Primality proving using elliptic curves

F. Morain

Laboratoire d'Informatique de l'École polytechnique

EJCIM, Nancy, 19/03/2007

Plan

Bibliography

- P. Ribenboim. *The new book of prime number records*. Springer-Verlag, 1996.
- D. E. Knuth. *The Art of Computer Programming: Seminumerical Algorithms*. Addison-Wesley, 2nd edition, 1981.
- H. Cohen. *A course in algorithmic algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer–Verlag, 1996. Third printing.
- R. Crandall and C. Pomerance. *Primes – A Computational Perspective*. Springer Verlag, 2000.
- FM. La primalité en temps polynomial [d'après Adleman, Huang; Agrawal, Kayal, Saxena]. Séminaire Bourbaki, Mars 2003.
- FM. Implementing the asymptotically fast version of the elliptic curve primality proving algorithm. *Math. Comp.*, vol. 76, 2007, 493–505.

I. Introduction

Why primes?

- Fundamental problem in computational number theory; also in mathematical computational systems.
- Cryptography.
- Theoretical computer science: to which complexity class does the decision problem **isPrime?** belong?

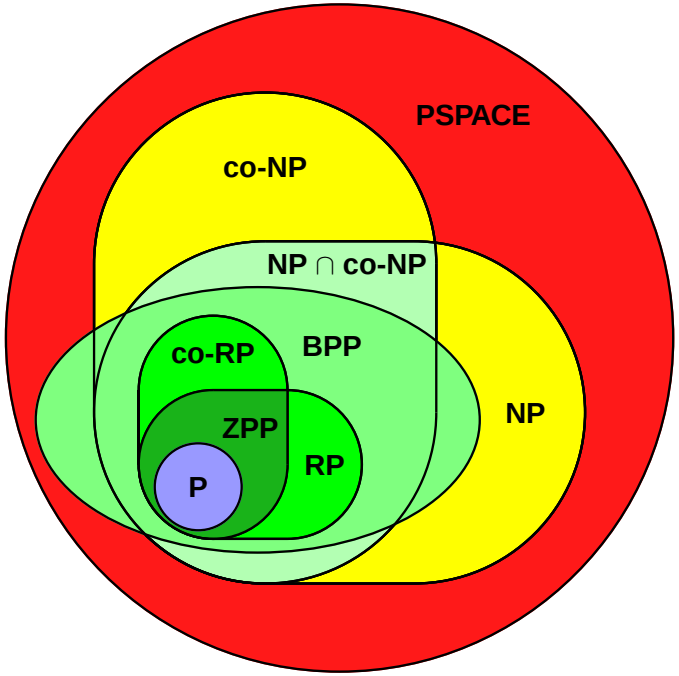
Practical problem: who is prime? Why? How much time is needed?

$$89, 341, 1955097530374556503981, 2^{511} + 111,$$

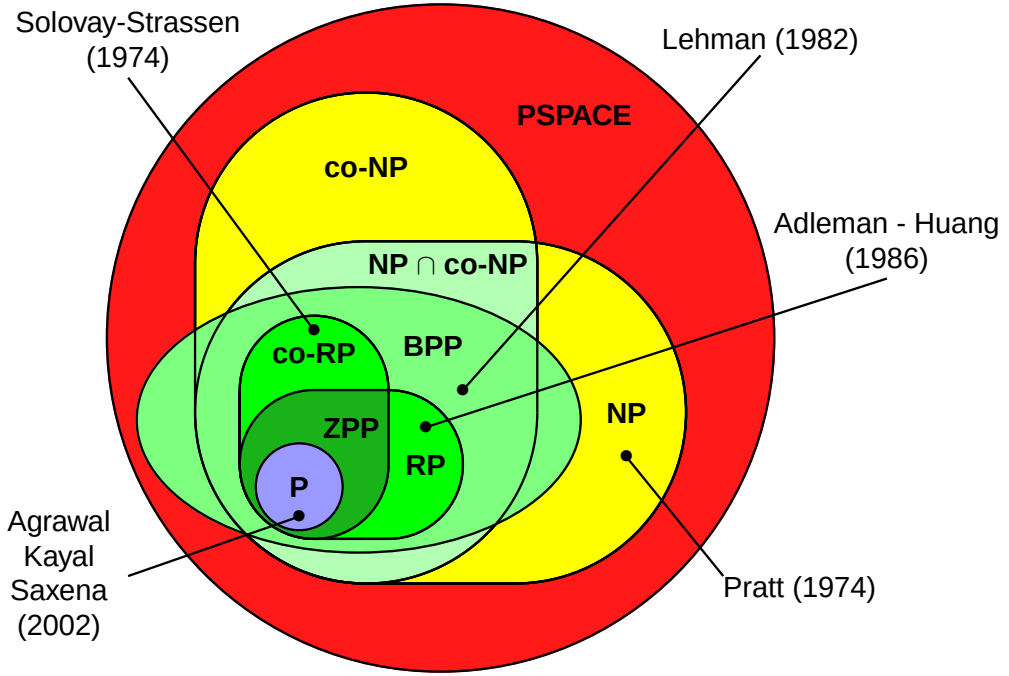
$$R_{1031}, \frac{2^{3539}+1}{3}, 4405^{2638} + 2638^{4405}$$

