Testing Primality in Polynomial Time

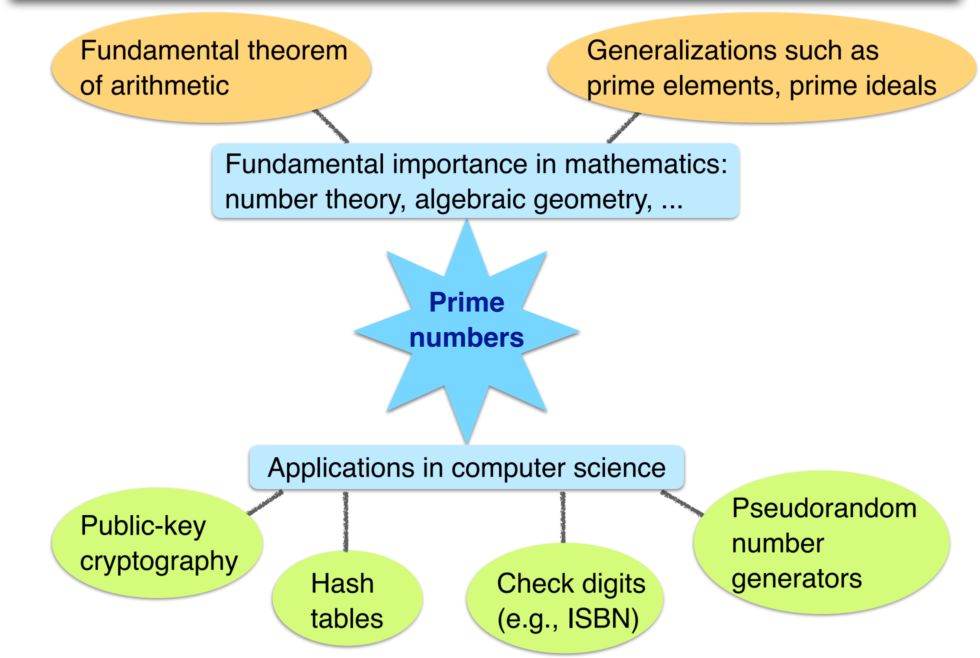
A groundbreaking result
by M. Agrawal, N. Kayal, and N. Saxena (2002)

Thomas Schneider

7 March 2016

Motivation and Background

The importance of being prime



The importance of testing primality

Decision problem PRIMES

Input: $n \in \mathbb{N}$ in binary representation
Question: Is n prime?

PRIMES is used in order to ...

Public-key cryptography	... generate reliable keys in RSA (most widely used cryptosystem)
Hash tables	... generate hash tables of favorable size with some hashing algorithms
Pseudorandom number generators	... obtain optimal period lengths

A brief history of primality testing

- Simple test, known by the ancient Greeks:

  For all natural numbers $i \leq \sqrt{n}$, test whether $i \mid n$.

  - Related to Sieve of Eratosthenes (≈ 240 BC)
  - Requires $O(\sqrt{n})$ steps ⇝ inefficient! Input length is $\lceil \log n \rceil$

- "Near" test, based on Fermat's little theorem (1640):

  For any prime p and any a: $a^p \equiv a \pmod{p}$

  - $a^n \pmod{n}$ can be computed efficiently (repeated squaring)
  - Alas, many composites satisfy the congruence for some a's
  - Still, FLT is the basis of many modern primality tests

A brief history of primality testing

- Upper complexity bound: PRIMES ∈ NP ∩ coNP [Pratt, 1974]
- Randomized polytime algorithms with probabilistic output (1970's)
  - Miller & Rabin 1975/80
  - Solovay & Strassen 1977

  both shown to be in PTime under *Extended Riemann Hypothesis*
- First quasi-polynomial test: (1983)
  - by Adleman, Pomerance, Rumely
  - Runtime $(\log n)^{O(\log \log \log n)}$
- Randomized algorithms with exact output in expected polytime (1980/90's)
  - Goldwasser & Kilian 1986
  - Adleman & Huang 1992

  new: produce easily verifiable short certificates for primality

The AKS algorithm

- First unconditional, exact PTime algorithm
  - 2002 by Agrawal, Kayal, Saxena (IIT Kanpur)
  - received huge resonance in scientific literature and the media
  - Gödel and Fulkerson prizes 2006
  - aka *cyclotomic AKS test*
  - uses relatively simple mathematics (number theory, basic algebra)
  - time bounds:
    - $O(\log^{10.5+\varepsilon} n)$ in the original version
    - can be improved with more machinery to $O(\log^{6+\varepsilon} n)$ [Lenstra Jr. & Pomerance 2005–15]
    - under certain assumptions even $O(\log^{3+\varepsilon} n)$

The AKS Algorithm

A simple characterization of primality

Proposition

Let $n \in \mathbb{N}$, $n \geq 2$, $a \in \mathbb{Z}$, $(a, n) = 1$.
Then n is prime iff

