Primality Tests and PRIMES in P

Nicholas Mosier

Middlebury College

December 8, 2019

Overview

Primality Testing

Definition

A **primality test** is an algorithm that takes in a positive integer n as input and returns whether n is prime or composite.

Primality Testing

Definition

A **primality test** is an algorithm that takes in a positive integer n as input and returns whether n is prime or composite.

Example

Algorithm Check the primality of a positive integer $n \geq 2$.

Input: $n \in \mathbb{N}$ with $n \geq 2$.

Output: Output `PRIME` if n is prime; otherwise output `COMPOSITE`.

```
for all 1 < k < n do
    if k | n then
        return COMPOSITE.
return PRIME.
```

Fermat's Little Theorem (FlT)

Theorem (Fermat's Little Theorem)

Let p be prime and $a \in \mathbb{N}$ with $p \nmid a$. Then

$$a^{p-1} \equiv 1 \pmod{p}.$$

Fermat's Little Theorem (FlT)

Theorem (Fermat's Little Theorem)

Let p be prime and $a \in \mathbb{N}$ with $p \nmid a$. Then

$$a^{p-1} \equiv 1 \pmod{p}.$$

Example

When $a = 2$, $p = 7$:

$$2^{7-1} = 64 = 7 \cdot 9 + 1 \equiv 1 \pmod{7}.$$

Fermat's Little Theorem (FIT)

Theorem (Fermat's Little Theorem)

Let p be prime and $a \in \mathbb{N}$ with $p \nmid a$. Then

$$a^{p-1} \equiv 1 \pmod{p}.$$

Example

When $a = 2$, $p = 7$:

$$2^{7-1} = 64 = 7 \cdot 9 + 1 \equiv 1 \pmod{7}.$$

But does Fermat's Little Theorem describe primality test?
That is, is the converse true?

Converse of Fermat's Little Theorem (First Attempt)

Conjecture

Let $n, a \in \mathbb{N}$ *with* $n \nmid a$. *If*

$$a^{n-1} \equiv 1 \pmod{n},$$

then n *is prime.*

Converse of Fermat's Little Theorem (First Attempt)

Conjecture

Let $n, a \in \mathbb{N}$ with $n \nmid a$. If

$$a^{n-1} \equiv 1 \pmod{n},$$

then n is prime.

Counterexample

When $a = 5$ and $p = 4$,

$$5^{4-1} = 125 = 31 \cdot 4 + 1 \equiv 1 \pmod{4}.$$

However, $p = 4$ is composite.

Converse of Fermat's Little Theorem (Second Attempt)

Conjecture

Let $n \in \mathbb{N}$*. If for all* $b \in \mathbb{N}$ *such that* $n \nmid b$*,*

$$b^{n-1} \equiv 1 \pmod{n},$$

then n *is prime.*

Converse of Fermat's Little Theorem (Second Attempt)

Conjecture

Let $n \in \mathbb{N}$*. If for all* $b \in \mathbb{N}$ *such that* $n \nmid b$*,*

$$b^{n-1} \equiv 1 \pmod{n},$$

then n *is prime.*

Counterexample

For all $n \nmid b$,

$$b^{560} \equiv 1 \pmod{561},$$

$$\text{but} \quad 560 = 3 \cdot 11 \cdot 16.$$

561 is a *Carmichael number*.

Lucas's Primality Test

Theorem

Let $n \in \mathbb{N}$ *and* $a \in \mathbb{Z}$ *with* $\gcd(n, a) = 1$. *If for all primes* $p \mid n - 1$, $a^{(n-1)/p} \not\equiv 1 \pmod{n}$, *then* n *is prime.*

Lucas's Primality Test

Theorem

Let $n \in \mathbb{N}$ *and* $a \in \mathbb{Z}$ *with* $\gcd(n, a) = 1$*. If for all primes* $p \mid n-1$, $a^{(n-1)/p} \not\equiv 1 \pmod{n}$*, then* n *is prime.*

True!

Generalization of Fermat's Little Theorem

Theorem

Let $a \in \mathbb{Z}$ *and* $n \in \mathbb{N}$ *with* $\gcd(a, n) = 1$ *and* $n \geq 2$*.* n *is prime if and only if*

$$(X + a)^n \equiv X^n + a \pmod{n}.$$

Generalization of Fermat's Little Theorem

Proof.

