern Pollard factorization algorithms (p and $p-1$), the elliptic curve method, the Dixon algorithm, and the quadratic sieve. To achieve the goal, general scientific **methods** were used: analysis of the subject area and mathematical apparatus, set theory, numbers and fields, planning, and experimental research. **Results achieved.** Experimental studies have shown that Pollard's algorithms are effective for numbers with small divisors, but lose performance as the size of the numbers increases. The elliptic curve method has proven its usefulness in finding medium-sized divisors and has shown better scalability compared to classical stochastic methods. The Dixon algorithm, despite its relative simplicity of implementation, has demonstrated stochastic fluctuations in execution time, which limits its practical value in scenarios with strict time constraints. The most stable and predictable results were achieved for the quadratic sieve, which confirmed its suitability for factoring medium-sized numbers and provided the smallest spread of time values under conditions of multiple runs on identical data sets. **Conclusions.** The experiments fully confirm the theoretical expectations regarding the performance of the methods under study. The results indicate that simple algorithms (Pollard, elliptic curve method) are appropriate for preprocessing and detecting weak keys, while the quadratic sieve is the optimal choice for factoring medium-sized numbers. For large RSA modules, it is practical to use more complex algorithms, such as the general number field sieve. Further development of factorization algorithms involves parallelizing the factorization process and developing algorithms that use screening of impossible solutions.

Keywords: integer factorization; cryptography; Pollard method; elliptic curve method; quadratic sieve; Dixon's algorithm; algorithm efficiency.

Introduction

The qualitative leap in information technology that we have witnessed over the past decade has significantly changed approaches to the organization of technical and technological processes in industry. First of all, it is worth noting the use of artificial intelligence in the organization of production processes. This includes the creation of enterprises in which most of the processes are performed by robotic platforms, research and creation of new materials with specified properties, product control, ranging from food and medicines to spacecraft and nuclear power plants, etc.

Along with the introduction of new technologies, the number of accompanying documents is also growing, including new technologies, technological maps, reports on research work, accounting documentation, and information about individuals involved in various projects. All this data is stored both on hard media and in electronic form. The electronic form allows for the rapid transfer of documents between executors and real-time coordination, but at the same time, the confidentiality of the information being transferred and stored must be ensured. The most common way to ensure confidentiality is encryption. This method allows information to be transmitted over open communication channels, stored in cloud storage, and read only with a key.

The encryption process itself is based on the use of cryptographic systems. There are two large groups of these systems: symmetric and asymmetric. While the cryptographic strength of symmetric encryption systems is based on key secrecy, asymmetric systems rely on solving problems that are difficult to solve. In the first case, a single key is used for encryption and decryption, while in the second, two keys are used, one for encryption and the other for decryption. One such problem is the factorization of large numbers.

Factorization, or the decomposition of integers into prime factors, is one of the fundamental problems of

ISSN 2522-9818 (print) Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2026. No. 2 (36)

number theory and cryptography. It underlies the security of modern cryptographic protocols, in particular the RSA system. The security of RSA is based on the computational complexity of factoring large composite numbers, which are the product of two large prime numbers.

In modern research, factoring is considered a key element of cryptanalysis, as well as a training and research platform for testing the algorithmic efficiency of various calculation methods. There is a wide range of factorization algorithms. These include classical methods such as Fermat's method, Pollard's method, and Dixon's algorithm, as well as modern lattice methods and algorithms based on elliptic curves. Each of these algorithms has its own limitations and areas of application, which determine their practical suitability for factoring numbers of different sizes.

The aim of this study is to investigate algorithms in terms of their suitability for RSA systems. For each algorithm, the execution time is evaluated and a comparison of efficiency is performed on a set of composite numbers of different sizes.

Relevance of the study

Number factorization, i.e., the decomposition of integers into prime factors, is one of the oldest mathematical problems that has attracted the attention of many mathematicians over the centuries. Eratosthenes (276–194 BC) was the first known mathematician to develop a simple algorithm for finding prime factors. It is called Eratosthenes' sieve and enumerates all prime numbers smaller than a given integer N. Other prominent mathematicians who have made various contributions to factorization are Fermat (1607–1665) and Euler (1707–1783) [1].

The development of computer technology in the mid-20th century allowed mathematicians to create new and effective factorization algorithms. Significant improvements in the theory and practice of factorization were driven by the development of the well-known RSA public-key encryption algorithm.

The basic concept of asymmetric cryptographic algorithms is the use of two different keys: one for encrypting a message and the other for decrypting it [2, 3].

A key feature of such algorithms is the irreversibility of the encryption procedure, even with the public key available [4]. That is, knowing the public encryption key and the encrypted text, it is impossible to recover the original message. Decryption is only possible with the use of the corresponding private key. Thus, the public key can be publicly available – its disclosure does not pose a security threat, since it does not allow the encrypted information to be read. The security of the algorithm is ensured by the complexity of the inverse operation – determining the original prime numbers by their product [5]. As a result, factorization of numbers is the mathematical basis of modern asymmetric cryptography.

As information security becomes increasingly important, cryptographic mechanisms have become an integral component of every information system. The use of cryptographic mechanisms and protocols can solve many security problems (confidentiality, integrity control, authenticity, and non-repudiation of information). The development of cryptographic technologies has a direct impact on the economic, sociological, and political aspects of society as a whole. The development and use of factorization algorithms, analysis, and evaluation of their effectiveness allow new requirements to be developed for future versions of the RSA system, in particular for the size of the system module and the keys involved in the process of encrypting and decrypting information.

Thus, the widespread use of cryptography today increases the importance of developing cryptographic algorithms, and research into the features of applying number factorization algorithms for RSA is relevant.

Literature review

The operation of factorization algorithms is considered in the works of contemporary Ukrainian and foreign authors [6–8].

