An Efficient Deterministic Primality Test

Joseph M. Shunia

December 2023

Abstract

A deterministic primality test with a polynomial time complexity of $\tilde{O}(\log^3(n))$ is presented. Central to our test is a novel polynomial reduction ring, applied to efficiently validate the polynomial congruence $(1+x)^n \equiv 1 + x^n \pmod{n}$, a condition that holds exclusively for prime n. The test is grounded in our main theorem, supported by a series of lemmas, which collectively show how the congruence is verified through polynomial expansion within our reduction ring.

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 main theorem posits a condition for an odd integer n to be prime, based on specific modular congruences related to the binomial transforms of powers of 2. The basis for our test is the following main theorem: Let n be an odd integer greater than 3. Denote D as the least integer strictly greater than 2 and less than n such that $n \not\equiv 1 \pmod{D}$. Then, n is prime if and only if a set of simultaneous modular congruences involving D, n, and powers of 2 hold. We show how these congruences are equivalent to checking the polynomial congruence $(1+x)^n \equiv 1 + x^n \pmod{n}$, which is known to hold for only prime integers n [10]. To efficiently calculate our congruences, we construct a specialized polynomial ring which limits the polynomial degree to $O(\log(n))$.

1.1 Structure of the Paper

This paper is structured as follows: We begin by presenting the main theorem which defines our primality test. The computation of certain congruences required for our test is infeasible using existing methods; to address this, we introduce an innovative approach utilizing a specialized polynomial reduction ring, which provides efficient calculations. We follow up with the proof of our main theorem, substantiated by supporting identities and lemmas. 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 pseudocode for our test, and a link to an open source implementation, to demonstrate how our test can be implemented.

2 Statement of Main Theorem

Theorem 1 (Main theorem)**.** Let n be an odd integer > 3. Denote D as the least integer greater than 2 such that $n \not\equiv 1 \pmod{D}$. If the following condition holds for all $0 \leq j < D$, then n is prime:

$$\sum_{k=0}^{n} \binom{n}{Dk+j} 2^k \equiv \begin{cases} 1 & \text{if } j = 0, \\ 0 & \text{if } 0 < j < D \text{ and } j \neq n \bmod D, \\ 2^{\lfloor \frac{n}{D} \rfloor} & \text{if } j = n \bmod D \end{cases} \pmod{n} \tag{1}$$

3 Preliminaries and Definitions

3.1 Polynomial Reduction Ring

In this section, we define a special polynomial ring R, with a modular variation M, and show how M can be used to efficiently calculate the congruences required by Theorem 1. The standard approach to calculating the congruences requires evaluating $\binom{n}{Dk+j}2^k \pmod{n}$ for each k, j and summing the results, which takes time exponential in n. Conversely, our approach offers an efficient polynomial time complexity of $\tilde{O}(D \log^2(n))$.

In a preliminary paper relating the central binomial coefficients and Gould's sequence to polynomial rings [11], we showed how a multivariate polynomial ring, analogous to the univariate polynomial ring we have defined herein, can be used generally to compute the binomial transforms of recursive integer sequences. We apply the same technique here.

Definition 1 (Polynomial reduction ring)**.** We construct a polynomial ring $R = \mathbb{Z}[x]$ in which addition is carried out as usual, and multiplication is followed by polynomial degree reduction via a special substitution function, denoted $\mathbf{r}$ [11]. The multiplication operation $*$ is defined as follows:

$$P(x) * Q(x) = \mathbf{r}(P(x) \cdot Q(x)) \text{ for all } P(x), Q(x) \in R, \tag{2}$$

$\mathbf{r}$ is an operation that enforces any defined substitution rules for the polynomial's terms upon multiplication. In our case, for the generator x, we have:

$$x * x^{d-1} = x^d = \mathbf{r}(x^d) = N, \text{ where } N \in R \tag{3}$$

For any polynomial in our ring R, the function $\mathbf{r}$ is distributed to the individual terms and implicitly applied to reduce terms. Whenever the term x^d appears, it is replaced with its defined mapping. The mappings in $\mathbf{r}$ are extended to terms of higher degrees, meaning $\mathbf{r}(x^{d+k}) = \mathbf{r}(x^d * x^k) = Nx^k$, for $k > 1$. $\mathbf{r}$ is applied

recursively to the polynomial until and its terms until the degree of the polynomial is less than d. For terms which do not match a defined substitution rule, $\mathbf{r}$ returns them as they are.

