Fully Deterministic ECM

Iram Chelli

LORIA (CNRS) - CACAO
Supervisor: P. Zimmermann

September 23, 2009

Introduction

- The Elliptic Curve Method (ECM) is currently the best-known general-purpose factorization algorithm for finding "small" prime factors in numbers having more than 200 digits.
- Heuristic expected running time to find a factor p of a number n $O(L(p)^{\sqrt{2}+o(1)}M(\log(n)))$ where $L(p) = e^{\sqrt{\log(p)\log(\log(p))}}$ and $M(\log(n))$ is the complexity of multiplications modulo n.
- The complexity of ECM is dominated by the size of the smallest factor p of n rather than the size of the number n to be factored.

Introduction

- ECM is a randomized algorithm. Monte-Carlo algorithm.
- In some applications, one would like to be able to be certain to remove all prime factors up to a given bound B: this is what we call a *deterministic* factoring algorithm.
- Ex: the relation collection phase of the Number Field Sieve (NFS) to help identify smooth integers.

Introduction

- The General Number Field Sieve (GNFS) is the best currently known algorithm for factoring hard integers, i.e., integers that have no small prime factors or other properties that would make them easy to factor.
- In particular, the modulus of keys used in the RSA asymmetric encryption system is chosen to be a hard composite integer.
- It is a general factorization algorithm: its run-time depends only on the size of the input number but not on the size or other properties of the prime factors.
- As factorization of the modulus reveals the private key, the cost of factoring the modulus with GNFS serves as an upper bound on the cost of breaking RSA.

Table of contents

Contribution

- We present a FDECM algorithm allowing to remove with certainty all prime factors less than $B = 2^{32}$ from a composite input number n.
- Designed to be the most implementation-independent possible.
- 203280221 primes below B : trial-division and GCD both are unpractical.

Contribution

- With traditional ECM it costs a few hundred random curves. It is quite fast but no guarantee that all factors will be found.
- With FDECM: even faster!
- We have optimized it to under a hundred well-chosen elliptic curves, which can be very fast in an optimized ECM implementation with optimized B_1 and B_2 smoothness bounds.
- This is the first detailed description of a fully deterministic ECM algorithm.

Curve equations - Definition

- Elliptic curve equation in Weierstrass form: $y^2 = x^3 + Ax + B$.
- The curve is defined as the set of points whose coordinates verify the equation, plus a point at "infinity".
- Elliptic curves in Weierstrass form are not very efficient in terms of computational cost.
- Elliptic curve equation in Montgomery's form: $by^2 = x^3 + a\,x^2 + x$.
- Montgomery's form can be obtained from Weierstrass form by the change of variables: $X = (3x + a)/3b$, $Y = y/b$, $A = (3 - a^2)/3b^2$, $B = (2a^3/9 - a)/3b^2$.

Elliptic curve defined by $y^2 = x^3 + 1$.

