An Efficient Deterministic Primality Test

Joseph M. Shunia

December 2023
Revised: January 2024

Abstract

A deterministic primality test with a polynomial time complexity of $\tilde{O}(\log^3(n))$ is presented. The test posits that an integer n satisfying the conditions of the main theorem is prime. Combining elements of number theory and combinatorics, the proof operates on the basis of simultaneous modular congruences relating to binomial transforms of powers of two.

1 Introduction

Primality testing has seen remarkable advancements over the past few decades. A significant breakthrough in this field was the AKS primality test, introduced by Agrawal, Kayal, and Saxena (2002) [1]. The AKS test was the first to offer determinism and polynomial-time complexity, a monumental achievement that resolved a longstanding open question in computational number theory [2]. However, despite its theoretical importance, the AKS test has practical limitations due to its relatively high polynomial time complexity, rendering it inefficient for most applications. Agrawal, Kayal, and Saxena gave a time complexity of $\tilde{O}(\log^{12}(n))$ for the AKS test [1]. This bound was lowered significantly by Lenstra and Pomerance (2011) to $\tilde{O}(\log^6(n))$ [3]. Despite this reduction, AKS remains impractical and is mostly unused.

In the field of cryptography, the unique properties of prime numbers are widely exploited to create cryptographic primitives. It is often the case that many large primes must be generated in rapid succession [4]. To make these cryptographic operations practical, fast probabilistic primality tests such as the Baille-PSW primality test (BPSW) [5] or Miller-Rabin (MR) [6] [7] are used instead of AKS when searching for large primes. Probabilistic primality tests are by definition non-deterministic and may erroneously report a composite integer as being prime. Composite integers which pass a probabilistic primality test are relatively rare and are known as pseudoprimes (PSPs) for the respective test [8]. When generating primes for cryptographic purposes, probabilistic primality tests are often combined or repeated with different parameters in order to achieve an acceptable error-bound that makes it almost certain that no composite integer will pass. However, reducing the error-bound requires additional compute and increases running-time, creating a trade-off.

We present a new deterministic primality test that operates in polynomial time with a time complexity of $\tilde{O}(\log^3(n))$. This efficiency gain opens new avenues for practical applications, particularly in cryptography, where fast and reliable primality testing is desirable [9].

Our test is based on a famous conjecture issued by Manindra Argawal while he was an undergraduate student [1]. Let n and r be two coprime positive integers. If the following polynomial congruence holds, then n is prime or $n^2 = 1 \pmod{r}$:

$$(x-1)^n \equiv x^n - 1 \pmod{n, x^r - 1} \tag{1}$$

We alter the conditions slightly to use roots of 2 instead of roots of unity. Through the proof of our

main theorem, we demonstrate that our modified test is equivalent to checking the polynomial congruence $(1+x)^n \equiv 1 + x^n \pmod{n}$, a congruence which is known to hold for only prime integers n [10].

1.1 Structure of the Paper

This paper is structured as follows: We begin by presenting the main theorem which defines our primality test. We follow up with supporting theorems, identities, and lemmas. Then, we present the proof of our main theorem. The proof of our main theorem demonstrates the test's validity for odd prime numbers and its failure for odd composite numbers. Through this, we establish the deterministic nature of our test. We then describe the algorithm used to compute our test and analyze its computational complexity. We conclude with a link to an open source implementation of our test.

2 Statement of Main Theorem

Theorem 1 (Main theorem)**.** Let n be an odd integer > 3 such that $2^{n-1} \equiv 1 \pmod{n}$. Denote d as the least prime greater than 2 such that $n \not\equiv 1 \pmod{d}$. If the following polynomial congruence holds, then either n is prime or n has a prime divisor $p \leq d$:

$$(1+x)^n \equiv 1 + x^n \pmod{n, x^d - 2} \tag{2}$$

3 Supporting Lemmas

Lemma 1. Given $a, b \in \mathbb{Z}^+$ with $b \nmid a$ and $1 < b < \left\lfloor \frac{a}{b} \right\rfloor$, then $\left\lfloor \frac{a}{b} \right\rfloor$ cannot divide a.

Proof. Let $q = \left\lfloor \frac{a}{b} \right\rfloor$. By definition, q is the greatest integer that is less than $\frac{a}{b}$. Thus, $q \cdot b < a < b \cdot (q+1)$.