Example 1. We take the ring R as defined in Definition 1 with $N = 2$, $d = 3$. Hence, $\mathbf{r}(x^3) = 2$. $P(x) := 1 + x + 3x^3 + x^4 + x^6, P(x) \in R$.

$$\begin{aligned}
\mathbf{r}(P(x)) &= \mathbf{r}(1 + x + 3x^3 + x^4 + x^6) && (4)\\
&= \mathbf{r}(1) + \mathbf{r}(x) + \mathbf{r}(3x^3) + \mathbf{r}(x^4) + \mathbf{r}(x^6) && (5)\\
&= 1 + x + \mathbf{r}(2 \cdot 3) + \mathbf{r}(2 \cdot x) + \mathbf{r}(2 \cdot x^3) && (6)\\
&= 1 + x + 6 + 2x + \mathbf{r}(2 \cdot 2) && (7)\\
&= 7 + 3x + 4 && (8)\\
&= 11 + 3x && (9)
\end{aligned}$$

3.2 Modular Polynomial Reduction Ring

Definition 2 (Modular polynomial ring M)**.** Let $R = \mathbb{Z}[x]$ be our polynomial ring as defined in Definition 1, where the multiplication is modified by a substitution function $\mathbf{r}$ as described previously. Let n be a positive integer. We define the modular variation of R, denoted as M, to be the ring of polynomials with coefficients in $\mathbb{Z}/n\mathbb{Z}$ and with multiplication modified by a corresponding substitution function $\mathbf{r}_M$. Formally, $M = (\mathbb{Z}/n\mathbb{Z})[x]$, where the coefficients of the polynomials in M are taken modulo n, and the multiplication in M is given by

$$P(x) * Q(x) = \mathbf{r}_M(P(x) \cdot Q(x)) \text{ for all } P(x), Q(x) \in M, \tag{10}$$

$\mathbf{r}_M$ is defined correspondent to $\mathbf{r}$ but operates within the context of the coefficients being in $\mathbb{Z}/n\mathbb{Z}$.

3.3 Polynomial Reduction Ring Axioms

We assert that the ring R with the modified operation $*$ still satisfies the standard ring axioms. Specifically, we show that $(R, +, *)$ is associative, commutative (with respect to addition), and has an additive identity and an additive inverse for every element. Additionally, the distributive property of multiplication over addition is preserved under the operation $*$.

3.3.1 Multiplicative Properties

Proposition 1 (Distributivity of $*$)**.** The operation $*$ is distributive over addition in the ring R.

Proof. The modified multiplication $*$ is distributive over addition because for any $P(x), Q(x), S(x) \in R$,

$$\begin{aligned}
P(x) * (Q(x) + S(x)) &= \mathbf{r}(P(x) \cdot (Q(x) + S(x))) && (11)\\
&= \mathbf{r}(P(x) \cdot Q(x) + P(x) \cdot S(x)) && (12)\\
&= \mathbf{r}(P(x) \cdot Q(x)) + \mathbf{r}(P(x) \cdot S(x)) && (13)\\
&= P(x) * Q(x) + P(x) * S(x) && (14)
\end{aligned}$$

Where the second equality uses the distributive property of the standard multiplication in $\mathbb{Z}[x]$ and the linearity of $\mathbf{r}$ with respect to polynomial addition. □

Proposition 2 (Associativity of $*$)**.** The multiplication operation $*$ in the ring R is associative.