Sufficiency ($\implies$). Suppose n is prime, and let $0 < i < n$. No positive integer less than the prime n divides n, so

$$\binom{n}{i} = \frac{n!}{(n-i)!\,i!} = n \cdot \underbrace{\frac{(n-1)!}{(n-i)!\,i!}}_{\in \mathbb{N}} \quad \implies n \mid \binom{n}{i}$$

$$\implies \quad \binom{n}{i} X a^i \equiv 0 \pmod{n}.$$

Finally, $a^n \equiv a \pmod{n}$ by Fermat's Little Theorem. Thus $(X+a)^n \equiv X^n + a \pmod{n}$.

Generalization of Fermat's Little Theorem

Proof.

Necessity ($\neg \implies \neg$). Suppose n is composite.
There exists a prime power $p^k \mid n$ but $p^{k+1} \nmid n$. Consider the term

$$\binom{n}{p} X^{n-p} a^p = \frac{n!}{(n-p)!\,p!} X^{n-p} a^p \stackrel{?}{\equiv} 0 \pmod{n}.$$

Let $f(m)$ denote the number of times p divides m.

$$\begin{aligned} & f(n!) = f((n-p)!) + f(n) = f((n-p)!) + k \\ \text{and} \quad & f(p!) = 1 \\ \implies \quad & f\binom{n}{p} = f(n!) - f((n-p)!) - f(p!) \\ & = f((n-p)!) + k - f((n-p)!) - 1 \\ & = k - 1. \end{aligned}$$

Generalization of Fermat's Little Theorem

Proof.

Necessity $(\neg \implies \neg)$.
Therefore, $p^{k-1} \mid \binom{n}{p}$ but $p^k \nmid \binom{n}{p}$.
Thus $n \nmid \binom{n}{p}$.
But $\gcd(a, n) = 1$, so

$$\begin{aligned} & \binom{n}{p} X^{n-p} a^p \not\equiv 0 \pmod{n} \\ \implies \quad & (X + a)^n \not\equiv X^n + a \pmod{n}. \end{aligned}$$

□

Algorithmic Complexity

Definition (P)

P denotes the set of problems that can be solved by some algorithm in polynomial time on the length of the input n.

Algorithmic Complexity

Definition (P)

P denotes the set of problems that can be solved by some algorithm in polynomial time on the length of the input n.

When $n \in \mathbb{N}$, the "length" of n is the number of bits (**b**inary dig**its**) in n, i.e. $\lceil \log_2(n) \rceil$.

Algorithmic Complexity

Definition (P)

P denotes the set of problems that can be solved by some algorithm in polynomial time on the length of the input n.

When $n \in \mathbb{N}$, the "length" of n is the number of bits (**b**inary dig**its**) in n, i.e. $\lceil \log_2(n) \rceil$.

Given an algorithm $\mathcal{A}$ and input $n \in \mathbb{N}$, let

$$T(\mathcal{A}, n) = \text{number of "steps" } A \text{ takes to terminate on input } n.$$

If

$$T(\mathcal{A}, n) = O(\log^k n)$$

for some $k \geq 0$, then $\mathcal{A} \in \mathtt{P}$.

Algorithmic Complexity

Example

The question `POW10` "is $n \in \mathbb{N}$ a power of 10" is in $\mathtt{P}$ because it can be solved by the following algorithm:

while $n > 1$ **do**
  $n \leftarrow \lfloor \frac{n}{10} \rfloor$
return `YES` if $n = 1$; `NO` if $n = 0$

This algorithm terminates in $\lceil \log_{10} n \rceil$ steps, so $\mathtt{POW10} \in \mathtt{P}$.

PRIMES

Definition

PRIMES denotes the question, given any positive integer $n \geq 2$, "is n prime?"

PRIMES

Definition

PRIMES denotes the question, given any positive integer $n \geq 2$, "is n prime?"

Theorem

PRIMES $\in$ *P*.

PRIMES

Definition

PRIMES denotes the question, given any positive integer $n \geq 2$, "is n prime?"

Theorem

PRIMES $\in$ *P*.

That is, there exists an algorithm that solves PRIMES in $O(\log^k n)$ steps for some $k \geq 0$.

The AKS Primality Test

“PRIMES is in P”

Authors: Manindra Agrawal, Neeraj Kayal, and Nitin Saxena

- Grad students at the Indian Institute of Technology Kapur
- Published in 2002.
- Appeared in *Annals of Mathematics*

The AKS Primality Test: Pseudocode

Algorithm Check the primality of a positive integer $n \geq 2$.