Suppose, for contradiction, that q divides a. Then there exists an integer k such that $a = k \cdot q$. Substituting $a = k \cdot q$ into the inequality $q \cdot b < a < b \cdot (q+1)$, we get $q \cdot b < k \cdot q < b \cdot (q+1)$. Dividing this inequality by q, we obtain $b < k < b + \frac{b}{q}$.

Since k is an integer, and $b \nmid a$ implies $k \neq b$, the next possible integer value for k is $b+1$. Therefore, $k = b+1$, which gives $a = k \cdot q = q \cdot (b+1)$. However, this leads to a contradiction: $a = q \cdot (b+1)$ implies $a \geq b \cdot (q+1)$, contradicting the established fact that $a < b \cdot (q+1)$. Hence, our assumption that q divides a is false. Therefore, $\left\lfloor \frac{a}{b} \right\rfloor \nmid a$. □

Lemma 2 (Upper bound on floor function and indivisibility)**.** Given $a, b \in \mathbb{Z}^+$ with $1 < b < \left\lfloor \frac{a}{b} \right\rfloor$, then $b \leq \lfloor \sqrt{a} \rfloor$.

Proof. Assume $a, b \in \mathbb{Z}^+$ and $1 < b < \left\lfloor \frac{a}{b} \right\rfloor$. By definition, $\left\lfloor \frac{a}{b} \right\rfloor$ is the greatest integer less than or equal to $\frac{a}{b}$. Hence, $\left\lfloor \frac{a}{b} \right\rfloor \leq \frac{a}{b}$. Since $b < \left\lfloor \frac{a}{b} \right\rfloor$, we have $b^2 < b \cdot \left\lfloor \frac{a}{b} \right\rfloor \leq a$. Taking square roots on both sides of the inequality $b^2 < a$ and considering that b and $\sqrt{a}$ are both positive, we get $b < \sqrt{a}$. Since b and $\sqrt{a}$ are positive integers, and $b < \sqrt{a}$, it follows that $b \leq \lfloor \sqrt{a} \rfloor$. □

Lemma 3 (Bounds on least non-divisor)**.** Let n be an integer such that $n > 3$, then there exists an integer $1 < d \leq \lfloor \log_2(n-1) \rfloor + 2$ which does not divide $n-1$.

Proof. If n is a composite integer > 3, clearly $n-1$ can have at most $\lfloor \log_2(n-1) \rfloor$ prime factors (when $n-1$ is a power of 2). Since 2 is the least prime that may divide $n-1$, there must exist a $D \leq \lfloor \log_2(n-1) \rfloor + 2$ which does not divide $n-1$. □

Lemma 4 (Multiplicative order inequality)**.** Let n be an odd composite integer greater than 3 such that $2^{n-1} \equiv 1 \pmod{n}$. Denote by d the least integer $2 < d < n$ which does not divide $n-1$. Then, $\left\lfloor \frac{n-1}{d} \right\rfloor \neq \operatorname{ord}_n(2)$.

Proof. Consider an odd composite integer $n > 3$ for which $2^{n-1} \equiv 1 \pmod{n}$. According to the properties of the multiplicative order modulo n, the smallest positive integer k such that $2^k \equiv 1 \pmod{n}$ defines $\operatorname{ord}_n(2)$, that is, $k = \operatorname{ord}_n(2)$. Since n is composite and $2^{n-1} \equiv 1 \pmod{n}$, this order, $\operatorname{ord}_n(2)$, must divide $n-1$.

Given d is the least integer greater than 2 and less than n that does not divide $n-1$, and $\operatorname{ord}_n(2)$ divides $n-1$, it follows that D cannot equal $\operatorname{ord}_n(2)$. Furthermore, by Lemma 1, we know that if an integer b does not divide an integer a, and $1 < b < \left\lfloor \frac{a}{b} \right\rfloor$, then $\left\lfloor \frac{a}{b} \right\rfloor$ does not divide a. Applying this to our current context with $a = n-1$ and $b = d$: The least odd composite integer such that $2^{n-1} \equiv 1 \pmod{n}$ is 341 [11]. It follows from Lemma 3 that D must be less than or equal to $\lfloor \log_2(n-1) \rfloor + 2$. For $n \geq 341$, clearly $d < \lfloor \sqrt{n} \rfloor$ which satisfies Lemma 2. Thus we have $1 < d < \left\lfloor \frac{n-1}{d} \right\rfloor$ and hence, $\left\lfloor \frac{n-1}{d} \right\rfloor$ cannot divide $n-1$.