Proof. To prove associativity, we need to show that for any $P(x), Q(x), S(x) \in R$:

$$(P(x) * Q(x)) * S(x) = P(x) * (Q(x) * S(x)) \tag{15}$$

Expanding the left-hand side:

$$\begin{aligned}(P(x) * Q(x)) * S(x) &= \mathbf{r}(P(x) \cdot Q(x)) * S(x) &(16)\\ &= \mathbf{r}(\mathbf{r}(P(x) \cdot Q(x)) \cdot S(x)) &(17)\end{aligned}$$

Similarly, for the right-hand side:

$$\begin{aligned}P(x) * (Q(x) * S(x)) &= P(x) * \mathbf{r}(Q(x) \cdot S(x)) &(18)\\ &= \mathbf{r}(P(x) \cdot \mathbf{r}(Q(x) \cdot S(x))) &(19)\end{aligned}$$

Since the standard multiplication in $\mathbb{Z}[x]$ is associative and $\mathbf{r}$ is a well-defined operation that respects this associativity, we have:

$$\mathbf{r}(\mathbf{r}(P(x) \cdot Q(x)) \cdot S(x)) = \mathbf{r}(P(x) \cdot \mathbf{r}(Q(x) \cdot S(x))) \tag{20}$$

Which shows that:

$$(P(x) * Q(x)) * S(x) = P(x) * (Q(x) * S(x)) \tag{21}$$

□

3.3.2 Additive Properties

Proposition 3 (Preservation of additive properties)**.** The commutative and associative properties of addition, and the existence of an additive identity and inverses, are maintained in the ring R.

Proof. The additive structure of R remains unchanged. Thus, the commutative and associative properties of addition, and the existence of an additive identity and inverses, are inherited directly from $\mathbb{Z}[x]$. □

Proposition 4. The ring $M = (\mathbb{Z}/n\mathbb{Z})[x]$ with the modified multiplication operation $*$, as defined by the substitution function $\mathbf{r}_M$, inherits the standard ring axioms from $R = \mathbb{Z}[x]$.

Proof. Since the ring M is structurally identical to R with the only difference being the coefficient domain ($\mathbb{Z}/n\mathbb{Z}$ instead of $\mathbb{Z}$), and the modified multiplication operation $*$ in M is defined similarly to R using $\mathbf{r}_M$, analogous to $\mathbf{r}$, M inherits the ring properties of R. This includes the associativity and commutativity of addition, the existence of an additive identity and inverses, the distributivity of multiplication over addition, and the associativity of multiplication. These properties are preserved under the transition from $\mathbb{Z}$ to $\mathbb{Z}/n\mathbb{Z}$ coefficients and the analogous definition of $*$ in M. □

4 Supporting Identities

Identity 1 (Powers of two)**.** Let $n, D \in \mathbb{Z}^+$, with $1 \leq D < n$:

$$2^n = \sum_{j=0}^{D-1} \sum_{k=0}^{n} \binom{n}{Dk + j} \tag{22}$$

Proof. By the binomial theorem, $2^n = \sum_{k=0}^{n} \binom{n}{k}$. We can split this sum up into D groups of at most $\lfloor \frac{n}{D} \rfloor$ terms, like so:

$$2^n = \sum_{k=0}^{n} \binom{n}{k} = \sum_{j=0}^{D-1} \sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk+j} \tag{23}$$

For $k > \lfloor \frac{n}{D} \rfloor$, we have $Dk + j > n$. In such case, $\binom{n}{Dk+j} = 0$. Hence, we can say:

$$\sum_{j=0}^{D-1} \sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk+j} = \sum_{j=0}^{D-1} \sum_{k=0}^{n} \binom{n}{Dk+j} \tag{24}$$

□

Identity 2 (Binomial transform of floored powers of two)**.** Let $n, D \in \mathbb{Z}^+$, with $1 \leq D < n$:

$$\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor} = \sum_{j=0}^{D-1} \sum_{k=0}^{n} \binom{n}{Dk+j} 2^k \tag{25}$$

Proof. By Identity 1, we have:

$$\sum_{k=0}^{n} \binom{n}{k} = \sum_{j=0}^{D-1} \sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk+j} \tag{26}$$

Applying this to $\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor}$, since the exponent in $2^{\lfloor \frac{k}{D} \rfloor}$ is partitioned by D, it splits over the groups. We may rewrite our equation as:

$$\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor} = \sum_{j=0}^{D-1} \sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk+j} 2^k \tag{27}$$