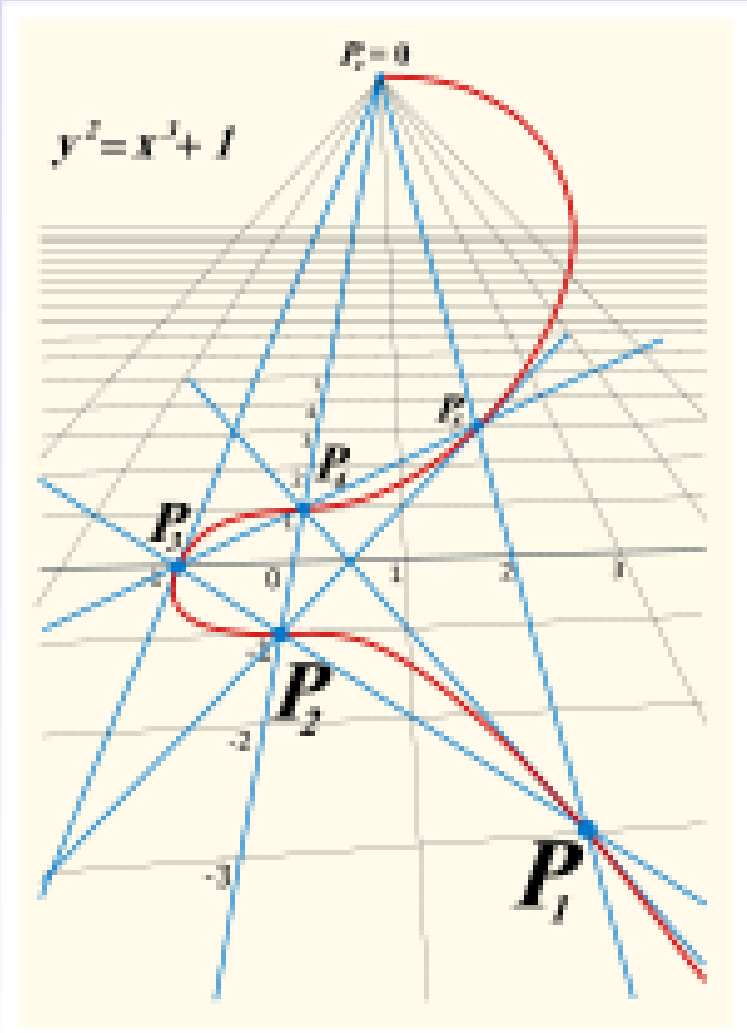


Figure: Infini

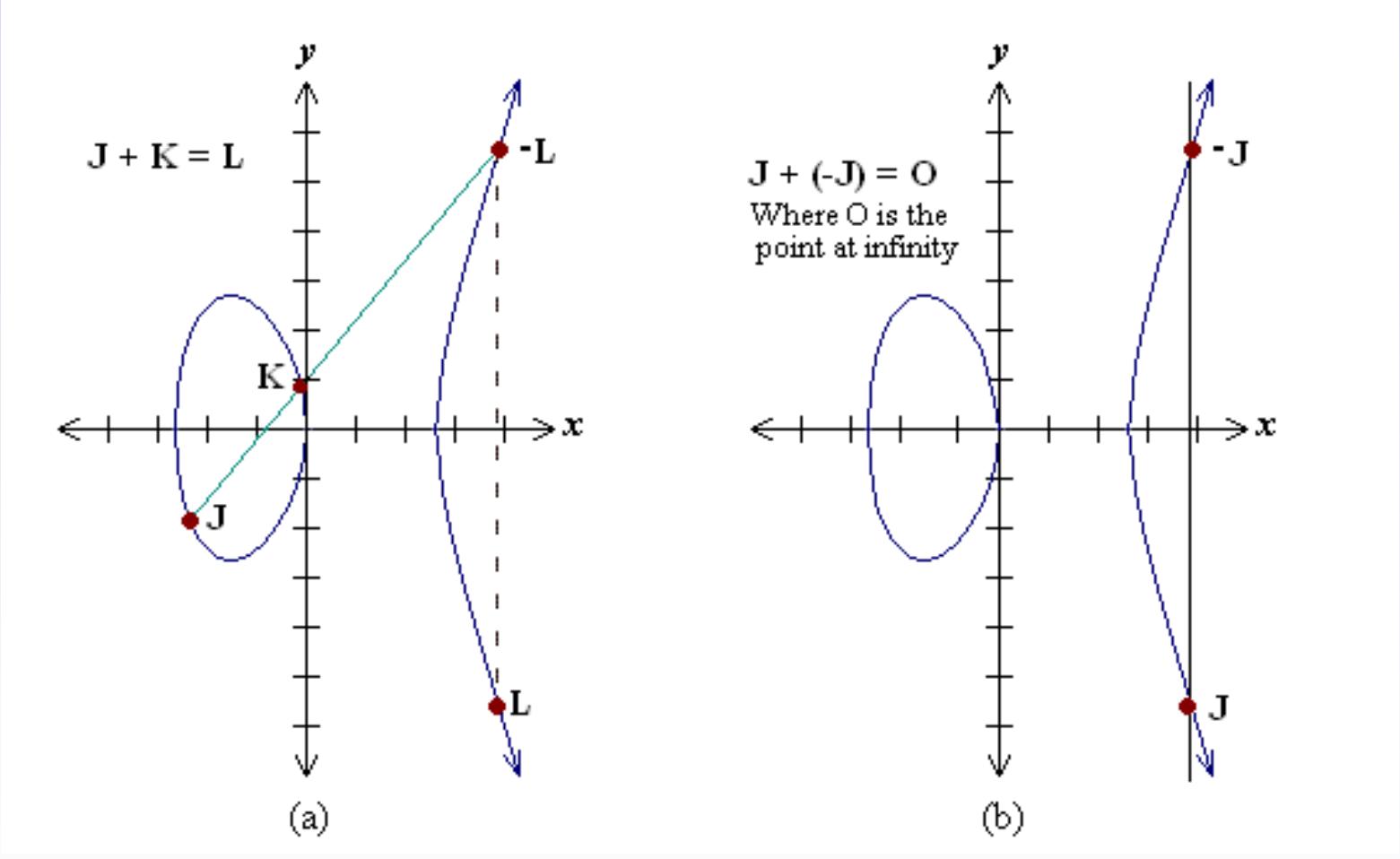
Elliptic curves in projective coordinates

