Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/170103

Improving AKS Algorithm. Proving the Simplicity of Integers

*O. Kozachok**

Department of Algebra and Computer Mathematics, Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

Received: 21 Sep. 2022, Revised: 22 Oct. 2022, Accepted: 24 Nov. 2022.
Published online: 1 Jan. 2023.

Abstract: Agrawal-Kayal-Saxena (AKS) theorem was proved, from which the above-mentioned algorithm is directly derived. The study has shown that the problem of simplicity is in the complexity class P of polynomial problems. An advanced AKS algorithm is proposed, which will simplify the initial AKS algorithm and ensure its implementation to determine the simplicity of integers. To implement this task it is necessary to reduce algorithm computational complexity. The block diagram of the algorithm for performing the advanced AKS test is presented. The theorems for prime numbers presented in the form of polynomials were formulated and proved. The AKS algorithm consisting of two phases is implemented: at the first stage, the corresponding parameters *r* and *s* were found, and at the second stage, the identity for different values of *b* presented in the form of consecutive squares was checked. The adequacy of the algorithm for checking numbers for simplicity is proved by the example of a generated arbitrary number of 500 orders. Comparative characterization of the improved AKS test and the Miller-Rabin test was carried out. 50,000 tests were conducted. The maximum test time was 2.3089 s.

Keywords: AKS algorithm, Miller-Rabin test, prime numbers, theorem, lemma

1 Introduction

The problem of estimating the distribution of prime numbers and finding them has been relevant since the time of Euclid. The formal relations presented in [1] to the mechanisms of forming composite numbers of form {6k ± 1}, k = 1, 2, 3, ... allowed substantiating and implementing new, more productive algorithms, simple numbers on arbitrary intervals of natural series, as well as proving Legendre and Brocard hypotheses, confirming Bertrand's postulate and linking other interval problems [1]. The obtained interval estimates are used to improve algorithms for information protection and encryption/decryption with prime numbers. Modern cryptography requires a large random prime in many situations. An example is the RSA cryptosystem. Integer simplicity checking is an algorithm that determines whether an input number is prime. All simplicity tests, in fact, look for a sign of the number complexity. If not found, they answer that the number is prime. There are usually two types of simplicity tests [2]:

determined - issue a certificate of superiority; they answer that the number n given at the input is prime only if it really is; probabilistic - they omit some complex numbers (indicating their primacy), but they are never wrong in the case of prime numbers.

There is a significant difference between the statement that a number is complex (due to the failure of the simplicity test) and the indication of its distribution (factorization). In particular, there is an effective (polynomial) algorithm for checking simplicity, while the still known factorization algorithms are much slower [3].

The naive test of simplicity is the simplest deterministic test of simplicity [3]. The number n given at the input is separated m = 2,3, ..., $\lfloor\sqrt{m}\rfloor$. If the remainder is 0 in any division, the test returns that n is complex, otherwise n is simple. Note that if n is complex, one also finds its nontrivial divisor. Unfortunately, this is a very slow method of testing simplicity and it requires $O\left(\sqrt{n}\,lg^2\,n\right)$ steps [3].

Probability tests for simplicity work according to this scheme:

One chooses a random number that matches the condition a∈ {0,1,2,…, n − 1}.

One checks the identity of the numbers a and n depending on the specific test. If the identity is false, then n is complex and a is a witness of complexity (but not necessarily a divisor of n). If the identity is true, then a passes the test (which does not necessarily mean that it is the first) [3]. Agrawal et al. [4] demonstrated the first effective O ($lg^{12}n$) deterministic simplicity test. After some processing, this algorithm works on O (lg^6n). However, in practice, this algorithm works much slower than fast probability tests. On the otherhand, if the generalized Riemann hypothesis is true, then every odd complex number n has a witness of complexity b in the Miller-Rabin test, so $0 < b < 2log^2n$. This would mean that the Miller-Rabin test is deterministic and works very quickly. In Chan and Norrish, the AKS simplicity algorithm is described [5]. This algorithm was first published in “PRIMES is in P” [4], [6]. This is a