As in Identity 1, for $k > \lfloor \frac{n}{D} \rfloor$, we have $Dk + j > n$. In such case, $\binom{n}{Dk+j} = 0$. Hence, we can say:

$$\sum_{j=0}^{D-1} \sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk+j} 2^k = \sum_{j=0}^{D-1} \sum_{k=0}^{n} \binom{n}{Dk+j} 2^k \tag{28}$$

□

5 Polynomial Reduction Ring Theorems

Theorem 2. Let $n, D \in \mathbb{Z}^+$ with $D > 2$. Define the ring M as in Definition 2 with $d = D-1$, $N = 2$, hence $\mathbf{r}(x^{D-1}) = 2$. Let x^k be a polynomial in M. Taking the sum of the coefficients modulo n in the polynomial expansion of x^k, gives $2^{\lfloor \frac{k}{D} \rfloor} \pmod{n}$.

Proof. Examining $D = 2$, we can see:

$$\begin{aligned}
x^0 &= 1 = 2^{\lfloor 0/2 \rfloor} \\
x^1 &= 1 = 2^{\lfloor 1/2 \rfloor} \\
x^2 &= 2 = 2^{\lfloor 2/2 \rfloor} \\
x^3 &= 2x = 2 = 2^{\lfloor 3/2 \rfloor} \\
x^4 &= 2x^2 = 4 = 2^{\lfloor 4/2 \rfloor} \\
x^5 &= 4x = 4 = 2^{\lfloor 5/2 \rfloor} \\
x^6 &= 4x^2 = 8 = 2^{\lfloor 6/2 \rfloor} \\
&\vdots
\end{aligned}$$

We proceed by induction. For $k = 0$, we have $x^0 = 1$, and the sum of coefficients is $2^{\lfloor \frac{0-1}{D} \rfloor} = 2^0 = 1$. Assume the theorem holds for some $k \geq 0$, i.e., the sum of coefficients modulo n in x^k is $2^{\lfloor \frac{k}{D} \rfloor} \pmod n$. We need to show it also holds for $k + 1$. Consider $x^{k+1} = x^k * x$. By the definition of $*$ and $\mathbf{r}$, the degree of x^{k+1} gets reduced by $\mathbf{r}$ every time it reaches D. The number of times this reduction happens is $\lfloor \frac{k}{D} \rfloor$. Therefore, the sum of coefficients in x^{k+1} should be $2^{\lfloor \frac{k+1}{D} \rfloor}$, as each reduction by $\mathbf{r}$ doubles the sum of coefficients. Hence, the theorem holds for $k + 1$.

By the principle of mathematical induction, the theorem is proven. □

Theorem 3. Let n an integer > 0. Let D be an integer > 2. Define the ring M as in Definition 2 with $d = D-1$, $N = 2$, hence $\mathbf{r}(x^{D-1}) = 2$. Let $(1+x)^n$ be a polynomial in M. Taking the sum of the coefficients modulo n in the polynomial expansion of $(1 + x)^n$ is equivalent to $\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor} \pmod n$.

Proof. By the binomial theorem, we have:

$$(1 + x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k \tag{29}$$

Hence, when we evaluate this polynomial at $x = 1$, we sum the coefficients of the polynomial to get:

$$\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor} \quad \text{(by Theorem 2)} \tag{30}$$

□

6 Supporting Lemmas

Lemma 1. Let $n, D \in \mathbb{Z}^+$, with $1 \leq D < n$. Let M be our modular polynomial reduction ring as defined in Definition 2 with $\mathbf{r}(x^{D-1}) = 2$. For $(1 + x)^n \in M$, we have:

$$[x^j](1 + x)^n \equiv \sum_{k=0}^{n} \binom{n}{Dk + j} 2^k \pmod n \tag{31}$$

Proof. By Identity 2, we have $\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor} = \sum_{j=0}^{D-1} \sum_{k=0}^{n} \binom{n}{Dk+j} 2^k$. In Theorem 3, we showed that evaluating $(1 + x)^n \in M$ at $x = 1$ yields $\sum_{k=0}^{n} \binom{n}{k} 2^{\lfloor \frac{k}{D} \rfloor}$. Upon expansion $(1 + x)^n$ will contain at most D

terms. By the nature of exponentiation within our ring, these groups are spaced uniformly over the binomial coefficients. That is, the coefficient of the term x^j in the expanded polynomial corresponds to the sum $\sum_{k=0}^{n} \binom{n}{Dk+j} 2^k \pmod{n}$. □

