The number-field sieve
Finding small factors of integers
Speed of the number-field sieve
Proving primality
in polynomial time
Proving primality more quickly
D. J. Bernstein
University of Illinois at Chicago
Compositeness proofs

If \( n \) is prime and \( b \in \mathbb{Z} \) then \( b^n - b \in n\mathbb{Z} \).

Have easy difference-of-squares factorization of \( b^n - b \),
depending on \( \text{ord}_2(n - 1) \).

e.g.: If \( n \in 5 + 8\mathbb{Z} \) is prime and \( b \in \mathbb{Z} \) then \( b \in n\mathbb{Z} \) or \( b^{(n-1)/2} + 1 \in n\mathbb{Z} \) or \( b^{(n-1)/4} + 1 \in n\mathbb{Z} \) or \( b^{(n-1)/4} - 1 \in n\mathbb{Z} \).
An integer \( n \geq 2 \) is “\( b \)-sprp” iff it divides one of the difference-of-squares factors of \( b^n - b \).

Every prime is \( b \)-sprp.

For each composite \( n \), most \( b \)'s have \( n \) not \( b \)-sprp.

Very few composites are 2-sprp. No known composites are “BPSW-$620$-prp.” But we think that there are infinitely many exceptions.
Given $n \geq 2$: Try random $b$. If $n$ is not $b$-sprp, have proven $n$ composite. Otherwise keep trying.

Given composite $n$, this algorithm finds compositeness certificate $b$.

Proven random cost $(\lg n)^{2+o(1)}$ to find certificate. Proven deterministic cost $(\lg n)^{2+o(1)}$ to verify certificate.

Can we do better? Open: Is there a compositeness certificate findable in cost $(\lg n)^{O(1)}$, verifiable in cost $(\lg n)^{1+o(1)}$?
Given prime $n$, this algorithm loops forever. After many $b$’s we are confident that $n$ is prime . . . but we don’t have a proof.

Do we need a proof?

For competent cryptographers: No.

For paranoid bankers: Yes.

For pure computational number theorists: Who cares? Proving primality is an interesting challenge.
Combinatorial primality proofs

Recall primality algorithm discussed yesterday.

Output of algorithm: primality proof for \( n \), or compositeness proof for \( n \).

Proven deterministic cost
\[
\leq (\lg n)^{10.5 + o(1)}.
\]

Conjectured deterministic cost
\[
\leq (\lg n)^{6 + o(1)}.
\]

Can we do better?
Complicated variant of algorithm and complicated proof produce better theorem:
Proven deterministic cost
\[ \leq (\lg n)^{6+o(1)}. \]

Open: Is there a primality-proving algorithm with proven deterministic cost
\[ \leq (\lg n)^{5+o(1)}? \]
Another variant of algorithm achieves better exponent at the expense of determinism. Proven random cost
\[ \leq (\log n)^{4+o(1)}.\]

Open: Is there a primality-proving algorithm with proven random cost
\[ \leq (\log n)^{3+o(1)}.\]

Open: Is there a primality-proving algorithm reasonably conjectured to have cost
\[ \leq (\log n)^{3+o(1)}?\]
Precomputed primality proofs

e.g.: An integer $n \in [2, 2^{48}]$
is prime iff it is a 2-sprp, 3-sprp, 5-sprp, 7-sprp, 11-sprp, 13-sprp, and 17-sprp.

Verifying this was extremely slow; but now that we know it, can quickly check primality of any $n \in [2, 2^{48}]$.

Conjectured cost $\leq (\lg n)^{3+o(1)}$ for primality proof after massive precomputation.
e.g.: An integer $n \in [2^{20}, 2^{100}]$ is prime iff

- $r^{(n-1)/2} \equiv \pm 1 \pmod{n}$ for all primes $r \leq 367$;
- $r^{(n-1)/2} \equiv -1 \pmod{n}$ for some odd prime $r \leq 367$ if $n \mod 8 = 1$;
- $2^{(n-1)/2} \equiv -1$ if $n \mod 8 = 5$;
- $n$ is not a perfect power; and
- $n$ has no prime divisors $< 2^{20}$.

Conjectured cost $\leq (\lg n)^{3+o(1)}$ for these “pseudosquares” primality proofs after somewhat less massive precomputation.
Open: Is there a primality-proving algorithm reasonably conjectured to have cost $\leq (\lg n)^{2+o(1)}$ after precomputation?

Open: Is there a primality-proving algorithm reasonably conjectured to have cost $\leq (\lg n)^{3+o(1)}$ after $n^{1/2+o(1)}$ precomputation?

Open: Is there a primality-proving algorithm reasonably conjectured to handle $(\lg n)^{O(1)}$ inputs $\approx n$ in cost $\leq (\lg n)^{3+o(1)}$ per input?
Primality proofs using curves

“Fast elliptic-curve primality proving” (FastECPP):

Conjectured cost \( \leq (\lg n)^{4+o(1)} \) to find certificate proving primality of \( n \).