- In homogenous form we have the equation $by^2z = x^3 + a\,x^2z + xz^2$.
- Points on the curve are represented in projective coordinates $(x : y : z)$, disregarding the y-coordinate simply noting $P = (x :: z)$.
- Avoid inversions which are computationally costly.
- A point $(x : y : z)$ in projective coordinates, $z \neq 0$, can be converted to $(x/z, y/z)$ in affine coordinates and conversely.

Group structure

- The set $E(K)$ of points of an elliptic curve E has an abelian group structure.
- Point O at infinity is the neutral element
- Inverse of a point $P = (x, y)$ of E is $-P = (x, -y) \in E$.
- Group law can be observed geometrically: geometrical construction of the sum of two points on an elliptic curve.

Geometrical group law view



Point doubling and addition laws

- In order to work on elliptic curves, we need to be able to compute point additions.
- Addition law for Curves in Montgomery's form require that the difference of the two points be known.
- For an elliptic curve in Montgomery's form: Let $P = (x_P :: z_P)$ and $Q = (x_Q :: z_Q)$ be two distinct points and let $P - Q = (x_{P-Q} :: z_{P-Q})$ be their difference.
- Their sum is: $x_{P+Q} = 4z_{P-Q} * (x_P x_Q - z_P z_Q)^2$, $z_{P+Q} = 4x_{P-Q} * (x_P z_Q - z_P x_Q)^2$.
- The doubling is: $x_{2P} = (x_P^2 - z_P^2)^2$, $z_{2P} = 4x_P z_P[(x_P - z_P)^2 + 4d\, x_P z_P]$, where $d = (a+2)/4$ and a is the coefficient in the elliptic curve equation.

Double and add

- The ECM algorithm needs computation of point kP from a given integer k and point P.
- Except for the doubling, we have no "multiplication by an integer" formula.
- Point nP is computed by means of repeated addings and doublings from the binary expansion of integer n.
- $k = 100 = (1100100)_2$

$$100P = 2(2(P + 2(2(2(P + 2P))))).$$

Algorithm DOUBLE AND ADD (Right to Left):

INPUT: Point P from E and $n \geqslant 1$
OUTPUT: Point $R = nP$

1. Set $Q = P$ and $R = O$ (point at infinity)
2. loop while $n > 0$
   if $n = 1 \pmod 2$ set $R = R + Q$
   set $Q = 2Q$ and $n = \lfloor n/2 \rfloor$
3. return R (which equals nP)

Lucas chains

- Special case of addition chains in which the sum of two terms can appear if and only if their difference also appears.
- These chains are used when working in Montgomery's form.
- Montgomery's PRAC is an heuristic algorithm designed to find short Lucas chains.
- Non-unicity: Many different Lucas chains exist.

The ECM method

- The number to be factored being n, we work over $\mathbb{Z}/n\mathbb{Z}$ as if it were a field.
- The only operation that might fail is ring inversion which is calculated using the Euclidean algorithm.
- If an inversion fails, then n is not coprime with that number and we find a factor of n by computing their gcd.