Lemma 2 (Upper bound on D)**.** Let n be an integer such that $n > 3$, then there exists an integer D such that $1 < D \leq \lfloor \log_2(n-1) \rfloor + 2$ and $n \not\equiv 1 \pmod{D}$.

Proof. In order to have $n \equiv 1 \pmod{D}$ for some integer D, there must exist an integer k such that $D \cdot k + 1 \equiv 0 \pmod{n}$. This implies that any such D will divide $n-1$. Given n is a composite integer > 3, clearly $n-1$ can have at most $\lfloor \log_2(n-1) \rfloor$ prime factors (if $n-1$ 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 $n-1$. This implies $n - 1 \not\equiv 0 \pmod{D}$ and hence it follows, $n \not\equiv 1 \pmod{D}$. □

Lemma 3 (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, the conditions of Theorem 1 hold unconditionally and n will pass the test.

Proof. For this proof, we will examine all of the cases individually and show that they are satisfied for prime n.

Case 1: $j = 0$
If $j = 0$, then all $\binom{n}{Dk} \equiv 0 \pmod{n}$ except for $k = 0$. Hence, we have:

$$\sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk} 2^k \equiv \binom{n}{0} 2^0 \equiv 1 \cdot 1 \equiv 1 \pmod{n} \tag{32}$$

Case 2: $0 < j < D$ and $j \neq n \bmod D$
If $0 < j < D$ and $j \neq n \bmod D$, all $\binom{n}{Dk+j} \equiv 0 \pmod{n}$. Hence, the entire sum $\sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk+j} 2^k$ must be equivalent to $0 \pmod{n}$.

Case 3: $j = n \bmod D$
If $j = n \bmod D$, then all $\binom{n}{Dk} \equiv 0 \pmod{n}$ except for $k = \lfloor \frac{n}{D} \rfloor$. Hence, we have:

$$\sum_{k=0}^{\lfloor \frac{n}{D} \rfloor} \binom{n}{Dk + (n \bmod D)} 2^k \equiv \binom{n}{n} 2^{\lfloor \frac{n}{D} \rfloor} \equiv 1 \cdot 2^{\lfloor \frac{n}{D} \rfloor} \equiv 2^{\lfloor \frac{n}{D} \rfloor} \pmod{n} \tag{33}$$

Conclusion: All of the individual conditions are satisfied when $n > 3$ is an odd prime. Hence, n will pass the test unconditionally. □

Lemma 4 (Composites fail)**.** Let n be an odd composite integer such that $n > 3$. Denote D as the least integer greater than 2 such that $n \not\equiv 1 \pmod{D}$. Then, the conditions of Theorem 1 cannot hold and n will fail the test unconditionally.

Proof. 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 $1 \leq k \leq n-1$.

Remark. It is important to note that for the polynomial congruence to be valid, x^n and $(1+x)^n$ must be irreducible in $(\mathbb{Z}/n\mathbb{Z})[x]$. In this context, "irreducible" simply means that the polynomial x^n does not collapse to an integer modulo n; the property of irreducibility is unrelated to the concept of degree reduction used by our polynomial rings.

Now, we define M to be the modular polynomial ring as in Definition 2. Recall that $M \in (\mathbb{Z}/n\mathbb{Z})[x]$ and that the standard ring axioms continue to hold in M. Let $\mathbf{r}(x^{D-1}) = 2$. This implies that $x^{D-1} \equiv 2 \pmod{n} \in M$.