Input: $n \in \mathbb{N}$ with $n \geq 2$.

Output: `PRIME` when n is prime; `COMPOSITE` when n is composite.

```
if n = a^b for some a, b ∈ ℕ with b ≥ 2 then
    return COMPOSITE
Find the smallest r ∈ ℕ such that o_r(n) > log^2(n).
if 1 < gcd(a, n) < n for some a ≤ r then
    return COMPOSITE
if n ≤ r then
    return PRIME
for all 0 ≤ a ≤ ⌊√φ(r) log n⌋ do
    if (X + a)^n ≢ X^n + a (mod X^r − 1, n) then
        return COMPOSITE
return PRIME
```

The AKS Primality Test: Pseudocode

Algorithm Check the primality of a positive integer $n \geq 2$.

Input: $n \in \mathbb{N}$ with $n \geq 2$.
Output: `PRIME` when n is prime; `COMPOSITE` when n is composite.

```
if n = a^b for some a, b ∈ ℕ with b ≥ 2 then
    return COMPOSITE
Find the smallest r ∈ ℕ such that o_r(n) > log²(n).
if 1 < gcd(a, n) < n for some a ≤ r then
    return COMPOSITE
if n ≤ r then
    return PRIME
for all 0 ≤ a ≤ ⌊√φ(r) log n⌋ do
    if (X + a)^n ≢ X^n + a (mod X^r − 1, n) then
        return COMPOSITE
return PRIME
```

The AKS Primality Test: Insights

Generalization of FIT

if $(X + a)^n \not\equiv X^n + a \pmod{n}$ **then**

    return `COMPOSITE`

else

    return `PRIME`

The AKS Primality Test: Insights

Generalization of FIT

if $(X + a)^n \not\equiv X^n + a \pmod{n}$ **then**
    return `COMPOSITE`
else
    return `PRIME`

AKS Algorithm (Last Step)

for all $0 \le a \le \lfloor \sqrt{\phi(r)} \log n \rfloor$ **do**
    if $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1, n}$ **then**
        return `COMPOSITE`
return `PRIME`

The AKS Primality Test: Time Complexity

Theorem

The algorithm terminates in $O\left(\log^{21/2} n\right)$ *time.*

The AKS Primality Test: Time Complexity

Theorem

The algorithm terminates in $O\left(\log^{21/2} n\right)$ *time.*

Sketch of proof.

The bottleneck is the last step. $\lfloor\sqrt{\phi(r)}\log n\rfloor$ different equations must be verified.

for all $0 \le a \le \lfloor\sqrt{\phi(r)}\log n\rfloor$ **do**
  if $(X+a)^n \not\equiv X^n + a \pmod{X^r - 1, n}$ **then**
    return `COMPOSITE`

- $\phi(r) = O(r)$.
- $r = O(\log^5 n)$.
- Verify $\lfloor\sqrt{\phi(r)}\log n\rfloor = O\left(\sqrt{r}\log n\right) = O(\log^{7/2} n)$ equations.

□

The AKS Primality Test: Time Complexity

Sketch of proof.

- Binary exponentiation on $(X + a)^n$ requires only $O(\log n)$ polynomial multiplications.
- Each polynomial multiplication is either a square or multipication by $(X + a)$, requiring at most $O(r)$ coefficient multiplications.
- Coefficients can be multiplied in $O(\log n)$ time.
- Thus each congruence can be verified in $O(r \log^2 n) = O(\log^7 n)$.
- So all equations can be verified in $O(\log^{7/2} n \cdot \log^7 n) = O(\log^{21/2} n)$ time.

□

Conclusion

- That AKS algorithm runs in polynomial time doesn't mean that the AKS algorithm is *fast* for all n.
- There exist far better algorithms for smaller $n \in \mathbb{N}$.

References I

[1, 2, 3]

M. Agrawal, N. Kayal, and N. Saxena, "Primes is in p," *Annals of mathematics*, pp. 781–793, 2004.

D. H. Lehmer, "Tests for primality by the converse of fermat's theorem," *Bulletin of the American Mathematical Society*, vol. 33, no. 3, pp. 327–340, 1927.

P. D. Schumer, *Introduction to number theory*. Brooks/Cole Publishing Company, 1996.