In [9], the dependence of the factorization of composite numbers on the time required to decompose this composite number using trial division, Fermat's algorithm, Pollard's p-method, Pollard's $(p-1)$-method, and Brent's method is compared. However, the authors do not take into account the conditions for using algorithms to solve cryptography problems.

In [10], a modification of Fermat's algorithm using Euler's function is considered. However, the author does not investigate the effectiveness of applying the proposed algorithm specifically for the implementation of encryption algorithms.

Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36) *ISSN 2522-9818 (print)*
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36) *ISSN 2524-2296 (online)*

In [11], the authors investigated three factorization algorithms: the divisor search algorithm, Fermat's algorithm, Pollard's p-method, and Shor's algorithm. The research is interesting in that emulating a quantum algorithm on a classical computer does not provide significant advantages. At the same time, the authors did not consider other algorithms used for the RSA system.

In [7, 12], the authors investigated the effect of the number of threads on the speed of execution of the division verification algorithm and their energy efficiency. However, the authors also did not take into account the peculiarities of the RSA system in their research.

Thus, modern research considers various factorization algorithms. However, in most cases, such studies do not analyze the possibility of using them for the RSA system.

The purpose of this study is to analyze the performance of factorization algorithms on a home computer using various composite numbers, including medium-sized numbers (~ 20 digits), to evaluate their efficiency and execution time. Particular attention is paid to algorithms that can be used to analyze the security of RSA systems with weak keys, as well as their educational value for demonstrating the principles of factorization.

Presentation of the main material

The factorization of natural numbers is a basic operation in arithmetic, which consists of representing a number $n \in \mathbb{N}$ as a product of prime factors. According to the fundamental theorem of arithmetic, every natural number $n > 1$ can be represented uniquely (up to the order of the factors) as a product of prime numbers.

The formalized formulation of the problem is as follows.

Let there be a given natural number

$$n \in \mathbb{N}, n > 1$$

The general problem of factorization is to find a set of prime numbers $\{p_1, p_2, ..., p_k\} \subset P$ and corresponding exponents $\{p_1, p_2, ..., p_k\} \subset P$ such that the following equality holds

$$n = p_1^{a_1} \cdot p_2^{a_2} \cdot ... \cdot p_k^{a_k}, \text{ where } p_i < p_{i+1}, \forall i$$

p_i are prime numbers, $a_i \geq 1$, $k \geq 1$ and the layout is unique with precision down to the rearrangement of factors [6, 7, 9].

The development of effective factorization algorithms allows for the improvement of both protection and cryptanalysis tools.

There are a large number of factorization algorithms.

1. Classical methods (based on quadratic congruences) [1, 9, 11].

These methods are based on finding numbers a, b, such that $a^2 \equiv b^2 \bmod n$.

a is not equal $\pm b$ by the module n.

Then the expression can be rewritten as

$$(a^2 - b^2) \equiv 0 \bmod n$$

or

$$(a-b)(a+b) \equiv 0 \bmod n.$$

It is possible to find a non-trivial divisor of the number n by calculating the greatest common divisor (GCD): $GCD(a-b, n)$ and $GCD(a+b, n)$. At least one of these GCDs will be a non-trivial divisor of n, which is the goal of factorization.

2. Stochastic and probabilistic methods [13].

Stochastic, or probabilistic, factorization methods do not guarantee finding a divisor within a specified time, but they have a high probability of success. Their effectiveness depends on the properties of the number itself or its unknown divisors. The basic idea is to use randomness to "stumble upon" a divisor. Unlike deterministic methods, they do not try all possible options, but make "lucky guesses".

3. Sieve methods [1, 6, 10, 11, 13].

Sieve methods are the most powerful known algorithms for factoring large integers. Their name comes from the main stage, which resembles the famous Sieve of Eratosthenes for finding prime numbers.

The general principle of these methods is to reduce the factorization problem to two basic steps:

– Sieving stage – efficient search for special "smooth" numbers (i.e., numbers that can be factored into small prime numbers).

– Linear algebra stage – combining the numbers found to construct a congruence of squares.

The ultimate goal, as in many other algebraic methods, is to find two different numbers a and b such that:

$$a^2 \equiv b^2 \bmod n$$

where n is a number that needs to be factored. If such a pair is found, the divisors can be easily calculated using $GCD(a-b, n)$ and $GCD(a+b, n)$.

ISSN 2522-9818 (print) Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2026. No. 2 (36)

The key idea that makes these methods so fast is that instead of checking millions of numbers one by one for "smoothness", the sieve allows you to do this en masse.

4. Algebraic methods [4, 6, 7, 13].

Algebraic factorization methods are based on using the properties of numerical structures, equations, and forms to find non-trivial divisors of composite numbers. Unlike classical or stochastic algorithms, which work mainly through brute force or random iterations, algebraic methods apply strict mathematical patterns that reduce the search space for factors.

The basic idea of such methods is to represent a given number *nnn* in the form of algebraic relations, such as equations or quadratic forms, and find solutions that lead to a non-degenerate divisor of the number.

5. Quantum algorithms [10].

Shore's algorithm is the most efficient theoretically known method for factorization on quantum computers. It performs decomposition in polynomial time, which threatens modern cryptography.

The most well-known algorithms of different classes are shown in Fig. 1.