$$2^{10000} + 177, 10^{5019} + 3^2 \cdot 7^5 \cdot 11^{11}, 2^{30\,402\,457} - 1.$$

Primalité sans complexité n'est que. . .



Primalité sans complexité n'est que. . .



Complexities of *primality proving* algorithms

Adleman, Pomerance, Rumely (1979); Cohen and Lenstra (1984): $O((\log N)^{c \log\log\log N})$.

Adleman-Huang $O((\log N)^?)$

Deterministic, proven: AKS ($\tilde{O}((\log N)^{10.5})$); H. W. Lenstra, Jr. & C. Pomerance ($\tilde{O}((\log N)^6)$).

Miller ($\tilde{O}((\log N)^4)$, under some RH)

Randomized versions of AKS, fastECPP ($\tilde{O}((\log N)^4)$ under various unproven hypotheses)

Pseudosquares ($\tilde{O}((\log N)^3)$ under GRH????)

Rem. $\tilde{O}$ is a natural notation today: fast arithmetic must be and is currently used.

II. The Fermat legacy

Thm. If N is prime and $a \in (\mathbb{Z}/N\mathbb{Z})^*$, then

$$a^{N-1} \equiv 1 \bmod N.$$

Use this as a compositeness test: if $2^{N-1} \not\equiv 1 \bmod N$, then N is composite.

But: $N = 341 = 11 \times 13,\ 2^{340} \equiv 1 \bmod 341$.

Prop. Let $N = \prod_i p_i^{\alpha_i}$. Then the number of false witnesses is $P(N) = \#\{a \in (\mathbb{Z}/N\mathbb{Z})^*, a^{N-1} \equiv 1 \bmod N\} = \prod_i \gcd(p_i - 1, N - 1)$.

Proof. $P(p_i^{\alpha_i}) = \gcd(p_i - 1, N - 1)$. Conclude using CRT. □

Ex. $P(341) = P(11 \times 31) = \gcd(10, 340)\gcd(30, 340) = 100$ (for $a \in \{2, 4, 8, 15, 16, 23, \ldots, 339, 340\}$).

An algorithm

function isCompositeWithFermat(N)
1. Choose $a \in_R \mathbb{Z}/N\mathbb{Z}\backslash\{0\}$.
2. Compute $g = \gcd(a, N)$; **if** $g > 1$, **then** return $(\texttt{yes}, g$ is a factor of $N)$.
3. **if** $a^{N-1} \not\equiv 1 \bmod N$, **then** return $(\texttt{yes}, a)$ **else** return "`I don't know`".

Prop. $\text{Prob}(\text{“}\texttt{Idon'tknow}\text{''}) = P(N)/(N-1)$.

Proof: Prob of exit in Step 2 is $(1 - \varphi(N)/(N-1))$ where $\varphi(N) = \text{Card}((\mathbb{Z}/N\mathbb{Z})^*)$ (Euler's function).
Prob of success in Step 3 is:

$$\frac{\varphi(N)}{N-1}\left(1 - \frac{P(N)}{\varphi(N)}\right). \square$$

So what?

- If N is prime, proba = 1 (since $P(N) = N - 1$).
- Very often, $P(N)$ is small. But it is possible to have $P(N) = \varphi(N)$ for Carmichael numbers (there is an infinite number of such numbers, following Alford, Granville, Pomerance).

What next?

- Solovay-Strassen;
- Artjuhov-Dubois-Selfridge-Miller-Rabin;
- Lucas, linear recurring sequences of ordre 2, 3, etc.;
- elliptic curves;
- polynomials (Grantham, . . .);
- combinaisons of tests ($N \pm 1$, Atkin).

Ultimate goal: find a very fast test with as small error probability as possible (Grantham, Zhang, etc.).