Since $\operatorname{ord}_n(2)$ is a divisor of $n-1$ and $\left\lfloor \frac{n-1}{d} \right\rfloor$ is not, it must be that $\left\lfloor \frac{n-1}{d} \right\rfloor$ is not equal to $\operatorname{ord}_n(2)$, as this would imply a contradiction with the nature of $\operatorname{ord}_n(2)$ as a divisor of $n-1$. Therefore, $\left\lfloor \frac{n-1}{d} \right\rfloor \neq \operatorname{ord}_n(2)$. □

Lemma 5 (Upper bound on d)**.** Let n be an odd composite integer > 3. Then there exists an odd prime integer $d \leq (1 + o(1)) \log(n)$ such that $n \not\equiv 0 \pmod{d}$.

Proof. Define $f(n)$ as the least odd prime not dividing n. For primorials $q_m = \prod_{k=1}^{m} p_k$, with $m \geq 2$, $f(q_m)$ is the $(m+1)$-th prime, as q_m is divisible by the first m primes (including 2).

By the Prime Number Theorem, $p_m \approx m \log m$, so for a primorial q_m, $f(q_m) = p_{m+1} \approx (m+1)\log(m+1)$.

Since the function f is maximized at primorials, for a general n, especially when n is large and not necessarily a primorial, we can estimate that $f(n) \leq (1 + o(1)) \log n$. This upper bound is asymptotic and becomes more accurate for larger n.

Therefore, for any odd composite integer n, the least odd prime non-divisor of n is bounded above by $(1 + o(1)) \log n$. □

Remark. The proof for Lemma 5 was adapted from a proof given by a MathOverflow user [12].

Lemma 6. Let $n > 3$ be an odd composite integer such that $2^{n-1} \equiv 1 \pmod{n}$. Let $d > 2$ be the least prime integer such that $n \not\equiv 1 \pmod{d}$. If $2^{\left\lfloor \frac{n-1}{d} \right\rfloor} \not\equiv 1 \pmod{n}$, then n has at least one prime divisor p such that 2 is not a dth power residue modulo p. That is, there are no solutions $b \in \mathbb{Z}_{\text{ı}}$ such that $b^d \equiv 2 \pmod{p}$.

Proof. Let $\delta = \gcd(p-1, d)$. For prime p, Euler's criterion states that $a \not\equiv 0 \pmod{p}$ is a dth power residue modulo p if and only if:

$$a^{\left\lfloor \frac{p-1}{\delta} \right\rfloor} \equiv 1 \pmod{p} \tag{3}$$

Applying to Euler's criterion to our situation, since p and d are both odd primes and $p \neq d$, we have that $a = 2$ is a dth power residue modulo p if and only if $d \mid p-1$ and $2^{\frac{p-1}{d}} \equiv 1 \pmod{p}$.

For the sake of contradiction, let's assume that for all prime divisors p_i of n, $d \mid p_i - 1$ and $2^{\frac{p_i-1}{d}} \equiv 1 \pmod{p}_i$. Then by the Chinese Remainder theorem, we must have $d \mid n-1$. However, this is a contradiction, as we have defined d such that $n \not\equiv 1 \pmod{d}$. Therefore, we deduce that there must exist at least one prime divisor p of n such that $2^{\left\lfloor \frac{p}{d} \right\rfloor} \not\equiv 1 \pmod{p}$. For this p, Euler's criterion does not hold and it follows that 2 is not a dth power residue modulo p. □

3.1 Primes Case

Lemma 7 (Primes pass)**.** Let n be an odd prime integer such that $n > 3$. Denote d as the least integer greater than 2 such that $n \not\equiv 1 \pmod{d}$. Then, $(1+x)^n \equiv 1 + x^n \pmod{n, x^d - 2}$.