*Corresponding author e-mail: oleksandrakozachok@ukr.net

breakthrough in terms of computational complexity theory, as AKS is the first fully deterministic polynomial test of simplicity [7]. The algorithm itself is simple and it is not difficult to prove its correctness. The AKS method is considered in many works, in particular, in Dimitrov [3]. This paper treats the problem based on the original article and the work of Bernstein [8], which gives a good overview of AKS improvements. First, the theorems underlying the AKS method will be formulated and proved. In practice, it turns out that in almost all cases, several Miller-Rabin tests are enough. For example, it has been estimated that there is only one compound odd number less than 2.5·1010, namely 3215031751, which is very pseudo-simple for bases 2,3,5 and 7. However, mathematicians are still looking for quick deterministic simplicity tests. Most recently, in 2002, Agrawal et al. [4] demonstrated the first effective O (lg12n) deterministic simplicity test. After some processing, this algorithm works on O (lg6n)). In practice, this algorithm works much slower than fast probability tests. On the other hand, if the generalized Riemann hypothesis is true, then every odd complex number n has a witness of complexity b in the Miller-Rabin test, so 0 <b <2log2n. This would mean that the Miller-Rabin test is deterministic and works very quickly [9]. The AKS algorithm allows one to get the result of checking a number for simplicity and determine whether the number is prime or compound. The advantage of the method is that the result of checking the number for simplicity is not probabilistic, and the method itself is not based on unproven assumptions [3].

The theoretical importance of the AKS algorithm lies, in fact, in solving an important mathematical problem that has been discovered for over 2,000 years. As for the practical question, the biggest application is to generate cryptographic keys, because it is necessary to calculate very large prime numbers. Several sources were used to compile this report [3]. The main contribution of Agrawal et al. [4] is to demonstrate the correctness and efficiency of the AKS algorithm. In contrast to the original article [6] and the book [10], this paper uses algebraic structures of groups, rings and fields, which naturally arise in the study of this problem [3]. The classical AKS simplicity verification algorithm is used to determine elliptic curve theory results, with special emphasis on the properties of isogeny between curves (Velu's theorem), and toThe theoretical importance of the AKS algorithm lies, in fact, in solving an important mathematical problem that has been discovered for over 2,000 years. As for the practical question, the biggest application is to generate cryptographic keys, because it is necessary to calculate very large prime numbers. Several sources were used to compile this report [3]. The main contribution of Agrawal et al. [4] is to demonstrate the correctness and efficiency of the AKS algorithm. In contrast to the original article [6] and the book [10], this paper uses algebraic structures of groups, rings and fields, which naturally arise in the study of this problem [3]. The classical AKS simplicity verification algorithm is used to determine elliptic curve theory results, with special emphasis on the properties of isogeny between curves (Velu's theorem), and to construct the extension of the function body from Kuven and Lervier elliptic cycles [3]. The author presented an elliptical criterion of simplicity [10], the ultimate goal of which was to present the algorithm "Elliptical AKS", in particular the construction of a corresponding instance of an elliptical cyclic ring using the theory of complex multiplication. The algorithm proposed by Lenstra [10] is a variant of the original AKS test [3]. This is not the most effective, but the easiest way to prove the simplicity of numbers. This algorithm can be used when the input number is a prime number in decimal notation [10]. Bernstein's advanced AKS algorithm can perform calculations in a few tens of seconds [8]. For primes above 40, this procedure will take up to several hours. After a thorough review of the improved version by Bernstein, Jin Zhengping, and others, improved algorithms were proposed, but they also had some shortcomings. However, the author noted that the algorithm still needs to be improved so that it can be used in practice [11,12,13,14]. None of the existing AKS algorithms is suitable for practical applications for security reasons, as it is impossible to achieve completeness of the algorithm without a huge amount of time to calculate [15,16].

The purpose of this study is to improve the original AKS algorithm to provide an efficient approach to solving the global problem of proving the simplicity of integers.

To achieve this goal, it is necessary to solve the following tasks:

- perform an analysis of the necessary and sufficient conditions for the AKS;

- perform an equivalence transformation in the equations of the algorithm using Fermat's little theorem;

- present an improved AKS algorithm for checking integers by successive squaring;

- prove that the developed algorithm can be represented as the Miller-Rabin algorithm with two additional restrictions when narrowing the set of base numbers.

2 Methods and materials

The problem of simplicity is a problem of a solution, and the author of the article consider a deterministic algorithm that solves it (AKS algorithm). If the study proves that this algorithm is also polynomial, then the simplicity problem is in P. The study will focus solely on seeing that AKS is polynomial. The study will not delve into the calculation of the best execution time but will overestimate the number of steps taken. Then, this will be about AKS improvements that will speed up the algorithm. To see that the algorithm is solved in polynomial time, one of the things the study needs to prove is that in step 2 the number r does notbecome too large. To do this, the following notation was entered.