The first (partial) primality proving algorithm

Thm. N is prime iff $(\mathbb{Z}/N\mathbb{Z})^*$ is cyclic of order $N-1$:

$$\left.\begin{array}{l} a^{N-1} \equiv 1 \bmod N \\ \forall\, p \mid N-1,\ a^{\frac{N-1}{p}} \not\equiv 1 \bmod N \end{array}\right\} \Rightarrow N \text{ is prime}$$

Certificate: $(N, \{p \mid N-1\}, a) \Rightarrow$ **isPrime?**$\in$ **NP** (Pratt).

Thm. (**Pocklington**, 1914) Let s s.t. $s \mid N-1$

$$\left.\begin{array}{l} a^{N-1} \equiv 1 \bmod N \\ \forall\, q \text{ prime } \mid s, \gcd(a^{\frac{N-1}{q}} - 1, N) = 1 \end{array}\right\} \Rightarrow \forall\, p \mid N,\ p \equiv 1 \bmod s$$

Coro. $s > \sqrt{N} \Rightarrow N$ is prime.

Example

$$\begin{array}{rcrl} N_0 & = & 100003, & N_0 - 1 = 2 \times 3 \times 7 \times N_1, \\ N_1 & = & 2381, & N_1 - 1 = 2^2 \times 5 \times 7 \times 17 \end{array}$$

p	2	5	7	17
$3^{(N_1-1)/p} \bmod N_1$	2380	1347	1944	949

$\Rightarrow N_1$ is prime

$$s = N_1 > \sqrt{N_0}$$

$$2^{N_0-1} \equiv 1 \bmod N_0, \gcd(2^{(N_0-1)/N_1} - 1, N_0) = 1$$

$\Rightarrow N_0$ is prime

Rem. We get a recursive primality proof (DOWNRUN) of depth $O(\log N)$.

Mersenne numbers

Thm. (Lucas-Lehmer) $N = 2^m - 1$ is prime iff the sequence $L_0 = 4$, $L_{n+1} = L_n^2 - 2 \bmod N$ is s.t. $L_{m-2} \equiv 0 \bmod N$.

m	date	who	machine
1 257 787	1996	Slowinski & Gage	SGI/Cray T90
1 398 269	1996	Armengaud	Pentium 90 MHz
...	...	...	(42 days)
20 996 011	17/11/03	Shafer	2 GHz Pentium 4 Dell Dime
24 036 583	15/05/04	Finley	2.4 GHz Pentium 4
25 964 951	18/02/05	Nowak	2.4 GHz Pentium 4
30 402 457	24/12/05	Cooper/Boone	
32 582 657	02/09/06	Cooper/Boone	

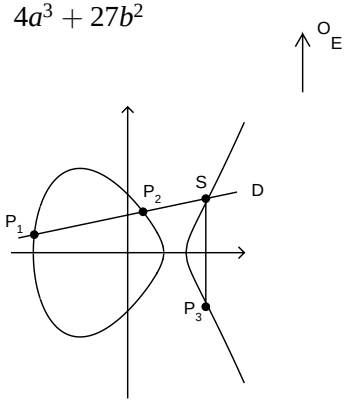
with GIMPS written by Woltman.
$M_{32\,582\,657}$ has 9 808 358 decimal digits.

III. Elliptic curves for primality proving

$$E : Y^2Z = X^3 + aXZ^2 + bZ^3$$

$$E(\mathbf{K}) = \{(X, Y, 1), Y^2 = X^3 + aX + b\} \cup \{(0, 1, 0)\}$$

$$j = 12^3 \frac{4a^3}{4a^3 + 27b^2}$$



Primality proving with elliptic curves

Use:

$$E(\mathbb{Z}/N\mathbb{Z}) = \{(x : y : z) \in \mathbb{P}^2(\mathbb{Z}/N\mathbb{Z}), y^2z = x^3 + axz^2 + bz^3\}$$

[with $\gcd(4a^3 + 27b^2, N) = 1$] + pseudo-addition.

Hasse's theorem: If N is prime, $\#E = N + 1 - t$ for some $|t| < 2\sqrt{N}$.

Thm. [Goldwasser & Kilian] Let m and s be two integers s.t. $s \mid m$, $E/(\mathbb{Z}/N\mathbb{Z})$ and $P \in E(\mathbb{Z}/N\mathbb{Z})$. If

$$[m]P = O_E$$
$$\forall\, q \text{ prime } \mid s, [m/q]P = (X : Y : Z), \gcd(Z, N) = 1$$

then $\forall\, p \mid N, \#E(\mathbb{Z}/p\mathbb{Z}) \equiv 0 (s)$.