Proof. The result follows trivially from the binomial theorem. □

3.2 Composites Case

A fundamental theorem in polynomial ring theory states that an integer n is prime if and only if $(1+x)^n \equiv 1 + x^n \pmod{n} \in \mathbb{Z}[x]$ [10]. This congruence was used as the basis for the AKS test [1]. A short proof of the theorem, given by Granville (2004) [10], is that since $(x+1)^n - (x^n + 1) = \sum_{k=1}^{n-1} \binom{n}{k} x^k$, we may have $(1+x)^n \equiv 1 + x^n \pmod{n}$ if and only if n divides $\binom{n}{k}$ for all k in the range $1 \leq k \leq n-1$. It is important to note that for the polynomial congruence to be valid, x^n and $(1+x)^n$ must be irreducible in $(\mathbb{Z}/n)[x]$.

Kopparty and Wang (2014) [13] proved several theorems related to limits on the counts of consecutive zero coefficients in polynomials over finite fields. We take a similar approach to show that composite integers will always fail our test.

Theorem 2 (Composites case)**.** Let $n = pq$ be an odd composite integer > 3 with p a prime divisor. Let d be the least positive prime integer greater than 2 such that $n \not\equiv 1 \pmod{d}$. Suppose $x^n = 2^{\lfloor \frac{n}{d} \rfloor} \not\equiv 1 \pmod{n}$, and consider the polynomial $f(x) = (1+x)^n - (1+x^n) \in \mathbb{Z}_p[x]$. If n does not have a prime divisor $\leq d$, then $f(x)$ is nonzero when reduced modulo $x^d - 2$.

Proof. We are given composite n with $2^{\lfloor \frac{n}{d} \rfloor} \not\equiv 1 \pmod{n}$. By Lemma 6, there must exist at least one prime divisor p_i of n such that 2 is not a dth power residue modulo p_i. Let p be that prime.

Consider the polynomial ring $\mathbb{Z}_p[x]$. We examine the reduction of $f(x)$ modulo $x^d - 2$, which gives us a polynomial $f(x) \pmod{x^d - 2} \in \mathbb{Z}_p[x]$. After reduction modulo $x^d - 2$, the polynomial $f(x)$ has $\deg(f(x)) = d - 1$, and can be written as $f(x) = \sum_{i=0}^{d-1} c_i x^i$.

Assume for contradiction that $f(x)$ is the zero polynomial in $\mathbb{Z}_p[x]/(x^d - 2)$. This would imply that all coefficients c_i are zero.

Since p is prime, $\mathbb{Z}_p$ is a finite field and $\mathbb{Z}_p[x]$ is a polynomial ring over this field. We aim to establish that $\mathbb{Z}_p[x]/(x^d - 2)$ forms a field. To do so, we must show that $x^d - 2$ is irreducible over $\mathbb{Z}_p[x]$.

We recall a classical theorem from field theory (Irreducibility Theorem):

Suppose $c \in F$ where F is a field, and $0 < d \in \mathbb{Z}$. The polynomial $x^k - c$ is irreducible over F if and only if c is not a qth power in F for any prime q dividing k, and c is not in $-4F^4$ when 4 divides k. [14]

In our case, $F = \mathbb{Z}_p$, $c = 2$, and $k = d$, where d is prime.

Regarding the first criterion, given d is prime, d does not have any divisors q, and hence it suffices to check only d itself. We have chosen p such that 2 is not a dth power residue modulo p, which informs that there is no element $b \in \mathbb{Z}_p$ such that $b^d \equiv 2 \pmod{p}$. The first criterion is satisfied. Since d is prime, it is not divisible by 4, and thus second criterion does not apply.

By the Irreducibility Theorem, $x^d - 2$ is irreducible over $\mathbb{Z}_p[x]$ and hence, the quotient ring $\mathbb{Z}_p[x]/(x^d - 2)$ forms a field.

Applying the Fundamental Theorem of Algebra for finite fields, if $\mathbb{Z}_p[x]/(x^d - 2)$ is a field and $\deg(f(x)) = d - 1$, then $f(x)$ can have at most $d - 1$ roots in $\mathbb{Z}_p[x]/(x^d - 2)$. The assumption that $f(x)$ is zero would imply it has p roots, which is a contradiction unless $p \leq d - 1$. However, this is clearly false, since we are given n which does not have a prime divisor $\leq d$.