Designation 1. Let $P(x) = \overline{a}_d x^d + \overline{a}_{d-1} x^{d-1} + \cdots + a_2 x^2 + \overline{a}_1 x + \overline{a}_0 \in Z_p[x]$ and $P(x)$ (1)

Let P(x) be a polynomial of degree less than or equal to d, the coefficients of which $a_i \in Z_p$ are classes modulo p of the coefficients $a_i \in Z_p$ for P(x) to all $a_i \in \{1,2,\ldots,n\}$. Thus:

$$\overline{P}(x) = \overline{a}_d x^d + \overline{a}_{d-1} x^{d-1} + \cdots + a_2 x^2 + \overline{a}_1 x + \overline{a}_0 \in Z_p[x] \quad (2)$$

Now it is needed to formulate and prove the desired theorem.

A block diagram implemented in Python was used to test sets of numbers for simplicity (Figure 1).

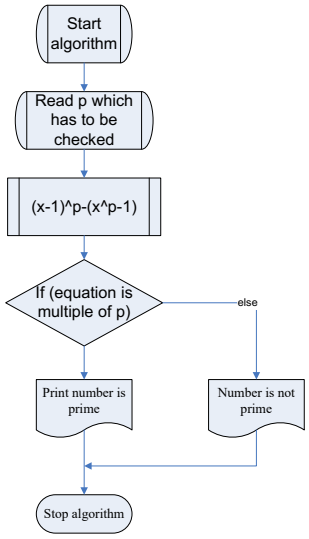

Fig. 1. Block diagram of the algorithm for performing the advanced AKS test

Source: Mathematical bases of AKS method

Briefly, the block diagram can be explained as follows. After the start of the algorithm, the polynomiality of the number is checked. If the number is polynomial, then the number is prime, and if it is not polynomial, then it is not prime.

Assume that for a prime number then *n* then $w(x)^n \equiv w(x^n)$ (mod n). Especially for monomony *w (x) = x – b,* the following takes place:

$$(x-b)^n = x^n - b^n = x^n - b \text{ (mod n)} \quad (3)$$

Such congruence of polynomials does not occur when n is a composite number. In fact, when a prime number p is divisible by n to the power of *a,* coefficients at x^{kn/p^α} of polynomial $(x-b)^n$ is not divisible by p for gcd *(k, n)* = 1. It is not possible to calculate the whole polynomial $(x-\alpha)^n$, since it has an exponential number of coefficients. However, it is possible to test the modulus of equality by some polynomial of degree r.

$$(x-\alpha)^n = x^n - \alpha \text{ (mod } n, (x^r - 1) \quad (4)$$

Polynomial $(x^r - 1)$ was chosen for its special properties. If one chooses a random polynomial of degree r instead, one would get a fairly strong probability test, and therefore one can use the property $(x^r - 1)$ in the proof. In addition, one proceeds from the fact that the calculation of the module $(x^r - 1)$ is very easy to perform.

The AKS algorithm mainly consists of two phases: in the first stage the corresponding parameters r and s are found, in the second stage the identity 1 is checked for the presence of s different values of b. If the number n passes all these tests, it must be an indicator of the prime number degree $n = p^a$, which can be pre-checked separately. Thus, the theorem can be formulated:

Theorem 1. Let the n — a positive integer, q and r are prime numbers such that q|r – 1 and $n^{(r-1)/q}$ mod $r \notin \{0.1\}$r 0 {0,1}. Let S — set of s integers, such that $gcd(n, b-b') = 1$ for different $b, b' \in S$. Suppose also that

$$\binom{s+q-1}{s} \geq n^{2|\sqrt{r}|} \quad (5)$$

and that for every $b \in S$ around $Z_n[x]/x^r - 1$ equality comes true

$$(x+b)^n \equiv x^n + b \quad (6)$$

Then n is a power of a prime number.

Proof. The idea of proof is contained in [10]. The formulation of the theorem and a brief proof can be found in Dimitrov [3].

Based on this theorem, one can show a polynomially determined algorithm for checking the simplicity of a number. It is necessary to show that there are corresponding numbers q, r, s, except for polynomials. In Bernstein [8], reference was made to Fourier's theorem that there are many primes r such that r-1 has a prime divisor q, bigger than $r^{2/3}$. This is a complex statement. In addition, based on well-established hypotheses, one can expect the existence of many prime numbers r, such that (r - l) / 2 is prime, which leads to a better estimate of the AKS algorithm complexity. Therefore, the AKS theorem has been improved, so there is no need to refer to the Fourier theorem.