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

Example: $n = 3$

$$\begin{aligned}(X+a)^3 &= X^3 + 3aX^2 + 3a^2X + a^3 \\ &\equiv X^3 + a^3 \pmod{3} \\ &\equiv X^3 + a \pmod{3} \quad \text{by Fermat's little Theorem}\end{aligned}$$

✓

Example: $n = 4$ $a = 1$

$$\begin{aligned}(X+1)^4 &= X^4 + 4X^3 + 6X^2 + 4X + 1 \\ &\equiv X^4 + 2X^2 + 1 \pmod{4}\end{aligned}$$

↯

A simple characterization of primality

Proposition

Let $n \in \mathbb{N}$, $n \geq 2$, $a \in \mathbb{Z}$, $(a, n) = 1$.
Then n is prime iff

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

Proof.

Simple number-theoretic argument, involving Fermat's little theorem

A naïve primality test

If $(X + 1)^n \equiv X^n + 1 \pmod{n}$, then n is prime, otherwise n is composite.

Input size: $\log n$

Runtime:

- Computation of $(X + 1)^n$ requires only $O(\log n)$ multiplications *(exponentiation via repeated squaring)*
- $(X + 1)^n$ has up to $n + 1$ coefficients! **Pseudo-polynomial!**

Remedy

Evaluate both sides of the congruence modulo a polynomial of small degree! 💡

Reducing the number of coefficients

Instead, we want to test whether

$$(X + a)^n \equiv X^n + a \pmod{X^r - 1,\ n}$$

for a "suitable, small enough" $r \leq O(\log^k n)$.

Observation

- All prime numbers satisfy this congruence,
- but not all composite numbers violate it!

Remedy

Verify the congruence for *several* values of a

↑ but only $O(\log^k n)$ many!

The AKS algorithm

Input: integer $n > 1$

1 if $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ then return `COMPOSITE`

2 find smallest r such that $o_r(n) > \log^2 n$

3 if $1 < (a, n) < n$ *for some* $a \leq r$ then return `COMPOSITE`

4 if $n \leq r$ then return `PRIME`

5 for $a = 1$ *to* $\lfloor \sqrt{\phi(r)} \log n \rfloor$ do

6   if $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1,\ n}$ then return `COMPOSITE`

7 return `PRIME`

2 $o_r(n)$ is the *order of n modulo r*:

- defined only for $(n, r) = 1$
- $o_r(n) \ :=\ $ smallest k with $n^k \equiv 1 \pmod{r}$

For every n, there is $r \leq \max\{3, \lceil \log^5 n \rceil\}$ with $o_r(n) > \log^2 n$.

The AKS algorithm

Input: integer $n > 1$

1 if $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ then return `COMPOSITE`

2 find smallest r such that $o_r(n) > \log^2 n$

3 if $1 < (a, n) < n$ *for some* $a \leq r$ then return `COMPOSITE`

4 if $n \leq r$ then return `PRIME`

5 for $a = 1$ *to* $\lfloor \sqrt{\phi(r)} \log n \rfloor$ do

6   if $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1,\ n}$ then return `COMPOSITE`

7 return `PRIME`

5 $\phi(r)$ is *Euler's totient function* of r: $\#\{i \mid 1 \leq i \leq r \text{ and } (i, r) = 1\}$

Termination and completeness

Input: integer $n > 1$

1 if $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ then return `COMPOSITE`

2 find smallest r such that $o_r(n) > \log^2 n$

3 if $1 < (a, n) < n$ *for some* $a \leq r$ then return `COMPOSITE`

4 if $n \leq r$ then return `PRIME`

5 for $a = 1$ *to* $\lfloor \sqrt{\phi(r)} \log n \rfloor$ do

6   if $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1,\ n}$ then return `COMPOSITE`

7 return `PRIME`

Obvious properties:

✓ Termination

✓ Completeness:
if n is prime, then the algorithm returns `PRIME`

Soundness and time bounds

Input: integer $n > 1$

1 if $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ then return `COMPOSITE`

2 find smallest r such that $o_r(n) > \log^2 n$

3 if $1 < (a, n) < n$ *for some* $a \leq r$ then return `COMPOSITE`

4 if $n \leq r$ then return `PRIME`

5 for $a = 1$ *to* $\lfloor \sqrt{\phi(r)} \log n \rfloor$ do

6   if $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1,\ n}$ then return `COMPOSITE`

7 return `PRIME`

It remains to show:

- Soundness:
  if the algorithm returns `PRIME`, then n is prime
- Polynomial runtime

Soundness

Suppose the algorithm returns PRIME.

1 **if** $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ **then return** COMPOSITE
2 find smallest r such that $o_r(n) > \log^2 n$
3 **if** $1 < (a, n) < n$ *for some* $a \le r$ **then return** COMPOSITE
4 **if** $n \le r$ **then return** PRIME
5 **for** $a = 1$ *to* $\lfloor \sqrt{\phi(r)} \log n \rfloor$ **do**
6 **if** $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1,\ n}$ **then return** COMPOSITE
7 **return** PRIME

- If it returns from line 4, then n is prime (cf. line 3).
- From now assume it returns from line 7.