As stated in the remark above, in order to use the polynomial congruence $(1+x)^n \equiv 1 + x^n \pmod{n}$ to test the primality of n, we need to ensure that x^n and $(1+x)^n$ are irreducible in M. Fortunately, so long as $n \not\equiv 1 \pmod{D}$, these polynomials are irreducible. Conversely, if $n \equiv 1 \pmod{D}$, the polynomial ring structure collapses from $(\mathbb{Z}/n\mathbb{Z})[x]$ to $\mathbb{Z}/n\mathbb{Z}$ when expanding $(1+x)^n$. This "collapse" of the polynomial ring structure to the integers is caused by the value of x^n being located at the coefficient $[x^1]$ in the expanded polynomials x^n and $(1+x)^n$. In such situation, when we calculate the binomial transform by expanding $(1+x)^n \in M$, the polynomial will expanded over degree 0, and hence, the binomial transform is calculated over the integers. In all other cases, that is, when $n \not\equiv 1 \pmod{D}$, the integrity of the polynomial ring structure is maintained since the binomial transform is taken over a polynomial degree that is greater than 0, and thus the result is a polynomial.

Therefore, so long as $n \not\equiv 1 \pmod{D}$, it is valid to check only the following polynomial congruence to determine the primality of n:

$$(1+x)^n \equiv 1 + x^n \pmod{n} \ \in M \tag{34}$$

In the present case, n is an odd composite integer, and by definition, we have $n \not\equiv 1 \pmod{D}$. Hence, and we may conclude that x^n and $(1+x)^n$ are irreducible in M.

It follows from Lemma 1 that verifying the polynomial congruence $(1+x)^n \equiv 1 + x^n \pmod{n}$ is equivalent to checking the defining congruences of our test:

$$\sum_{k=0}^{n} \binom{n}{Dk+j} 2^k \equiv \begin{cases} 1 & \text{if } j = 0, \\ 0 & \text{if } 0 < j < D \text{ and } j \neq n \bmod D, \\ 2^{\lfloor \frac{n}{D} \rfloor} & \text{if } j = n \bmod D \end{cases} \pmod{n}$$

Finally, since checking the above congruences is equivalent to checking $(1+x)^n \equiv 1+x^n \pmod{n}$, composite n will fail the test unconditionally. □

7 Proof of the Main Theorem

Proof of Theorem 1. Let n be an odd integer > 3. If n is prime, then by Lemma 3, the congruences for our test hold and n will pass the test. Conversely, if n is composite, then by Lemma 4 the congruences for our test cannot hold and n will fail the test. Hence, the theorem is proven. □

8 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. Find the least integer D that is greater than 2 such that $n \not\equiv 1 \pmod{D}$.

4. Compute the polynomial expansion of $x^n \bmod n$ in the ring M with degree D, and store the result (See: Definition 2).
5. Compute the polynomial expansion of $(1+x)^n \bmod n$ in the ring M with degree D, and store the result.
6. If $(1+x)^n \neq 1+x^n$, output COMPOSITE.
7. Output PRIME;

8.1 Time Complexity Analysis

8.1.1 Algorithm Overview

The given algorithm is a primality test that involves several computational steps, including modular arithmetic and polynomial exponentiation in the ring M. To avoid confusion with our polynomial ring M, we will denote the time complexity of multiplication as the lower-cased $m(n)$.

8.1.2 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 $O(\log(n))$ steps, with each step requiring $O(1)$ time for the mod operation. Hence, the overall complexity is $O(\log(n))$.

3. **Computing** $x^n \bmod n \in M$:

   Computing the polynomial expansion of $x^n \pmod{n}$ in the ring M 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** $(1+x)^n \bmod n \in M$:

   Computing the polynomial expansion of $(1+x)^n \pmod{n}$ in the ring M with degree D involves exponentiating a polynomial in M 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** $(1+x)^n = 1+x^n \pmod{n} \in M$: 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.

8.1.3 Overall Time Complexity

The dominant time complexity in the algorithm comes from computing $(1+x)^n \pmod{n} \in M$. 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 $m(n) = O(\log(n)\log\log(n))$ [12]. This would give our algorithm an overall time complexity of:

$$T(n) = O(\log^2(n)m(n)) = O(\log^2(n)\log(n)\log\log(n)) = O(\log^3(n)\log\log(n)) = \tilde{O}(\log^3(n)) \tag{35}$$

8.1.4 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.