Theorem 2. Let n — positive integer, r is a prime number. Let $v = ord_r(n)$ - order n modulo r. Let S be the set of s integers such that $gcd(n, b-b') = 1$ for different $b, b' \in S$. Assume that for every d that is a divisor $\phi(r)/v$

$$\binom{s+q-1}{s} \geq n^{2|\sqrt{\phi(r)/d}|} \quad (7)$$

and that for every $b \in S$ around $Z_n[x]/x^r$ there will be fair equality:

$$(x+b)^n \equiv x^n + b \quad (8)$$

Then n is a power of a prime number.

Note that the largest constraint is most likely for the maximum $d = <p\ (r)\ fv$ (if one omits the whole part).

When n is the prime root of the modulus r, then $v=\varphi(r)$ and the theorem is simplified:

Theorem 3. Let n — a natural number, and r is a prime number. Let n be a simple root modulo r. Let S be the set of s integers, such that $gcd(n, b-b') = 1$ for different $b, b' \in S$. Suppose also that

$$\binom{S+\phi(r)-1}{S} \geq n^{2\lfloor\sqrt{\phi(r)}\rfloor} \quad (9)$$

and that for every$b \in S$ around $Z_n[x]/x^r - 1$, there will be the following equality:

$$(x+b)^n \equiv x^n + b \quad (10)$$

Then n is a power of a prime number.

Proof of the basic AKS theorem

First, the proof of Theorem 3 is shown, which is a simpler version, where one assumes that n is the prime root of the module r. Assume that p is a prime divisor of n. The author formulates the proof of the theorem and divide it into several lemmas.

Lemma 4. There are $(i_1, j_2) \neq (i_2, j_2)$: $i_1, i_2, j_2, h \geq 0$, such as for $t = n^{i_1}p^{j_1}, u = n^{i_2}p^{j_2}$ there are $t \equiv u(mod\ r)$, t = u (mod r), moreover $|t-u| \langle n^{2\lfloor\sqrt{\phi(r)}\rfloor}$.

Proof of the lemma. It is known that $x^t = x^u$ (mod x^r - 1) only when $t = u$(mod r). Let us consider the value (mod r) for pairs *(i, j),* that meet the condition $0 \leq i, j \leq \lfloor\sqrt{\phi(r)}\rfloor$. These values may be different: $|Z_r^*| = \phi(r)$, whereas there are more *(i , j)* pairs, because $1 + \lfloor\sqrt{\phi(r)}\rfloor^2$. Thus, based on the Dirichlet principle, there will be different pairs $(i_1, j_1) \neq (i_2, j_2)$, such as $n^{i_1}p^{j_1} \equiv n^{i_2}p^{j_2}(mod\ \ r)$. Let us mark $t = n^{i_1}p^{j_1}$ and $u = n^{i_2}p^{j_2}$. One should note that $1 \leq t, u \leq n^{2\lfloor\sqrt{\phi(r)}\rfloor}$, thus, $(t-u) < n^{2\lfloor\sqrt{\phi(r)}\rfloor}$.

Remark. One should know that $t = u$, then, concerning $i_1 \neq i_2$, n should be a power of p.

Lemma 5. Let G — set of polynomials $\{\prod_{b \in S}(x+b)^{e_b} : e_b \geq 0\}$. Let t, u — numbers from the previous lemma. For each polynomial $g \in G$ there is $g^t = g^u$ in $Z_p[x]/x^r - 1$.

Proof of the lemma. Identity $g^l = g^u$ — multiplicative. Therefore, it is enough to show this for monomials $g = x + b$.

Because for $b \in S$

$$(x+b)^n \equiv x^n + b(mod\ \ n, x^r - 1), \quad (11)$$

then

$$(x+b)^n \equiv x^n + b(mod\ \ p, x^r - 1) \quad (12)$$

Substituting x^{n^i} instead of x , one gets:

$$\left(x^{n^i}+b\right)^n \equiv x^{n^{i+1}} + b\left(mod\ \ p, x^{n^i r} - 1\right) \quad (13),$$

because $x^r - 1 | x^{kr} - 1$,

$$\left(x^{n^i}+b\right)^n \equiv x^{n^{i+1}} + b(mod\ \ p, x^r - 1) \quad (14)$$

and therefore, by induction on *i:*

$$(x+b)^{n^i} = x^{n^i} + b(mod\ \ p, x^r - 1)\,. \quad (15)$$

Given the fact that $w(x^p) \equiv w(x)^p$ (mod p), one gets:

$$(x+b)^{n^i p^i} = \left(x^{n^i}+b\right)^{p^i} = x^{n^i p^i} + b(mod\ \ p, x^r - 1) \quad (16)$$

Since $x^r = 1$ and $t = u$ (mod r), then

$$(x+b)^t = x^t + b = x^u + b = (x+b)^u(mod\ \ p, x^r - 1) \quad (17)$$

3 Results

Improving AKS method

The AKS algorithm can be improved in several ways by weakening the required estimates. As a result, one can improve the time complexity of the algorithm (by a constant), because to prove the theorem requires smaller parameters r, s, and then one will have fewer calculations to perform. Most of the amendments concern the estimates of one of the proofs.