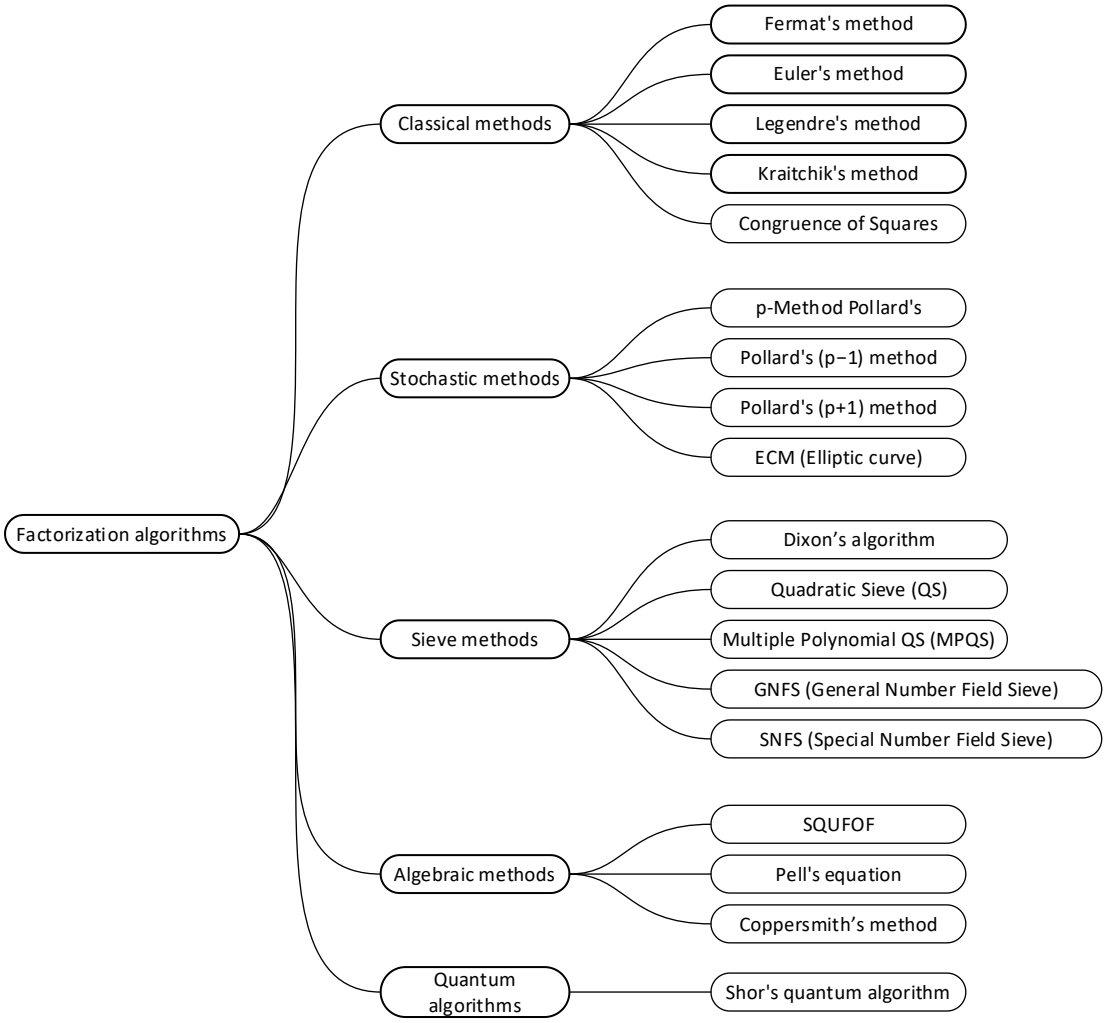

Fig. 1. Classification of factorization algorithms

Despite the significant variety of factorization algorithms, their use in RSA systems is somewhat limited.

Algorithms of a special structure, which include Fermat's method, Euler's method, Pollard's $(p-1)$ method, Williams' $(p+1)$ method, Bent's method, Pell's equation, SQUFOF, Legendre's method, Legendre-Gauss's method, and Kraitchik's method, are only effective in cases where the divisors have a special form (for example, $p-1$ or $p+1$ are easily factorized, or the numbers are close to each other). As a result, they are not applicable to RSA systems, since prime

Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36) *ISSN 2522-9818 (print)*
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36) *ISSN 2524-2296 (online)*

numbers for RSA are chosen randomly and specifically checked for the absence of weak structures.

Stochastic and probabilistic methods are suitable for finding "small" prime divisors. ECM, in particular, is the most effective method for finding small prime factors in large numbers. For RSA systems, such methods can only be effective in the case of incorrect key generation (for example, when one of the prime factors is significantly smaller than the other). In correctly generated RSA keys, prime numbers have the same length, so these methods are ineffective.

Lattice-type algorithms are among the most powerful classical algorithms. QS is effective for medium-sized numbers (up to ~ 110 digits). For very large numbers, such as RSA-1024, the only known practical method is GNFS. All successful attacks on RSA keys with a length of 768 bits and less are based on GNFS. Algebraic methods are used to construct a factor base at an auxiliary stage in the implementation of lattice methods (QS, GNFS).

The only known algorithm that makes RSA fundamentally vulnerable is Shor's quantum algorithm. However, its practical application requires a high-powered quantum computer, while emulation of this algorithm on a classical computer does not demonstrate high efficiency.

Therefore, lattice methods remain the most relevant methods for systems [14].

To compare factorization algorithms, this study chose methods that can find RSA module divisors in cases of poor key generation or demonstrate the principle of RSA key construction. All algorithms were implemented in Python, and Google Colab cloud platform was chosen as the execution environment. The experimental study was conducted on a sequence of numbers in the range from 4 to 1,000,000 with an interval of 10,000. This approach ensured equal execution conditions for each of the algorithms and enabled a correct comparison of their effectiveness.

Pollard's Rho method (Pollard's p-method) was chosen as the first algorithm for the study. It is usually used to find small divisors. If the RSA module was built with a "bad" prime number (for example, not a random number, but a number that has a certain structure), this method can quickly find it. Therefore, it is used to check RSA keys for weakness.

The idea of the algorithm is based on the birthday paradox and the idea of cycles in pseudo-random number sequences. First, a sequence of numbers is generated. To do this, an arbitrary initial number x_0 is selected. The sequence of numbers $x_1, x_2, x_3, \ldots$ is generated using a pseudorandom function $x_1, x_2, x_3, \ldots$, where N is the number to be decomposed, and c is a small constant (for example, $c = 1$) [8, 15].