9 Implementation Details

9.1 Reference Implementation

Sample open source .NET and Python implementations, along with test data, are available on the author's Github page [13].

9.2 Pseudocode Implementation

To demonstrate how our test may be implemented, we offer a pseudocode implementation of the key functions involved.

9.2.1 IsPrime Function

Require: An integer $n > 1$
  function IsPrime(n)
    if $n \bmod 2 = 0$ **then**
      if $n = 2$ **then**
        return true ▷ n is prime
      else
        return false ▷ n is composite
      end if
    end if
    if $n = 3$ **then**
      return true ▷ n is prime
    end if
    $D \leftarrow 2$
    $log \leftarrow \text{Log2}(n)$
    for $i \leftarrow 3$ **to** $\text{Max}(log + 2, 3)$ **do** ▷ Find D, the least non-divisor of $n - 1$
      $D \leftarrow i$
      $m \leftarrow (n - 1) \bmod D$
      if $m \neq 0$ **then**
        break
      end if
    end for
    $v_x \leftarrow \text{Pow}(2, \lfloor n/D \rfloor, n)$
    $polyDegree \leftarrow D - 1$ ▷ Subtract 1 to account for zero indexing in polynomials
    $poly \leftarrow \text{PolyPow}([1, 1], n, n, polyDegree, [2])$
    if $poly[0] \neq 1$ **then**
      return false ▷ n is composite
    end if
    $x_{\text{index}} \leftarrow n \bmod D$
    for $i \leftarrow 1$ **to** D **do** ▷ Check polynomial equality
      if $i = x_{\text{index}}$ **then**

```
            if poly[i] ≠ v_x then
                return false                                   ▷ n is composite
            end if
        else
            if poly[i] ≠ 0 then
                return false                                   ▷ n is composite
            end if
        end if
    end for
    return true                                                ▷ n is prime
end function
```

9.2.2 PolyPow Function

```
function PolyPow(poly_A, k, n, d, polyMapping)
    polyB ← [1]                          ▷ Initialize polyB as a polynomial with constant term 1
    while k > 0 do
        if k mod 2 = 1 then
            polyB ← PolyMul(polyA, polyB, n)                   ▷ Multiply polynomials modulo n
            polyB ← PolyReduce(polyB, d, polyMapping, n)       ▷ Reduce degree of polyB
            if k = 1 then
                break
            end if
        end if
        polyA ← PolyMul(polyA, polyA, n)                       ▷ Square polyA modulo n
        polyA ← PolyReduce(polyA, d, polyMapping, n)           ▷ Reduce degree of polyA
        k ← k/2
    end while
    return polyB
end function
```

9.2.3 PolyReduce Function

```
function PolyReduce(polyA, d, mappingPoly, n)
    if Length(polyA) ≤ d then
        return polyA
    end if
    polyB ← Clone(polyA)                                       ▷ Copy the polynomial to a new array
    polyDegree ← Degree(mappingPoly)                           ▷ Find degree of the mapping polynomial
    degreeDelta ← d − polyDegree
    for i ← Length(polyB) − 1 downto d + 1 do
        if polyB[i] = 0 then
            continue
        end if
        for j ← i − 1 − degreeDelta, k ← polyDegree downto 0 do
            polyB[j] ← polyB[j] + polyB[i] × mappingPoly[k]    ▷ Degree reduction step
        end for
        polyB[i] ← 0
    end for
    polyC ← new array of size d + 1
    for i ← 0 to min(Length(polyC), Length(polyB)) − 1 do      ▷ Take coefficients modulo n
```

```
        polyC[i] ← polyB[i]
        if n ≠ 0 then
            polyC[i] ← polyC[i] mod n
        end if
    end for
    return polyC
end function
```

10 Acknowledgements

The author would like to thank the kind users at `mersenneforum.org` for their helpful feedback.

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] Joseph M. Shunia. A polynomial ring connecting central binomial coefficients and gould's sequence, 2023. URL `https://arxiv.org/abs/2312.00302`.

[12] 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.

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