Corrections when finding u, t

Lemma 4 shows what one can find $u \equiv t$ (mod r) of type $n^i p^i, i, j \geq 0$, such that $|u-t| < n^{2\lfloor\sqrt{\phi(r)}\rfloor}$. One can make the following improvements.

One should note that in $\frac{Z_p[x]}{h(x)}$ raising the polynomial to the pth degree is a reversible n

= 47940917743537973692644475812229992529786814872499245562292570755330643813134833171004584999222590110235044488677770714926858791225185306522918439447628041208121912916298768103979134963553045853413830416284659305548512617203710238027925195359522683135371055658904063286578635833582214067827360654259330234894372332159730250159696606450783603910731017154458224336849289932324339688284385593377864655882803248863009189826723891296818317568544802394638532517956101892005378546853479529063224213885215 03

For a given number n, $r = 2755759$; $\phi(r) = o_r(n) = 27555758$, (20)

$i = j = 0.047\phi(r) = 129520$, (21) $d = 0.5\phi(r) = 1377879$, (22)

$log_2(0.5 \cdot 129520) = log_2(0.5 \cdot 129520) = 1581626.22$, and $log_2(1377897) = 1581407.72$.

The set S1 is presented as follows.

Assume that $n < 10^{500} = 2^{1650}$. Then $\sqrt{n} < 2^{825}$.

Let us consider the number 2 in the following form: $2^{2u_i} < \sqrt{n}$. Then $2, 2^2, 2^4, 2^8, 2^{16}, 2^{32}, 2^{64}, 2^{128}, 2^{256}, 2^{512} \in S_1$

Let us consider the number 3 in the following form: $3^{2u_i} < \sqrt{n}$. It follows that $3, 3^2, 3^4, 3^8, 3^{16}, 3^{32}, 3^{64}, 3^{128}, 3^{256}, 3^{512} \in S_1$

operation, since the size of the multiplicative group $p^{deg\ \ h} - 1$, then:

$$(w(x)^p)^{p^{deg\ \ h}} = w(x)^{p^{deg\ \ h}} = w(x)^{p^{deg\ \ h-1}} w(x) = w(x) \quad (18)$$

Let us suppose that $g^{tp^k} = g^{up^k} \Rightarrow g^t = g^u$. Therefore, it suffices to consider t, u of type $(n/p)^i p^j$, moreover *n/p* is integer. Replacing $n^i p^i$ with $(n/p)^i p^j$ leads directly to the estimate $u, t \le n^{\lfloor\sqrt{\phi(r)}\rfloor}$.

Thus, one should check whether n is an indicator of prime number power. With a positive solution of the algorithm, n is a composite number (Figure 1).

Let us find the smallest prime number $r \ge 3$. Let us choose the following integers:

$$\text{d} \qquad 0 \le d \le \phi(r) - 1$$
$$0 \le i \le d$$
$$0 \le j \le \phi(r) - 1 - d \quad (19)$$

The author chooses S that satisfies the requirement $\binom{2S}{i}\binom{d}{i}\binom{2S-i}{j}\binom{\phi(r)-1-d}{j} \ge n^{\sqrt{\phi(r)/3}}$. Let us define the set

$$S_1 = 2, 2^2, \ldots., 2^{2^{u_t}}; 3, 3^2, \ldots., 3^{2^{u_t}}; 5, 5^2, \ldots., 5^{2^{u_t}};$$

$b, b^2, \ldots., b^{2^{u_t}}$. One should note that the sum of a series (set) is equal to $S_1 = \sum_{k=1}^{t}(u_k + 1)$.

Let us choose a natural number of 500 orders and investigate it using the advanced AKS algorithm. To test, the author uses the simplest Fermat method by dividing by prime numbers. Let us generate the most probable prime number:

Number 4 is 2^2 and it will not be considered, as it has been presented above.

For number 5: $5^{2u_i} < \sqrt{n}$. Then $5, 5^2, 5^4, 5^8, 5^{16}, 5^{32}, 5^{64}, 5^{128}, 5^{256}, 5^{512} \in S_1$. For other numbers, the author checks similarly.