Coro. $s > (\sqrt[4]{N} + 1)^2 \Longrightarrow N$ is prime.

Certificate : $(E, m, s, \{q \mid s\}, P)$.

Algorithm

Rationale: there should exist some curve with cardinality easy to factor in the interval.

In a nutshell: choose E at random until $\#E = 2q$ (computed with Schoof's algorithm), q prime, and then do the same for q.

Pb : $\{\#E = 2q\} \neq \emptyset$? Cf. $\mathbb{P} \cap [x, x + x^{0.525}] \neq \emptyset$ (Baker *et al.*).

Thm. GK terminates in (random) time $O((\log N)^9)$ on average for all primes $\leq x$, except for those belonging to $\mathcal{E}(x)$ of cardinality

$$\#\mathcal{E}(x) \ll \frac{x/\log x}{2^{2\frac{\log\log x}{\log\log\log x}}}.$$

Thm. (Adleman & Huang, 1986) **isPrime?$\in$ RP**.

Idea: use genus 2 curves, for which the Hasse-Weil is large enough.

A parenthesis: SEA

Schoof's algorithm: designed around 1985. Made efficient by Atkin, Elkies, etc. $\Rightarrow$ SEA.

Latest SEA record: AEnge + FM, 2006.
Timings on an AMD 64 Processor 3400+ (2.4GHz), with NTL, (excluding the time for computing modular equations):

what	500dd	1000dd	1500dd	2005dd	2500dd
Total	10h	180h	77d	195d	404d

Fine points: modular equations of large index ≥ 5000 (AEnge); fast eigenvalue computation (PGaudry/FM; PMihăilescu/FM/ÉSchost); fast isogeny computations (ABostan/FM/BSalvy/ÉSchost).

Rem. For p small, situation is complety different, ask Pierrick!

Conclusion for primality proving: seems hopeless!

IV. (fast)ECPP

Practical primality (Atkin)

Replace random curves by elliptic curves with complex multiplication by an order in an imaginary quadratic field $\mathbf{K} = \mathbb{Q}(\sqrt{-D})$ for which $4N = U^2 + DV^2$.

A suitable E having $N + 1 - U$ points.

Thm. (Katre's version) If $p = x^2 + 4y^2$ with $x \equiv 1 \bmod 4$ and $a \not\equiv 0 \bmod p$, then $E : Y^2 = X^3 + aX$ has cardinality

$$p + 1 - \begin{cases} \pm 2x & \text{if } (a/p)_4 = \pm 1, \\ -4y & \text{otherwise with } y \text{ s.t. } 2y(a/p)_4 = x. \end{cases}$$

For larger D, E is computed using a degree $h(-D)$-polynomial (class polynomial).

Algorithms for computing $H_D(X)$

- good old floating point complex numbers (. . . , AOLAtkin+FM, AEnge+FM, . . .):
  - naive way: h evaluations of j, followed by `PolyFromRoots` $O()$;
  - $\tilde{O}(h^2)$ (Enge04; see also R. Dupont for the fast evaluation of η).
- rather new p-adic methods (Couveignes/Henocq – ANTS V, Bröker/Stevenhagen, Lercier/Riboulet-Deyris): $\tilde{O}(h^2)$.

ECPP = Pocklington + CM curves

[**Step 1.**] find $-D$ s.t. $4N = U^2 + DV^2$ and $m = N + 1 \mp U = 2N'$ with N' prp.

[**Step 2.**] Build $\overline{E}/\overline{\mathbb{Q}}$ having CM by $\mathcal{O}_K$ (independent of N).

[**Step 3.**] Reduce $\overline{E}$ modulo N to get E.

[**Step 4.**] Find $P \neq O_E$ on E s.t. $[N']P = O_E$. If a factor f of N is found, return ("composite", f). If such a point cannot be found in reasonable time, return ("I don't know", "no P found").

[**Step 5.**] N is "prime"; $N \leftarrow N'$ and goto Step 1.

Conj. ECPP costs $\tilde{O}((\log N)^5)$.

fastECPP

Genesis:
• A. K. Lenstra & H. W. Lenstra, Jr.: Algorithms in number theory in HANDBOOK OF THEORETICAL COMPUTER SCIENCE, 1990: sketch analysis of ECPP and cite Shallit's improvement.
• During FM's algorithmic trip for Bourbaki: dig the paper (fall 2002).
• Exchange of emails with Dan Bernstein following his improvement on AKS, discussing ECPP alleged complexity (spring 2003).
• Why not try in FM's new mpi ECPP implementation? (spring 2003 ff).