The central congruence revisited

Due to line 6, we have

$$(X + a)^n \equiv X^n + a \pmod{X^r - 1,\ n}$$

Let p be a prime divisor of n. Then

$$(X + a)^n \equiv X^n + a \pmod{X^r - 1,\ p}$$

$$(X + a)^p \equiv X^p + a \pmod{X^r - 1,\ p}$$

Since p divides n, we have

$$(X + a)^{\frac{n}{p}} \equiv X^{\frac{n}{p}} + a \pmod{X^r - 1,\ p}$$

$\rightsquigarrow$ Both n and $\frac{n}{p}$ behave like p in this congruence.

AKS call them introspective for the polynomial $X + a$.

Introspective numbers

We fix r and p.

$m \in \mathbb{N}$ is called introspective for polynomial $f(X)$ if

$$f(X)^m \equiv f(X^m) \pmod{X^r - 1,\ p}$$

Lemma

1. If m, m' are introspective for $f(X)$, then so is $m \cdot m'$.
2. If m is introspective for $f(X)$ and $f'(X)$, then also for $f(X) \cdot f'(X)$.

Proof: elementary number theory

The remainder of the proof

Let $\ell := \lfloor \sqrt{\phi(r)} \log n \rfloor$

1. Define sets I of all products of powers of $\frac{n}{p}$ and p
   and P of all products of ℓ powers of $X + a$
   Easy to see: every $m \in I$ is introspective for every $f(X) \in P$.
2. Define groups G_1 = "I modulo r"
   and G_2 = "P modulo $(h(X), p)$"
   ($h(X)$: an irreducible factor of $X^r - 1$)
3. Show: if n is not a power of p, then $|G_2| \le n^{\sqrt{|G_1|}} < |G_2|$ (algebra)
4. Conclude that $n = p^k$ and $k = 1$ (line 1)

   1 **if** $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ **then return** COMPOSITE
5. $\rightsquigarrow$ $n = p$ prime!

Polynomial bound on the runtime

1 **if** $n = a^b$ *for* $a \in \mathbb{N}$ *and* $b > 1$ **then return** COMPOSITE
2 find smallest r such that $o_r(n) > \log^2 n$
3 **if** $1 < (a, n) < n$ *for some* $a \le r$ **then return** COMPOSITE
4 **if** $n \le r$ **then return** PRIME
5 **for** $a = 1$ *to* $\lfloor \sqrt{\phi(r)} \log n \rfloor$ **do**
6   **if** $(X + a)^n \not\equiv X^n + a \pmod{X^r - 1, n}$ **then return** COMPOSITE
7 **return** PRIME

Runtime dominated by lines 5–6

$O(\log^{10.5+\varepsilon} n)$: $O(\sqrt{r} \log n)$ iterations
$\times\, O(\log n)$ multiplications of degree-r polynomials
$\times\, O(r \log^{1+\varepsilon} n)$ per multiplication

Improvements on the time bounds

1. Under certain conjectures, $r \le O(\log^2 n)$
   $\rightsquigarrow$ overall time bound $O(\log^{6+\varepsilon} n)$
2. Modification of the AKS algorithm with proven bound $O(\log^{6+\varepsilon} n)$
   (Lenstra Jr. and Pomerance 2002–15)
3. Under another (debated) conjecture,
   achieve $r \le O(\log n)$ and test only *one* congruence
   $\rightsquigarrow$ overall time bound $O(\log^{3+\varepsilon} n)$

Discussion

Lessons learned

- Primality can be tested in deterministic polynomial time.
- Runtime $\log^{10.5} n$ can be shown using relatively simple maths.
- Runtime can be improved to $\log^6 n$ (with more complex arguments).
  Conjecture: $\log^3 n$
- Does the result break RSA? No!
- The breakthrough is theoretical, not practical:
  AKS is by far outperformed by existing randomized algorithms

Life after “PRIMES is in P”

- Accelerations
  described by various authors already in 2002–3
  (Bernstein, Lenstra, Poonen, Vaaler, Voloch)
  - restricting the size of r, ℓ → speedup by 2 million!
  - faster integer squaring
  - use of different polynomials in place of $X + a$
- Combining AKS and randomness
  reduces runtime from $\log^6 n$ to $\log^4 n$
  (Berrizbeitia 2005, Cheng 2003, Bernstein 2007)
- Competing deterministic approach based on pseudosquares
  reduces runtime to $\log^3 n$ under reasonable assumptions
  confirmed for $n \leq 2^{80}$ (Lukes 1996)
- Hope for “Graph Isomorphism is in P”?
  pseudo-polynomial upper bound shown in 2015 by Babai

The end

Questions? ¿Preguntas? Fragen? Vragen?

Thank you very much
for your attention!

Pytania? Kysymyksiä? Vrae? Întrebări? Questões? Вопросы?