It follows that

$gcd(n, |b - b'|) = 1, forall$pairwise differentb, b' ∈ S1 ;

$gcd(n, |b + b'|) = 1, forall$pairwise differentb, b' ∈ S1

$gcd(n, |bb' - 1|) = 1, forall$b, b' ∈ S1

$gcd(n, |bb' + 1|) = 1, forall$b, b' ∈ S1

$b^{n-1} = 1\, mod \quad n forall b \in S_1$

$(x - b)^n = x^n - b(mod \quad n, x^r - 1) forall b \in S_1$

Table 1. Testing the adequacy of the advanced AKS algorithm

Range n	Prime numbers	Test time, s
2^2 <n< 2^{16}	12	0.1005
2^{16} <n< 2^{64}	32	0.1082
2^{64} < n < 2^{128}	48	1.0006
2^{128} < n < 2^{256}	50	1.2039
2^{256} < n < 2^{512}	256	2.3089

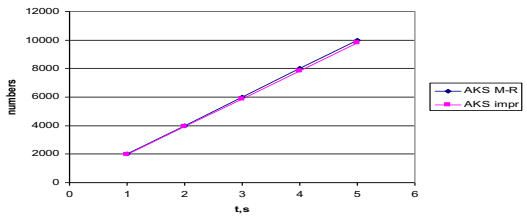

Fig. 2. Comparative characteristics of Miller-Rabin algorithms and AKS algorithm

Let us consider arbitrary ranges of prime numbers presented in Table 1, for example, in the number of 10,000 random numbers each. All numbers were tested according to the Miller-Rabin algorithm and using the advanced AKS test (Figure 2). The Miller–Rabin primality test or Rabin–Miller primality test is a probabilistic primality test: an algorithm which determines whether a given number is likely to be prime. It is of historical significance in the search for a polynomial-time deterministic primality test. Its probabilistic variant remains widely used in practice, as one of the simplest and fastest tests known.

The maximum testing time was 2.3089 s. All 50,000 tests show that the two algorithms give consistent results. The error of calculations according to the Miller-Rabin test and the improved AKS algorithm is zero in all cases. An arbitrary set of basic numbers according to two algorithms is 22 prime numbers.

4 Discussion

It is known that tests to establish the simplicity of numbers, including in cryptographic schemes, face difficulties related to the execution time of the algorithm [15,17,18]. The author has proposed an improvement of the AKS algorithm for the set of integers represented by consecutive squares. The original AKS algorithm pays more attention to obtaining the best execution time, and therefore differs in some respects from the one formulated in this paper. In statements instead of "not more than r – 1", they indicate "n ot more than $2\sqrt{r}\, log_2 n$". To see this, one would also need to make some changes to the previous results, for example, as a result, 2.25 hypothesis l > t-1 can be replaced by a weaker condition $l \ge 2\sqrt{t}\, log_2 n - 1$. All this leads to the optimization of the execution time in the algorithm, because one can replace the limit r of the third step by $2\sqrt{r}\, log_2 n < r$. In addition, it should be noted that it is not really necessary to require the primacy of r. [9] states that

one can study irreducible factors r modulo the prime number p, and r and p are mutually prime, and Lemma 2.29 will still be tested (although the proof becomes much more complicated). Given that this lemma is the only place in the proof of Theorem 2, where the primacy of r is applied, one concludes that this assumption can be excluded from the theorem and, consequently, from the algorithm [3]. Again, this speeds up the algorithm for two reasons: 1. One can do without checking the simplicity on r (in this case the Eratosthenes sieve) 2. The smallest positive integer r that checks that the order (n) > 4 (log2 n)2 cannot be simple and is therefore found earlier than in the case of its simplicity requirement. Note that step 2 is accelerated. In the literature on efficient algorithms, one can find information on other AKS improvements using the most well-known polynomial arithmetic algorithms [3]. Usually, the same problem can have different algorithms that solve it, so one needs to somehow compare them to know which one is most appropriate [19,20]. Intuitively, algorithmic complexity is a theoretical metric that is applied to algorithms. The comparison between algorithms can be approached in two main ways: temporal complexity (the time required by the algorithm to solve the problem) and spatial complexity (the amount of memory required by the algorithm) [11,21,22,23,24]. In the current study, the author limited themselves to temporal complexity, so when this is about ordinary complexity, this is about temporal complexity. Thus, the author concludes that the algorithmic complexity is the number of (temporary) resources required by the algorithm to solve the problem, and, therefore, allows one to determine the effectiveness of this algorithm. The complexity of a problem is defined as the complexity of the best algorithm that solves it, and its study is known as complexity theory [25,26,27,28]. To understand these definitions, it is necessary to introduce the concepts of algorithm execution time and asymptotic growth. In Bernstei [8], the improvements provided by other researchers in this field are analyzed. Shortly after Agrawal et al. [4] published their paper, several people developed improvements to this algorithm. Initially, with adjustments made by Lenstra and Pomerance [29], AKS was set for $r = O\left((log_2 n)^{4'5}\right)$. Later, they also made changes to the algorithm of Berstein, as well as Berrizbeitia [8]. The basic idea is essentially the same as in AKS, but another Q polynomial is used to prove congruence in step 3. This significantly improves execution time, but instead makes it much more difficult to prove.