Since the sequence is generated modulo N, it is cyclic. The algorithm searches for repetitions not in the sequence modulo N itself, but in the sequence modulo p, where p is a prime factor of N. Due to the birthday paradox, the probability that two numbers in the sequence will be equal modulo p is significantly higher than modulo N. For an efficient search for a match, Floyd's cycle detection algorithm, also known as the "tortoise and hare method", is used.

The formalized algorithm can be written as follows:

1. Select random initial values x_0 and $y_0 = x_0$.
2. Select constant $c \neq 0$.
3. Repeat:

– $x_i = \left(x_{i-1}^2 + c\right) \bmod N$

– $y_i = \left(y_{i-1}^2 + c\right) \bmod N$, and then again $y_i = \left(y_i^2 + c\right) \bmod N$ (i.e. $y_i = x_{2i}$)

– Calculate $d = GCD\left(|y_i - x_i|, N\right)$.

– If $d > 1$ and $d \neq N$, then d is a simple multiplier of the number N. The algorithm is complete.

– If $d = N$, this means that we have "jumped" over the multiplier. In this case, we need to start over with different initial values x_0 or c.

– If $d = 1$, continue iteration.

The complexity of this algorithm is approximately $O\left(N^{1/4}\right)$.

The results of applying the algorithm are shown in Fig. 2.

Analysis of the results shows that the execution time of Pollard's p-method is on the order of microseconds, varying from approximately 5.8×10^{-7} to 4.9×10^{-5} seconds. Most measurements are concentrated near the lower limit, indicating the stability and relatively high efficiency of the method for the selected input data. At the same time, there are isolated peak values that exceed the average level by more than an order of magnitude. This may be due to the specifics of the random component of the algorithm, which sometimes leads to longer searches for non-trivial divisors.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36)
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36)

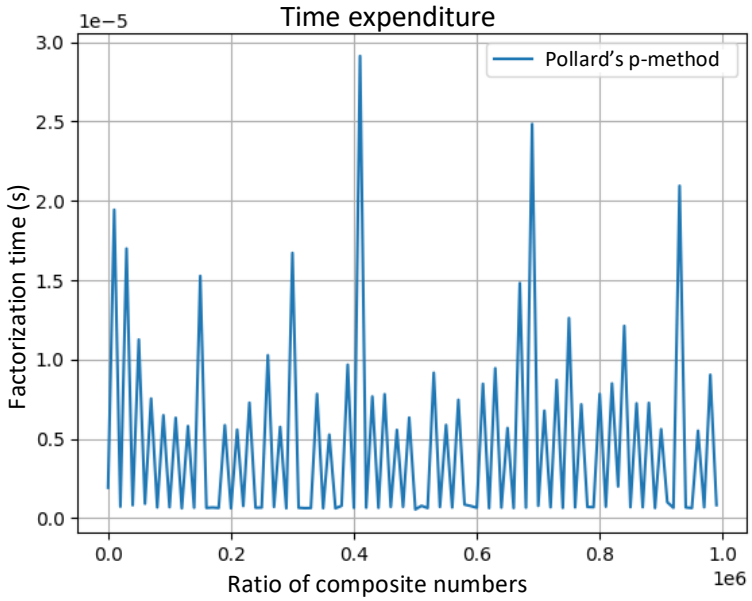

Fig. 2. Results of testing the Pollard's p -method algorithm

Thus, Pollard's p-method demonstrates high performance for factorization problems on small and medium-sized numbers, but is characterized by uneven execution time due to its probabilistic nature. This makes it suitable for practical application in conditions where execution time fluctuation is acceptable, but requires caution when estimating guaranteed computational costs.

The second algorithm considered is Pollard's $(p-1)$ method. This is a specialized factorization method that is effective for decomposing a number N if N has a prime factor p for which the number $p-1$ is smooth. A smooth number is a number whose prime factors are all less than a certain specified limit.

The algorithm is based on Fermat's little theorem, which states that if p is a prime number and a is an integer not divisible by p, then $a^{p-1} \equiv 1 (\mathrm{mod}\ p)$.[8, 16]

If $p-1$ is a smooth number, it can be decomposed into small prime factors. The algorithm uses this fact to find a multiple of $p-1$ (or $k(p-1)$).

The formalized algorithm can be written as follows:

1. Choose any initial number a, for example, $a = 2$.

2. Select the smoothness threshold B. This number determines how "smooth" $p-1$ should be.

3. Calculate K as the product of all powers of prime numbers less than or equal to B.

4. Calculate $a^K (\mathrm{mod}\ N)$ using the algorithm for rapid exponentiation by modulus.

5. Calculate $d = GCD(a^K - 1, N)$.

6. If $d > 1$ and $d \neq N$, then d is a simple factor of the number N. The algorithm has been successfully completed.

7. If $d = 1$, then the chosen boundary B is too small, and $p-1$ is not smooth with respect to B. It is necessary to increase B and repeat.

8. If $d = N$, then the chosen number a is not suitable. This can also happen if N has several factors p_i with smooth $p_i - 1$. You need to start again with another a.

The time complexity of this algorithm for factoring a sequence of numbers is shown in Fig. 3.

The results show that the execution time of Pollard's $(p-1)$ method varies widely: from values of the order of $10^{-6} - 10^{-5}$ seconds to individual peaks reaching 0.015–0.020 seconds. Most of the results are concentrated in the microsecond range, confirming the high efficiency of the algorithm for most of the numbers processed. At the same time, the presence of significant deviations from the average level indicates a substantial dependence of the execution time on the structure of the factorized number, in particular on the properties of its prime divisors and the order of elements in the multiplicative group modulo.

Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36) *ISSN 2522-9818 (print)*
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36) *ISSN 2524-2296 (online)*

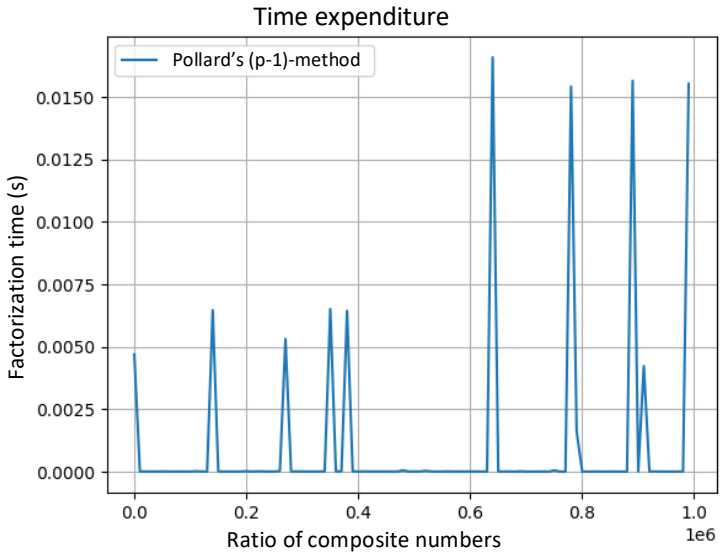

Fig. 3. Results of algorithm $(p-1)$ – Pollard's method testing

The Pollard $(p-1)$ method demonstrates asymmetric performance. In most cases, it works very quickly, but for some numbers it may require significantly more time.

The next algorithm under consideration is the Elliptic Curve Method (ECM). This algorithm is used to find non-trivial factors of integers. It is particularly effective for finding relatively small prime factors of large numbers, and its performance depends on the size of the smallest prime factor, not the size of the number itself, making it one of the most powerful modern methods for finding medium-sized factors.

ECM combines the ideas of Pollard's $(p-1)$ method and the group structure of points on an elliptic curve. Instead of performing calculations with numbers modulo N, ECM performs addition operations on points on an elliptic curve modulo N [11].

The algorithm selects a random elliptic curve E and a point P on it. An elliptic curve is generally described by the equation $y^2 = x^3 + ax + b \bmod N$.

The operation of "addition" can be defined on the points of an elliptic curve. This operation is associative, and the points form an Abelian group. Similar to the $(p-1)$ method in ECM, point P is "added" to itself K times, and $K \cdot P$ is calculated. The calculation of $K \cdot P$ is performed modulo N.

The main advantage of ECM is that the order of the group of points on an elliptic curve modulo p (where p is a prime factor of N) can be different for different curves.

The formalized algorithm can be written as follows:

1. Select the smoothness limit B_1.
2. Select a random elliptical curve E and a point P on it.
3. Calculate $Q = K \cdot P \pmod N$, where K – the product of all prime numbers less than or equal to B_1, in the appropriate degrees.
4. When calculating Q, each step checks whether an error occurs when dividing by $0 \pmod N$. If the GCD of the denominator and N is greater than 1, the factor is found.
5. If no multiplier is found, you can increase the limit B_1 or try another curve and point P.
6. The computational complexity of ECM depends on the size of the smallest prime factor p of the number N and is approximately equal to $O\left(e^{\sqrt{2\log p \log\log p}} \cdot \log^2 N\right)$. The complexity is sub-exponential with respect to p, not N. This means that the smaller the prime factor, the faster ECM will find it.

The probability of successfully finding a factor increases with the number of attempts (i.e., with the number of elliptic curves used).

The time required for factorization using this algorithm is shown in Fig. 4.

The results obtained demonstrate significant variability in the execution time of the ECM method. The values range from about 10^{-5} seconds to tens of seconds, with both relatively fast factorizations (e.g., in the millisecond range) and significant peak

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

loads recorded – in particular, 6.58 seconds and even 14.26 seconds. This uneven performance is explained by the mathematical nature of ECM. Its effectiveness depends significantly on the choice of random elliptic curves and the properties of the desired prime divisor. If the parameters are "favorable", the algorithm quickly finds the factor, but in unfavorable conditions, a large number of iterations are required, resulting in exponential peaks in execution time.

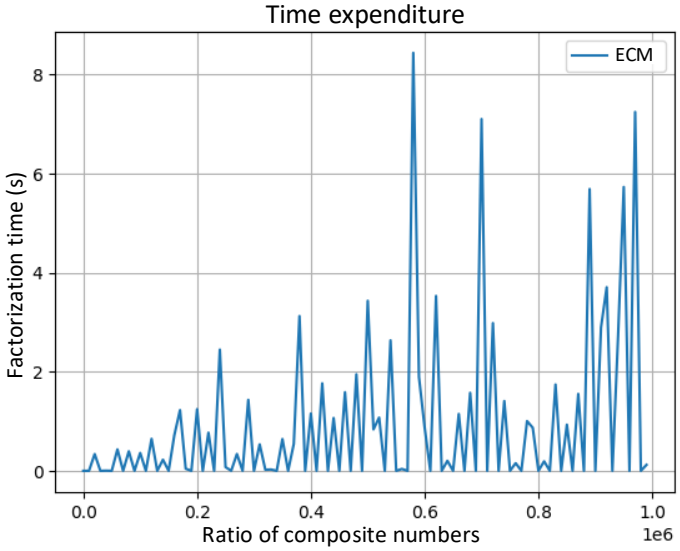

Fig. 4. Results of testing the ECM algorithm

Quadratic Sieve (QS) is one of the most efficient classical algorithms for factoring large integers. It belongs to the class of algorithms based on the idea of finding congruence of squares and is the best choice for factoring numbers up to 100–120 digits.

The algorithm is based on the idea of finding two numbers x and y such that $x \not\equiv$ and $x^2 \equiv y^2 \pmod N$. If such a pair is found so $x^2 - y^2 \equiv \pmod N$, then $x^2 - y^2 \equiv \pmod N$ is divisible by N. In this case, with high probability $GCD(x-y, N)$ or $GCD(x+y, N)$ will give a non-trivial factor of the number N [16].