Scaling: fastECPP

Main cost of ECPP: compute $\sqrt{-D}$ for many D's until one is good. Need $\tilde{O}((\log N)^2)$ for a cost of $\tilde{O}((\log N)^4)$.

Shallit's trick: precompute a factor basis of $\sqrt{q_1}, \ldots, \sqrt{q_r}$ and build discriminants as products of the q_i's. We still get $\tilde{O}((\log N)^2)$ discriminants, but with $r = O(\log N)$ square-root computations. The total cost drops to $\tilde{O}((\log N)^4)$.

Main cost now: many PRP tests, followed by building of class polynomials of degree $h(-D) = O(\sqrt{D})$ (+ finding roots).

The current record

$$N = ((((((((((2^3+3)^3+30)^3+6)^3+80)^3+12)^3+450)^3 \\ +894)^3+3636)^3+70756)^3+97220$$

is prime (20,562 decimal digits).
See http://www.cs.uwaterloo.ca/journals/JIS/VOL8/Caldwell/caldwell78.html .

- The computations were started on 32-bit machines (Sep-Oct 2005), and finished on nine 64-bit bi-processors (Feb-June 2006).
- Cumulated timings are given w.r.t. AMD Opteron(tm) Processor 250 at 2.39 GHz.
  1st phase: 1900 days (396 for sqrt; 384 for Cornacchia; 1353 for PRP tests).
  2nd phase: 319 days (8 days for building all H_D's; 277 for solving $H_D \bmod p$).
- The certificate is 48Mb (compressed). It takes 10 days to check the 1765 proof steps on a single processor.

Andreas Enge strikes!

Compute $H_D[w](X)$ in time $\tilde{O}(D)$ for small invariants w, using fast evaluation of η-functions using floating point numbers; faster using AGM (R. Dupont).

Largest examples for 15,071dd:

D	h
$294,699,719$	24444
$2,261,873,887$	20544
$4,744,754,740$	15360
$8,581,560,955$	12160

Records: $D = 357,116,231$, smallest discriminant with $h = 40,000$ (thanks to S. Louboutin's algorithm), time=14,000 sec on an Opteron 2.4GHz. Size of the polynomial $\approx 1Gb$ (uncompressed), height $= 88,000$ bits.

Where's the problem, then? Memory.

Programs

- First program: PASCAL (1987).
- Thesis: LELISP (1987–1990).
- V3.4.1: C with `BigNum`, put in anon ftp in 1991.
- As time goes by, more work is done.
- Intermediary version given to MAGMA.
- V5.6.1: program cleaned from top to bottom; only one binary put in anon ftp (for DecAlpha).
- V6.4.5: after ANTS-III, since 1991; most recent (available) one V6.4.5; intermediate one in MAGMA; 0.39s for 256 bits, 3.42s for 512, 49.45s for 1024 (Pentium 450 MHz).
- V11.0.5: still being in development.

Lessons from supporting ECPP these last 15 years

- 1992: 1505dd; 2003: 10,000dd (Franke, Kleinjung, Wirth); 2004: 15,071dd.
- $\Rightarrow$ hegemony is bad: stimulation from the Wind* world and its crazy prime provers (cf. PRIMO of Marcel Martin).
- One lesson: more and more powerful computers + GMP $\Rightarrow$ fast methods begin to be fast in the real life (really fast available multiplication of integers, fast algorithms from computer algebra, etc.) $\Rightarrow$ larger and larger (D, h) the key to fastECPP.
- Strategies for large numbers can be against intuition.

V. Conclusions

Combining the tests?

- **AKS with ECPP:** if $N-1$ is not convenient, use ECPP to find $N' \in]N+1-2\sqrt{N}, N+1+2\sqrt{N}[$ which is convenient. Heuristic complexity $\tilde{O}((\log N)^4)$ (?).
- **ECPP and JS:**
  - dual elliptic pseudoprimes (PM+FM).
  - $N \equiv 1 \bmod t(N)$ are easier. À la Cheng: Find a suitable N' s.t. $N' \equiv 1 \bmod t$ and use JS. But the last phase dominates everything...!
- **AKS and JS:** ??? They seem to share a lot of properties.

Conclusions

Many answers:

- easy to understand/implement: ADSMR;
- fast, even if not proven: Jacobi, ECPP, BM?;
- certificate: ECPP;
- available: Jacobi sums in **`pari`**, ECPP (MAGMA);
- deterministic polynomial: AKS, HWL+CP.

Open problem for instance: improve AKS to make it usable!