5 Conclusions

The advanced AKS algorithm provides an effective approach to solving the global problem of proving the simplicity of integers. The author has improved the original AKS algorithm, as it is characterized by algorithm complexity and difficulties in implementing calculations. High reliability can be achieved with a public key scheme based on the discrete logarithm of an elliptic curve when the number n of 500 orders is of interest. An advanced algorithm can be used on a supercomputer to test the simplicity of n. However, this type of complete algorithms is not feasible for RSA (Rivest-Shamir-Adleman algorithm) in electronic transactions. The analysis of necessary and sufficient conditions for AKS is presented in the article and the equivalence transformation in the algorithm equations is performed using the little Fermat theorem. An advanced AKS algorithm for checking integers by sequential squaring in the form of squares $b, b^2, b^4, b^8, b^{16}, b^{32}, b^{64}, b^{128}, b^{256}, b^{512} \in S_1$ is presented. It is proved that these algorithms can be applied to a number $-b, -b^2, -b^4, -b^8, -b^{16}, -b^{32}, -b^{64}, -b^{128}, -b^{256}, -b^{512} \in S_1$. Thus, the algorithm can be used to test twice as many numbers. Accordingly, the computational complexity of the algorithm is halved. It is proved that the developed algorithm can be represented in the form of the Miller-Rabin algorithm with two additional restrictions when narrowing the set of base numbers. The results of experimental studies have shown that the efficiency of the advanced AKS algorithm may be higher than the Miller-Rabin algorithm and is more suitable for practical application.

Acknowledgments

Not applicable.

Declaration of conflicting interests

The author declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author received no financial support for the research, authorship, and/or publication of this article.

Data availability

Data will be available on request.

References

[1] K. Ahmad, A. Kamal and K. A. B. Ahmad, *Prime number*, in Emerging security algorithms and techniques. K. Ahmad, M. N. Doja, N. I. Udzir and M. Pratap, Eds. Chapman and Hall/CRC, New York: 27-45, (2019).

[2] B. N. Tiwari, J. K. Kuipo, J. M. Adeegbe and N. Marina, Optimized AKS primality testing: a fluctuation theory perspective, *Cryptography*, **3**, 12 (2019).

[3] D. G. Dimitrov, On the software coputation of the formulae for the n-th prime number, *Notes Number Theory Discrete Math.*, **25**, 198-206 (2019).

[4] M. Agrawal, N. Kayal and N. Saxena, PRIMES is in P, *Ann. Math.*, **160**, 781-793 (2004).

[5] H. L. Chan and M. Norrish, Mechanisation of the AKS algorithm, *J. Autom. Reason.*, **65**, 205-256 (2021).

[6] F. Bornemann, PRIMES is in P: a breakthrough for «Everyman», *Notices Am. Math. Soc.*, **50**, 545-553 (2003).

[7] D. Garcia-Martin, E. Ribas, S. Carrazza, J. I. Latorre and G. Sierra, The Prime state and its quantum relatives, *Quantum*, **4**(371) (2020), doi: 10.22331/q-2020-12-11-371.

[8] D. J. Bernstein, Proving primality in essentially quartic random time, *Mathematics of Computation*, **76**, 389-403 (2007).

[9] F. Qin, L. Yao, C. Lu, C. Li, Y. Zhou, C. Su, B. Chen and Y. Shen, Phenolic composition, antioxidant and antibacterial properties, and in vitro anti-HepG2 cell activities of wild apricot (Armeniaca Sibirica L. Lam) kernel skins, *Food Chem. Toxicol.,* **129**, 354-364 (2019).