The method for finding such a congruence consists in searching for a set of numbers z_i such that $z_i^2 \pmod N$ is a smooth number (i.e., it can be factorized from a small "factor base").

The computational complexity of the quadratic sieve is sub-exponential and is expressed by the formula $L_N[1/2, c] = e^{(c+o(1))\sqrt{\ln N \ln\ln N}}$, where $c = 1$ for the basic algorithm.

The time required for factorization using this algorithm is shown in Fig.5.

The results of experiments with a quadratic lattice (QS) demonstrate high stability of execution time. Most values are concentrated in the range of $5.0\times10^{-5} - 7.0\times10^{-5}$ seconds, which indicates relatively predictable computational behavior of the algorithm. Only a few experiments showed deviations, in particular peak values of about 1.0×10^{-4} seconds, but such cases are rare. Unlike stochastic methods, such as ECM, the quadratic sieve manifests itself as a more deterministic algorithm with uniform load. This is due to its structured construction based on the factor base and the method of linear algebra. It is thanks to this that QS is considered one of the most effective universal methods for factoring medium-sized numbers.

Thus, the experimental results confirm the theoretical estimates – the quadratic sieve ensures consistency and stability of execution time when calculating the factors of numbers in the studied range. This makes it practically suitable for factorization attacks on small and medium-sized RSA modules.

The last algorithm studied is the Dixon algorithm. It is a probabilistic method for factoring integers. The algorithm belongs to a class of algorithms based on finding congruences of squares and is the theoretical basis for more practical and faster algorithms, such as the Quadratic Sieve (QS).

Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36) *ISSN 2522-9818 (print)*
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36) *ISSN 2524-2296 (online)*

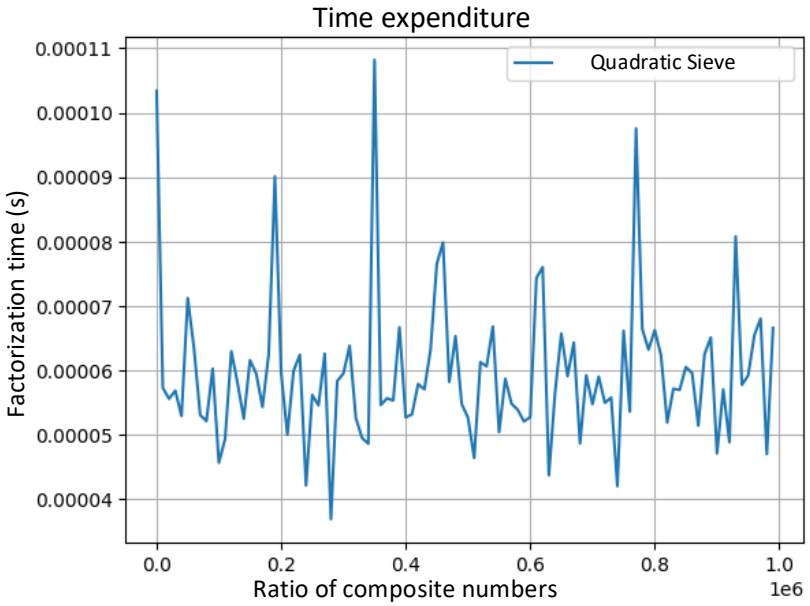

Fig. 5. Results of testing the Quadratic Sieve method

The basic principle of Dixon's algorithm is to find two numbers x and y such that $x \not\equiv \pm y \pmod N$ and $x^2 \equiv y^2 \pmod N$. If such a pair is found, then $x^2 - y^2 \equiv \pmod N$, which means is divisible by N [17, 18].

The method for finding such a pair is based on the system of smooth numbers.

The time required for factorization using this algorithm is shown in Fig.6.

For small composite numbers, the algorithm takes about 3.5×10^{-5} seconds to run. These values are quite close to QS, which shows the effectiveness of the method for small numbers.

With an increase in composite numbers, there is a gradual increase in the duration of calculations, in some cases up to $0.0015 - 0.0023$ s. This is because Dixon requires more iterations to find dependencies in the factor base as the size of the number increases. There is a strong variability of values between neighboring numbers (values can jump from 0.0006 to 0.0017 s). This indicates the stochasticity of the algorithm, i.e., the efficiency depends on the random selection of squares and the solution of linear systems.

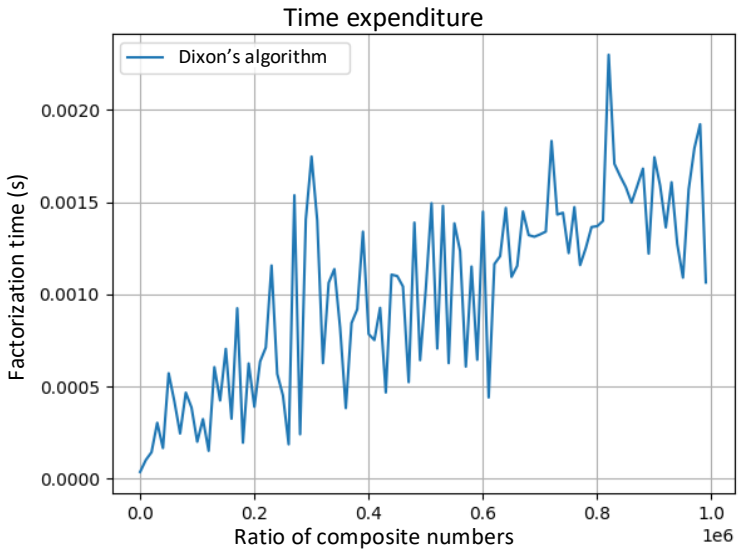

Fig. 6. Results of testing the Dixon algorithm

ISSN 2522-9818 (print) Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2026. No. 2 (36)

The upper peaks during the experiment are isolated, but significantly exceed the average values, which confirms low stability compared to QS. The Dixon algorithm demonstrates average efficiency for the numbers studied. It is faster than Pollard's methods $(p, p-1)$ on large numbers, but inferior to the quadratic sieve in terms of stability and predictability of running time. This confirms its status as a transitional method between simpler factorization algorithms and more advanced ones (QS, NFS).