Therefore, we conclude that $f(x)$ must be nonzero in $\mathbb{Z}_p[x]/(x^d - 2)$ for at least one prime p which divides n. □

4 Proof of the Main Theorem

Proof of Theorem 1. Let n be an odd integer > 3. If n is prime, then by Lemma 7, the polynomial congruence holds and n will pass the test. Conversely, if n is composite, then by Lemma 2 the polynomial congruence does not hold and n will fail the test. Hence, the theorem is proven. □

5 Algorithm

INPUT: An integer $n > 1$.

1. If $n \equiv 0 \mod 2$:
   (a) If n equals 2, output PRIME.
   (b) Otherwise, output COMPOSITE.
2. If n equals 3, output PRIME.
3. If $2^{n-1} \bmod n$ does not equal 1, output COMPOSITE.
4. Find the least prime integer d that is greater than 2 such that $n \not\equiv 1 \pmod{d}$.
5. If n has a prime divisor less than d, output COMPOSITE.
6. Compute the polynomial expansion of $x^n \bmod n$ in the ring $(\mathbb{Z}/n)[x]/(x^d - 2)$ with degree d, and store the result.
7. Compute the polynomial expansion of $(1 + x)^n \bmod n$ in the ring $(\mathbb{Z}/n)[x]/(x^d - 2)$ with degree d, and store the result.
8. If $(1 + x)^n \neq 1 + x^n$, output COMPOSITE.
9. Output PRIME;

5.1 Time Complexity Analysis

5.1.1 Algorithm Overview

The given algorithm is a primality test that involves several computational steps, including modular arithmetic and polynomial exponentiation in the ring $(\mathbb{Z}/n)[x]/(x^d - 2)$.

5.1.2 Analysis of Individual Operations

1. **Check for Even** n:
   This step involves calculating $n \bmod 2$ and has a time complexity of $O(1)$.

2. **Finding** d:

   Finding the least integer $d > 2$ such that $n \not\equiv 1 \pmod{d}$ takes at most $O((1 + o(1)) \log(n))$ steps, with each step requiring $O(1)$ time for the mod operation. Hence, the overall complexity is $O((1 + o(1)) \log(n))$.

3. **Computing** $x^n \bmod n \in (\mathbb{Z}/n)[x]/(x^d - 2)$:

   Computing the polynomial expansion of $x^n \pmod{n}$ in the ring $(\mathbb{Z}/n)[x]/(x^d - 2)$ with degree d is equivalent to calculating $2^{\lfloor \frac{n}{d} \rfloor} \pmod{n}$. This step requires modular exponentiation with a $\log(n)$-digit base and a $\log(n)$-digit exponent. The time complexity of modular exponentiation is $O(\log(n)M(n))$.

4. **Computing** $(1 + x)^n \bmod n \in (\mathbb{Z}/n)[x]/(x^d - 2)$:

   Computing the polynomial expansion of $(1 + x)^n \pmod{n}$ in the ring $(\mathbb{Z}/n)[x]/(x^d - 2)$ with degree d involves exponentiating a polynomial in $(\mathbb{Z}/n)[x]/(x^d - 2)$ with $D = O(\log(n))$ terms. Exponentiation using repeated squaring takes $O(\log(n))$ steps, and each step requires $O(M(n) \log(n))$ time due to the multiplication of polynomials of size $O(\log(n))$. Therefore, the overall complexity is $O(\log^2(n)m(n))$.

5. **Checking the equality** $(1 + x)^n = 1 + x^n \pmod{n} \in (\mathbb{Z}/n)[x]/(x^d - 2)$: The final steps involve comparing the equality of coefficients in the polynomials $(1 + x)^n$ and $1 + x^n$. This requires $O(\log(n))$ comparisons, which are themselves $O(1)$ operations.

5.1.3 Overall Time Complexity

The dominant time complexity in the algorithm comes from computing $(1+x)^n \pmod{n} \in (\mathbb{Z}/n)[x]/(x^d-2)$. Therefore, the overall time complexity of the algorithm is $T(n) = O(\log^2(n)M(n))$.