Smoothness criteria

- Let n be a positive integer, with prime factorization: $n = \prod_{i=1}^{m} p_i^{\alpha_i}$ and let B be a positive integer.
- n is **B-smooth** if all of its prime factors p_i verify $p_i \leqslant B$.
- It is **B-powersmooth** if for all i we have, $p_i^{\alpha_i} \leqslant B$.
- n is **(B_1, B_2)-smooth** if it is B_1-powersmooth for all but it's largest prime factor p_m, and we have $\alpha_m = 1$ and $p_m \leqslant B_2$.
- $1108233 = 3^2 * 7^3 * 359$ is (7^3,359)-smooth.

ECM algorithm

- We choose a random nonsingular elliptic curve E over $\mathbb{Z}/n\mathbb{Z}$, where n is the number to factor, and a point P on it.
- We seek a positive integer k such that $[k]P = \mathcal{O}$ modulo an (unknown) prime divisor p of n but not modulo n.
- Then, in the computation of $Q = [k]P$ mod n the attempted inversion of an element not coprime to n reveals the factor p by simple computation of $\gcd(x_Q, n)$.
- For this to happen k needs to be a multiple of $g_p = \#\mathcal{E}(\mathbb{F}_p)$ which is also unknown.

Stage 1

- Stage 1 will compute $Q = [k]P$ where $k = \prod_{\pi \leqslant B_1} \pi^{\lfloor \log(B_1)/\log(\pi) \rfloor} = \mathrm{lcm}\,(1, 2, \ldots, B_1)$.
- It is successful if the order g of the curve is B_1-powersmooth.
- The main cost of stage 1 is the elliptic curve arithmetic.

Stage 2

- Stage 2 is an improvement of the initial ECM algorithm.
- When the order g is not B_1-smooth because of a single prime factor q above B_1.
- A bound $B_2 > B_1$ is set so that we have a chance that this prime factor q is below $B2$.
- It is successful if g is (B_1, B_2)-smooth. Prime q will then be found in stage 2.

Brent-Suyama's parametrization

- It is simple and of widespread use: allows for easy reproduction of the results obtained.
- This elliptic curve parametrization uses a single integer (or rational value) $\sigma > 5$.
- From this parameter we then compute:

  $u = \sigma^2 - 5$, $v = 4\sigma$,
  $x_0 = u^3 \pmod{n}$, $z_0 = v^3 \pmod{n}$,
  $a = (v-u)^3(3u+v)/(4u^3v) - 2 \pmod{n}$, $b = u/z_0$ and,
  $y_0 = (\sigma^2 - 1)(\sigma^2 - 25)(\sigma^4 - 25)$.
- It yields an elliptic curve in Montgomery's form:
  $by^2z = x^3 + ax^2z + xz^2$.

Brent-Suyama's parametrization

- Suyama's and Montgomery's form parametrizations both ensure 12 divides g over $\mathbb{F}_q$.
- This known factor increases the probability of success of ECM for fixed given parameters.

Using ECM with prime input numbers

- Normally, ECM is run on a *composite* input number n that we are trying to factor.
- Here what we want to determine is whether a given prime p will be found by ECM when it is used on an input number n such that $p|n$, but the cofactor is undetermined.
- We use ECM in an unusual way: input number p is prime, and ECM returns p if $g = \#\mathcal{E}(\mathbb{F}_p)$ is $(B1, B2)$-smooth and fails if not.
- If for a given σ a prime p is found by ECM with given $(B1, B2)$, then it will always be found when the same curve (with same parameters) is run on a number n such that $p|n$!

The influence of B_1, B_2 bounds

- The choice of B_1 and B_2 bounds impacts both the running time of ECM and the number of primes found by the algorithm.
- Too high: ECM gets slower, higher probability to return composites or the input number itself.
- Too low fewer primes are found: we will need to run more curves.
- We want to minimize the running time while simultaneously maximizing the number of primes found.

B_1, B_2 values minimizing time over found primes ratio.

- 2^{27}: $B_1 = 140$, $B_2 = 7600$, time(sec.): 29.59 hits: 247667 Ratio: 119μs.
- 2^{28}: $B_1 = 140$, $B_2 = 9200$, time(sec.): 31.44 hits: 208562 Ratio: 151μs.
- 2^{29}: $B_1 = 220$, $B_2 = 10400$, time(sec.): 40.6 hits: 211617 Ratio: 192μs.
- 2^{30}: $B_1 = 220$, $B_2 = 8800$, time(sec.): 38.71 hits: 163260 Ratio: 237μs.
- 2^{31}: $B_1 = 260$, $B_2 = 11600$, time(sec.):48.63 hits: 161424 Ratio: 301μs.
- 2^{32}: $B_1 = 260$, $B_2 = 11600$, time(sec.):48.83 hits: 130144 Ratio: 375μs.