Conclusions

The experimental study allows us to draw the following conclusions regarding the effectiveness of factorization methods.

Pollard's algorithms showed some variability in their performance, which makes them suitable for practical application in conditions where execution time fluctuations are acceptable, but requires caution when estimating guaranteed computational costs. As a result, it is advisable to use these methods in combination with other factorization algorithms, which allows compensating for their uneven performance.

The elliptic curve method (ECM) showed higher efficiency compared to Pollard's algorithms when searching for divisors of average size. The results of the experiments confirm the theoretical position on the advisability of its use in the initial stages of factorization of large numbers.

The Dixon algorithm demonstrated stochastic behavior and significant fluctuations in execution time, which corresponds to its probabilistic nature. The results confirmed its limited practical value compared to more modern methods.

The quadratic sieve (QS) showed the best results among all implemented algorithms, ensuring stability and relatively predictable execution time. This confirmed the theoretical complexity estimates and proved the feasibility of using this method for factoring medium-sized numbers.

Thus, the experiments fully confirm the theoretical expectations regarding the performance of the methods studied. The results obtained indicate that simple algorithms (Pollard, ECM) are appropriate for preprocessing and detecting weak keys, while the quadratic sieve is the optimal choice for factoring medium-sized numbers. For large RSA modules, it is practical to use more complex algorithms, such as the general number field sieve (GNFS).

It should be noted that when solving the problem of compromising the RSA system, factorization algorithms have some differences compared to the mathematical problem of factorization. This difference is due to the number of factors used to construct the number to be factored. Based on the research conducted, the following directions for the further development of factorization algorithms can be formulated. The first is the parallelization of the factorization process, and the second is the screening of impossible solutions.

Conflict of interest

The author declares that there is no conflict of interest, particularly of a financial, personal, authorial, or any other nature, that could influence the research or the results published in this article.

Funding

The research was conducted without financial support.

Data availability

The manuscript has no associated data.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies to write this paper.

References

1. Mahato, P. and Shah, A. (2023), "A review of prime numbers, squaring prime pattern, different types of primes and prime factorization analysis", *International Journal for Research in Applied Science and Engineering Technology*, 11, pp. 2036–2043. DOI: https://doi.org/10.22214/ijraset.2023.54904
2. Klesov, O.I. (2016), *Elementary Number Theory and Elements of Cryptography*, TViMS, Kyiv, 412 p., available at: https://ela.kpi.ua/handle/123456789/30046 (accessed 11 September 2025).

Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36) *ISSN 2522-9818 (print)*
Innovative technologies and scientific solutions for industries. 2026. No. 2 (36) *ISSN 2524-2296 (online)*

3. Pieprzyk, J. (2019), "Integer Factorization – Cryptology Meets Number Theory", *Scientific Journal of Gdynia Maritime University*, 1(109), pp. 7–20. DOI: https://doi.org/10.26408/109.01
4. Pevnev, V., Yudin, O., Sedlaček, P. and Kuchuk, N. (2024), "Method of testing large numbers for primality", *Advanced Information Systems*, 8(2), pp. 99–106. DOI: https://doi.org/10.20998/2522-9052.2024.2.11
5. Fedorchenko, V., Yeroshenko, O., Shmatko, O., Kolomiitsev, O. and Omarov, M. (2024), "Methods of information systems protection", *Advanced Information Systems*, 8(4), pp. 82–92. DOI: https://doi.org/10.20998/2522-9052.2024.4.11
6. Kudinov, M. and Muntean, P. (2025), "Modern Number Factorization Algorithms: Efficiency Analysis and Applications", *Collection of Abstracts of Scientific Reports by Higher Education Applicants of Berdiansk State Pedagogical University*, 208 p. DOI: https://doi.org/10.5281/zenodo.15521215
7. Prots'ko, I.O. and Gryschuk, O.V. (2019), "Computation factorization of number at chip multithreading mode", *Radio Electronics, Computer Science, Control*, 3, pp. 117–122. DOI: https://doi.org/10.15588/1607-3274-2019-3-13
8. Montgomery, P.L. (1994), "A Survey of Modern Integer Factorization Algorithms", available at: https://ir.cwi.nl/pub/18252/18252B.pdf (accessed 11 September 2025).
9. Detto, S. (2025), "The New Fermat-Type Factorization Algorithm", *arXiv preprint*, arXiv:2503.07151, pp. 1–33. DOI: https://doi.org/10.48550/arXiv.2503.07151
10. Kozukalov, M. and Boiko, M. (2020), "Research and analysis of number factorization algorithms", *ΛΌГОΣ. The Art of Scientific Thought*, pp. 64–68. DOI: https://doi.org/10.36074/2617-7064.10.012
11. Boudot, F., Gaudry, P., Guillevic, A., Heninger, N., Thomé, E. and Zimmermann, P. (2022), "The state of the art in integer factoring and breaking public-key cryptography", *IEEE Security & Privacy*, 20(2), pp. 80–86. DOI: https://doi.org/10.1109/MSEC.2022.3141918
12. Yareschenko, V. and Kosenko, V. (2024), "Low-energy coding method in data transmission systems", *Innovative Technologies and Scientific Solutions for Industries*, 3(29), pp. 121–129. DOI: https://doi.org/10.30837/2522-9818.2024.3.121
13. Rabah, K. (2006), "Review of methods for integer factorization applied to cryptography", *Journal of Applied Sciences*, 6(2), pp. 458–481. DOI: https://doi.org/10.3923/jas.2006.458.481
14. Lteif, G. (n.d.), "Integer Factorization Algorithms: A Comparative Analysis", available at: https://softwaredominos.com/home/science-technology-and-other-fascinating-topics/integer-factorization-algorithms-a-comparative-analysis/ (accessed 11 September 2025).
15. Barnes, C. (n.d.), "Integer Factorization Algorithms", available at: https://connellybarnes.com/documents/factoring.pdf (accessed 11 September 2025).
16. Wang, B., Hu, F., Yao, H. and Wang, C. (2020), "Prime factorization algorithm based on parameter optimization of Ising model", *Scientific Reports*, 10(1), 7106. DOI: https://doi.org/10.1038/s41598-020-62802-5
17. Somsuk, K. (2020), "The new integer factorization algorithm based on Fermat's Factorization Algorithm and Euler's theorem", *International Journal of Electrical and Computer Engineering (IJECE)*, 10(2), pp. 1469–1476. DOI: https://doi.org/10.11591/ijece.v10i2.pp1469-1476
18. Kendre, S. (n.d.), "Integer Factorization Algorithms", available at: https://sauravkendre.medium.com/integer-factorization-algorithms-8f3937502bcc (accessed 11 September 2025).