Proven deterministic cost \( \leq (\lg n)^{3+o(1)} \) to verify certificate.
Variant using genus-2 hyperelliptic curves:

Proven random cost \((\lg n)^{O(1)}\) to find certificate proving primality of \(n\).

Proven deterministic cost \(\leq (\lg n)^{3+o(1)}\) to verify certificate.
Variant using elliptic curves with large power-of-2 factors:

Proven existence of certificate proving primality of $n$.

Proven deterministic cost $\leq (\lg n)^{2+o(1)}$ to verify certificate.

Open: Is there a primality certificate verifiable in cost $(\lg n)^{1+o(1)}$?
Verifying curve proofs

Main theorem in a nutshell: If an elliptic curve \( E(\mathbb{Z}/n) \) has a point of prime order \( q > (\lceil n^{1/4} \rceil + 1)^2 \) then \( n \) must be prime.

Proof in a nutshell: If \( p \) is a prime divisor of \( n \) then the same point mod \( p \) has order \( q \) in \( E(\mathbb{F}_p) \), but \( \#E(\mathbb{F}_p) \leq (\sqrt{p} + 1)^2 \), so \( n^{1/2} < p \).
More concretely:

Given odd integer $n \geq 2$, $a \in \{6, 10, 14, 18, \ldots \}$, integer $b$, $\gcd \{n, b^3 + ab^2 + b\} = 1$, $\gcd \{n, a^2 - 4\} = 1$, prime $q > (\lceil n^{1/4} \rceil + 1)^2$:

Define $x_1 = b$, $z_1 = 1$, $x_{2i} = (x_i^2 - z_i^2)^2$, $z_{2i} = 4x_i z_i (x_i^2 + ax_i z_i + z_i^2)$, $x_{2i+1} = 4(x_i x_{i+1} - z_i z_{i+1})^2$, $z_{2i+1} = 4b(x_i z_{i+1} - z_i x_{i+1})^2$.

Claim: If $z_q \in n \mathbb{Z}$ and $\gcd \{n, x_q\} = 1$ then $n$ is prime.
For each prime $p$ dividing $n$:

$$(a^2 - 4)(b^3 + ab^2 + b) \neq 0 \text{ in } \mathbb{F}_p,$$

so $(b^3 + ab^2 + b)y^2 = x^3 + ax^2 + x$ is an elliptic curve over $\mathbb{F}_p$.

$(b, 1)$ is a point on curve.

Inductive claims:

if $z_i \neq 0$ in $\mathbb{F}_p$ then

$i(b, 1) = (x_i/z_i, \ldots)$ on curve;

if $x_i \neq 0, z_i = 0$ in $\mathbb{F}_p$ then

$i(b, 1) = \infty$ on curve.

$x_q \neq 0, z_q = 0$ in $\mathbb{F}_p$
so $q(b, 1) = \infty$ on curve.

So $n$ is prime.
Oops: Nobody has written down full proofs of these claims. Maybe the claims aren’t true in certain annoying special cases.

Traditional solution: Recognize and exclude all of the annoying cases by checking conditions such as \( \gcd\{n, z_i\} = 1 \) for each \( i \) used in computation.

Messy; slows down computation; but adequate for current proofs.
Finding curve proofs

To prove primality of $n$: Choose random $E$. Use Schoof’s algorithm to compute $\#E(\mathbb{Z}/n)$.

Compute $q = \#E(\mathbb{Z}/n)/2$. If $q$ doesn’t seem prime, try another $E$.

If $q \geq n$ or $q \leq ([n^{1/4}] + 1)^2$: $n$ is small; easy base case.

Otherwise:
Recursively prove primality of $q$.
Choose random point $P$ on $E$.
If $2P = \infty$, try another $P$.
Now $2P$ has prime order $q$. 
Schoof’s algorithm costs \((\lg n)^{5+o(1)}\).

Conjecturally find prime \(q\) after \((\lg n)^{1+o(1)}\) curves on average. Reduce number of curves by allowing larger ratios \(#E(\mathbb{Z}/n)/q\).

Recursion involves \((\lg n)^{1+o(1)}\) levels. Reduce number of levels by allowing and demanding larger ratios \(#E(\mathbb{Z}/n)/q\).

Overall cost \((\lg n)^{7+o(1)}\).
Faster way to generate curves with known number of points: generate curves with small-discriminant “complex multiplication” (CM). Reduces conjectured cost to \((\lg n)^{4+o(1)}\).

CM has applications beyond primality proofs: e.g., can generate CM curves with low embedding degree for pairing-based cryptography.
Complex multiplication

Consider positive squarefree integers $D \in 3 + 4\mathbb{Z}$.
(Can allow some other $D$’s too.)

If prime $n$ equals $(u^2 + Dv^2)/4$ then “CM with discriminant $-D$” produces curves over \(\mathbb{Z}/n\) with $n + 1 \pm u$ points.