[10] B. N. Prasad Rao and M. Rangamma, *A primality test and a theorem on twin primes*, in Mathematical analysis and computing: ICMAC 2019. R. N. Mohapatra, S. Yugesh, G. Kalpana and C. Kalaivani, Eds. Springer, Singapore, 1-6 (2019).

[11] A. Hegde and P. Devaraj, *Heuristics for the construction of counterexamples to the agrawal conjecture*, in Mathematical analysis and computing. ICMAC 2019. R. N. Mohapatra, S. Yugesh, G. Kalpana and C. Kalaivani, Eds. Springer, Singapore, 537-543 (2019).

[12] H. Li and R. Wang, *A primality test based on modular hyperbolic curves,* in 2020 2nd Int. Conf. Machine Learning, Big Data and Business Intelligence (MLBDBI), 32-35 (2020).

[13] M. Karatay, A. Aylanç and S. Ozkan, Algorithm on finding twin prime numbers, *J. Mod. Technol.Eng.*, **4**, 190-194 (2019).

[14] H. L. Chan, *Primality testing is polynomial-time: a mechanised verification of the AKS algorithm*. Ph.D. dissertation, The Australian National University, Australia (2019).

[15] R. Patgiri, M. D. Borah and L. D. Singh, *SecretStore: a secrecy as a service model to enable the Cloud Storage to store user's secret data*, in 2021 26th IEEE Asia-Pacific Conf. Commun. (APCC), 198-204 (2021).

[16] A. P. Ferreira and R. Sinnott, *A performance evaluation of containers running on managed kubernetes services*, in 2019 IEEE Int. Conf. Cloud Comput. Technol. Sci. (CloudCom), 199-208 (2019).

[17] D. Huang and Y. Kang, Primality testing for numbers of the form h· 2n±1, *J. Syst. Sci. Complex.*, **32**, 1473-1478 (2019).

[18] F. O. Igbinovia and J. Krupka, *Computational complexity of algorithms for optimization of multi-hybrid renewable energy systems*, in 2018 Int. Conf. Power System Technol. (POWERCON), 4498-4505 (2018).

[19] D. Symak, V. Sabadash, J. Gumnitsky and Z. Hnativ, Z., Kinetic regularities and mathematical modelling of potassium chloride dissolution, *Chem. Chem. Technol.*, **15**, 148-152 (2021).

[20] S. Galbraith, J. Massimo and K. G. Paterson, *Safety in numbers: on the need for robust Diffie-Hellman parameter validation*, in IACR International Workshop on Public Key Cryptography. D. Lin and K. Sako, Eds. Springer, Cham, 379-407 (2017).

[21] F. Mechkene, An efficient algorithm to find all primes in a given interval, *Turk. J. Math. Comput. Sci.*, **11**, 74-77 (2019).

[22] M. R. Albrecht, J. Massimo, K. G. Paterson and J. Somorovsky, *Prime and prejudice: primality testing under adversarial conditions*, in Proc. 2018 ACM SIGSAC Conf. Comput. Commun. Secur., 281-298 (2018).

[23] V. Sabadash, J. Gumnitsky and O. Lyuta, Combined adsorption of the copper and chromium cations by clinoptilolite of the sokyrnytsya deposit, *J. Ecol. Eng.*, **21**, 42-46 (2020).

[24] B. K. Yusuf and K. A. B. Mahmood, Towards cryptanalysis of a variant prime numbers algorithm, *Comput. Sci.*, **5**, 14-30 (2020).

[25] D. Buell, *Mathematics, computing, and arithmetic*, in Fundamentals of Cryptography. Undergraduate Topics in Computer Science. I. Mackie Ed. Springer, Cham, 99-122 (2021).

[26] A. Adve, C. Robichaux and A. Yong, Computational complexity, Newton polytopes, and Schubert polynomials, *arXiv:1810.10361,* **1**, 1-12 (2018).

[27] K. Knežević, *Generating prime numbers using genetic algorithms*, in 2021 44th Int. Convention Inform. Commun. Electronic Technol. (MIPRO), 1224-1229 (2021).

[28] A. Beketaeva and A. Naimanova, Flow structure of the transverse jet interaction with supersonic flow for moderate to high pressure ratios. *Int. J. Mech.,* **12**, 88-95 (2018).

[29] H. W. Lenstra and C. Pomerance, A rigorous time bound for factoring integers, *J. Am. Math. Soc.,* **5**, 483-516 (1992).

Oleksandra Kozachok - Department of Algebra and Computer Mathematics, Taras Shevchenko National University of Kyiv, Kyiv, Ukraine, Volodymyrska 60, 03022.