Harvey and van Der Heoven (2021) have given an algorithm for integer multiplication which has a time complexity $M(n) = O(\log(n) \log\log(n))$ [15]. This would give our algorithm an overall time complexity of:

$$T(n) = O(\log^2(n)M(n)) = O(\log^2(n) \log(n) \log\log(n)) = O(\log^3(n) \log\log(n)) = \tilde{O}(\log^3(n)) \tag{4}$$

5.1.4 Conclusion

The overall complexity is polynomial in the size of n when expressed in terms of bit operations, making the algorithm efficient for large values of n.

6 Implementation Details

Sample open source .NET and Python implementations, along with test data, are available on the author's Github page [16].

7 Acknowledgements

I am indebted to Professor Seth Pettie for his invaluable insights and advice on both the content and presentation of this paper. His expertise in academic writing and research methodologies has been incredibly helpful, and I am particularly grateful for his mentorship and constructive criticism that significantly shaped this work. I also extend my gratitude to his graduate students for their input and earnest interest in my research.

I owe a great deal of inspiration to the proofs by Kopparty and Wang in their paper on polynomial roots and coefficients over finite fields [13]. Their ingenious approach was instrumental in the development of the key proof in my paper. I also extend my heartfelt gratitude to Professor Kopparty, for his encouraging and constructive feedback, which came at a challenging time when I was ready to give up.

I extend my sincere gratitude to Professor Oded Goldreich for his guidance and support throughout the publication process at the ECCC. His patience and motivational guidance, particularly during moments of uncertainty, were invaluable. His expertise in navigating the complex landscape of academic publishing has been indispensable.

A special thanks to the community at `mersenneforum.org`, particularly to users Charybdis, R. D. Silverman, and Dr. Sardonicus, for their insightful feedback and scrutiny of my early proof attempts. Their contributions were fundamental in refining my approach.

Disclaimer: The views and any errors in this paper are solely my own and do not necessarily reflect those of the individuals acknowledged herein.

References

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in p. *Annals of Mathematics*, pages 781–793, 2002.

[2] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[3] Hendrik W Lenstra and Carl Pomerance. Primality testing with gaussian periods. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 369(1951): 3376–3390, 2011.

[4] Hendrik W Lenstra. Factoring integers with elliptic curves. *Annals of mathematics*, pages 649–673, 1987.

[5] Robert Baillie and Samuel S Jr Wagstaff. Lucas pseudoprimes. *Mathematics of Computation*, 35(152): 1391–1417, 1980.

[6] Michael O Rabin. Probabilistic algorithm for testing primality. *Journal of Number Theory*, 12(1): 128–138, 1980.

[7] Gary L Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13(3):300–317, 1976.

[8] Samuel S Jr Wagstaff. Pseudoprimes and a generalization of artin's conjecture. *Acta Arithmetica*, 41 (2):141–150, 1983.

[9] Carl Pomerance. The use of elliptic curves in cryptography. *Advances in Cryptology*, pages 203–208, 1984.

[10] Andrew Granville. It is easy to determine whether a given integer is prime. *Bulletin of the American Mathematical Society*, 42:3–38, 2004. URL `https://www.ams.org/journals/bull/2005-42-01/S0273-0979-04-01037-7`.

[11] N. J. A. Sloane. Entry a001567 in the on-line encyclopedia of integer sequences. `https://oeis.org/A001567`, 2023. Fermat pseudoprimes to base 2.

[12] 2734364041 (https://mathoverflow.net/users/111215/2734364041). Least number coprime to a given integer. MathOverflow, 2021. URL `https://mathoverflow.net/q/409792`. URL:https://mathoverflow.net/q/409792 (version: 2021-12-01).

[13] Swastik Kopparty and Qiang Wang. Roots and coefficients of polynomials over finite fields. *Finite Fields and Their Applications*, 29:198–201, 2014. ISSN 1071-5797. doi: https://doi.org/10.1016/j.ffa.2014.04.002. URL `https://www.sciencedirect.com/science/article/pii/S1071579714000574`.

[14] Gregory Karpilovsky. *Topics in Field Theory*. North-Holland Mathematics Studies. North-Holland, 1989. ISBN 9780444705207.

[15] Joris van Der Hoeven David Harvey. Integer multiplication in time o(n log n). *Annals of Mathematics*, 2021. doi: 10.4007/annals.2021.193.2.4.hal-02070778v2.

[16] Joseph M. Shunia. A sample .net implementation of the primality test. `https://github.com/jshunia/Shunia.Primes`, 2023. Accessed: December 2023.