Received (Надійшла) 01.11.2025
Accepted for publication (Прийнята до друку) 21.05.2026
Publication date (Дата публікації) 27.06.2026

Відомості про авторів / About the Authors

Пєвнєв Володимир Яковлевич – доктор технічних наук, доцент, Національний аерокосмічний університет "Харківський авіаційний інститут", професор кафедри комп'ютерних систем, мереж і кібербезпеки, Харків, Україна;
Vladimir Pevnev – Doctor of Technical Sciences, Associate Professor, National Aerospace University "Kharkiv Aviation Institute", Professor of the Computer Systems, Networks and Cyber Security Department, Kharkiv, Ukraine;
e-mail: v.pevnev@csn.khai.edu
ORCID ID: https://orcid.org/0000-0002-3949-3514
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=57194525720

ISSN 2522-9818 (print) *Сучасний стан наукових досліджень та технологій в промисловості. 2026. № 2 (36)*
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2026. No. 2 (36)*

Юдін Олесь Вікторович – Національний аерокосмічний університет "Харківський авіаційний інститут", здобувач вищої освіти ступеня доктора філософії кафедри комп'ютерних систем, мереж і кібербезпеки, Харків, Україна;
Oles Yudin – National Aerospace University "Kharkiv Aviation Institute", Postgraduate Student of the Computer Systems, Networks and Cyber Security Department, Kharkiv, Ukraine;
e-mail: o.yudin@csn.khai.edu
ORCID ID: https://orcid.org/0009-0006-8079-0488
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=58882520600

КЛАСИФІКАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ АЛГОРИТМІВ ФАКТОРИЗАЦІЇ

Предметом дослідження є алгоритми факторизації, а саме експериментальна перевірка ефективності сучасних алгоритмів цілочисельної факторизації, виявлення закономірностей між алгоритмами факторизації та розміром чисел, що факторизуються, з погляду часу, необхідного для виконання факторизації. **Мета роботи** – аналіз продуктивності алгоритмів факторизації з використанням різних складених чисел, зокрема чисел середнього розміру (~20 розрядів), що дає змогу оцінити їх ефективність і час виконання. У процесі дослідження необхідно виконати такі **завдання:** систематизувати сучасні алгоритми факторизації, експериментально перевірити ефективність сучасних алгоритмів факторизації Полларда (p та $p–1$), методу еліптичної кривої, алгоритму Діксона й квадратичного решета. Для досягнення мети використовувалися загальнонаукові **методи:** аналіз предметної галузі та математичного апарату, теорія множин, числа й поля, планування та експериментальне дослідження. **Досягнуті результати.** Експериментальні дослідження продемонстрували, що алгоритми Полларда ефективні для чисел з малими дільниками, але втрачають продуктивність зі збільшенням розміру чисел. Метод еліптичної кривої довів свою доцільність у пошуку дільників середнього розміру та кращу масштабованість порівняно з класичними стохастичними методами. Алгоритм Діксона, незважаючи на відносну простоту реалізації, продемонстрував стохастичні коливання часу виконання, що обмежує його практичну цінність у сценаріях зі строгими часовими обмеженнями. Найбільш стабільних і передбачуваних результатів було досягнуто для квадратичного решета, яке довело його придатність для факторизації чисел середнього розміру й забезпечило найменший розкид значень часу за умов багаторазового виконання на ідентичних наборах даних. **Висновки.** Експерименти повністю підтверджують теоретичні очікування щодо продуктивності досліджуваних методів. Результати свідчать про те, що прості алгоритми (метод Полларда й метод еліптичної кривої) ефективні для попереднього оброблення та виявлення слабких ключів, тоді як квадратичне решето є оптимальним вибором для факторизації чисел середнього розміру. Для великих модулів RSA практично використовувати більш складні алгоритми, наприклад загальне решето числового поля. Подальший розвиток алгоритмів факторизації передбачає паралелізацію процесу факторизації та розроблення алгоритмів, що застосовують скринінг неможливих розв'язків.

Ключові слова: цілочисельна факторизація; криптографія; метод Полларда; метод еліптичної кривої; квадратичне решето; алгоритм Діксона; ефективність алгоритму.

Бібліографічні описи / Bibliographic descriptions

Пєвнєв В. Я., Юдін О. В. Класифікація та експериментальні дослідження ефективності алгоритмів факторизації. *Сучасний стан наукових досліджень та технологій в промисловості*. 2026. № 2 (36). С. 109–120. DOI: https://doi.org/10.30837/2522-9818.2026.2.109

Pevnev, V., Yudin, O. (2026), "Classification and experimental studies of the factorization algorithms efficiency", *Innovative Technologies and Scientific Solutions for Industries*, No. 2 (36), P. 109–120. DOI: https://doi.org/10.30837/2522-9818.2026.2.109