Assuming $D \leq (\lg n)^{2+o(1)}$:
Cost $\sim (\lg n)^{2.5+o(1)}$.
Fancier algorithms: $\sim (\lg n)^{2+o(1)}$. 
First step: Find all vectors $(a, b, c) \in \mathbb{Z}^3$ with $\gcd\{a, b, c\} = 1$, $-D = b^2 - 4ac$, $|b| \leq a \leq c$, and $b \leq 0 \Rightarrow |b| < a < c$.

How?
Try each integer $b$ between $-\lfloor \sqrt{D/3} \rfloor$ and $\lfloor \sqrt{D/3} \rfloor$.
Find all small factors of $b^2 + D$.
Find all factors $a \leq \lfloor \sqrt{D/3} \rfloor$.
For each $(a, b)$, find $c$ and check conditions.
Second step: For each \((a, b, c)\) compute \(j(-b/2a + \sqrt{-D/2a}) \in \mathbb{C}\) to high precision.

Some wacky standard notations:

\[q(z) = \exp(2\pi i z).\]

\[\eta^{24} = q \left( 1 + \sum_{k \geq 1} (-1)^k q^{k(3k-1)/2} \right)^{24} + \sum_{k \geq 1} (-1)^k q^{k(3k+1)/2} \right)^{24}.

\[f^{24}_1(z) = \eta^{24}(z/2)/\eta^{24}(z)\]

\[j = (f^{24}_1 + 16)^3 / f^{24}_1.\]
How much precision is needed?

Answer: \[ \leq (\lg n)^{1+o(1)} \text{ bits}; \]
\[ \leq (\lg n)^{0.5+o(1)} \text{ terms in sum}; \]
\[ \leq (\lg n)^{1+o(1)} \text{ inputs } (a, b, c); \]
\[ \text{total cost } \leq (\lg n)^{2.5+o(1)}. \]

In practice: No need to carefully analyze precision. 
Start with low precision; 
if precision is too small, 
retry with double precision.

Later steps of computation will notice if precision is too small.
Third step: Compute product $H_{-D} \in \mathbb{C}[x]$
of $x - j(-b/2a + \sqrt{-D}/2a)$
over all $(a, b, c)$.

Amazing fact: $H_{-D} \in \mathbb{Z}[x]$.
The $j$ values are algebraic integers generating a “class field.”

$\leq (\lg n)^{1+o(1)}$ factors.
Cost $\leq (\lg n)^{2+o(1)}$. 
Fourth step: Find a root $r$ of $H_D$ in $\mathbb{Z}/n$.

Easy since $n$ is prime.

Amazing fact: the curve $y^2 = x^3 + (3x + 2)r/(1728 - r)$ has $n + 1 + u$ points for some $(u, v)$ with $4n = u^2 + Dv^2$. 
FastECPP using CM

To prove primality of \( n \):

Choose \( y \in (\lg n)^{1+o(1)} \).

For each odd prime \( p \leq y \), compute square root of \( p \) in quadratic extension of \( \mathbb{Z}/n \). Also square root of \(-1\).

Each square root costs \( (\lg n)^{2+o(1)} \).

Total cost \( (\lg n)^{3+o(1)} \).
For each positive squarefree $y$-smooth $D \in 3 + 4\mathbb{Z}$ below $(\lg n)^{2+o(1)}$, compute square root of $-D$ in quadratic extension of $\mathbb{Z}/n$.

Each square root costs $(\lg n)^{1+o(1)}$: simply multiply square roots of primes.

Total cost $(\lg n)^{3+o(1)}$. 
For each $D$ having $\sqrt{-D} \in \mathbb{Z}/n$, find $u, v$ with $4n = u^2 + Dv^2$, if possible.

This can be done by a half-gcd computation. Each $D$ costs $(\lg n)^{1+o(1)}$. Total cost $(\lg n)^{3+o(1)}$. 
Conjecturally there are \((\lg n)^{1+o(1)}\) choices of \((D, u, v)\).

Look for \(n + 1 \pm u\) having form \(2q\) where \(q\) is prime. More generally:
remove small factors from \(n + 1 \pm u\); then look for primes.

Each compositeness proof costs \((\lg n)^{2+o(1)}\).
Total cost \((\lg n)^{3+o(1)}\).
Conjecturally have several choices of \((D, u, v, q)\), when \(o(1)\)'s are large enough.

Use CM to construct curve with order divisible by \(q\).
Cost \(\leq (\lg n)^{2.5+o(1)}\); negligible.

Problems can occur.
Might have \(n + 1 + u\)
when \(n + 1 - u\) was desired, or vice versa. Curve might not be isomorphic to curve of desired form
\(y^2 = x^3 + ax^2 + x\).
Can work around problems, or simply try next curve.
Recursively prove $q$ prime.
Deduce that $n$ is prime.

$\leq (\lg n)^{1+o(1)}$ levels of recursion.
Total cost $\leq (\lg n)^{4+o(1)}$.

Verification cost $\leq (\lg n)^{3+o(1)}$. 