Methodology: Making ECM implementation-independant

- ECM factorisation program “ecm_check” may find primes p for which $E(p)$ is not B1-B2 smooth.
- Ex: For $B = 2^{27}$, $B_1 = 315$ and $B_2 = 5355, 117564$ additional primes found by “ecm_check”. This is about 4% extra primes.
- We check the σ-chains constructed with ecm-check with Magma considering smoothness of the curve order $g = \#E(\mathbb{F}_p)$ for every prime found p to make sure it will indeed be found by any relatively “standard” ECM implementation.

Methodology: Making ECM implementation-independant

- Improvement: considering starting point order instead of the curve order. This is a divisor of the curve order: it has significantly higher probability to be smooth with the same B_1, B_2 parameters.
- Without sacrificing to generality because the starting point P is fully determined by the choice of σ.
- Let g the order of the starting point P. The multiplier e is precomputed from B_1.
- End of stage one: point eP has order $g' = g/(g, e)$.

Methodology: Making ECM implementation-independant

- If $g' = 1$ then stage one was successful.
- Otherwise if g' is prime and $B1 < g' \leqslant B2$ then it will be found in stage two.
- Implemented as a Magma script: outputs primes for which the starting point order does not verify any of the above propreties.

Finding all primes up to $B = 2^{32}$.

- We work in intervals of same bitlength which are more convenient to handle.
- After the first one, the size of each interval is approximately the double of the preceding.
- FDECM running time then also would approximately double for each interval.
- The cost-per-prime is not fixed: increases with the size of the input factor and size of $B_1 - B_2$ smoothness bounds.

Building σ-chains for sets of non-consecutive primes

- We now consider the case where the primes we are looking for are not consecutive.
- Primes for which an algebraic polynomial $f(x)$ has a root mod p, which is a subset of all possible primes.
- It is of interest in NFS: given an algebraic polynomial $f(x)$, the primes appearing in the factorization of $f(a/b)$ are those for which $f(x)$ has a root mod p.
- Ex: The polynomial that was used in the factoring of RSA200 by GNFS.

Practical case: The RSA200 subset of primes

- The polynomial is : $f(x) = X5 * x^5 + X4 * x^4 + \ldots + X0$.
- The coefficients are :
  $X5 = 374029011720$,
  $X4 = 27110656377956301 18$,
  $X3 = 194000719431775138658927 14$,
  $X2 = -338034706092024130946804623 60399$,
  $X1 = -120887311888241287002580512992469303610$,
  $X0 = 3876720300079932118978295952993877119517 0960$.
- This polynomial has a root for a subset of 128740271 elements of the set of 203280221 primes less than 2^{32}.

Outstanding σ values

- $\sigma = 11$ appears to perform noticeably better than random curves for finding primes.
- Other curves of the infinite family of $\sigma = 11$ perform similarly with an above-average performance: they have rational sigmas.
- Another infinite family of curves, those for $\sigma = 9/4$ appear to show the same type of behaviour.
- First family appears to favour primes p congruent to 1 mod 4 whereas the second favours primes in the congruence class 3 mod 4.

Results achieved

- A 124-element integer σ-chain which allows to find all primes $p < 2^{32}$.
- With optimized 26-element rational chain: All primes up to 2^{32} are now found with 99 curves, and 97 for the RSA200 subset.

Switching from ECM to gcd or trial division and when

- Last third of the curves invariantly accounts for a very small proportion ,i.e., less than 1/1000 th, of the primes found.
- Less costly to take a gcd of the number we want to factor with the product of those primes or simply trial-divide by the remaining primes after ECM has been performed.
- Which is faster? when is it best to switch from ECM to gcd or trial division?
- Depends on the size of the input number to factor: we have run tests on 100, 1000, 10000-digit primes (GMP).
- Testing platform: Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40GHz.

Conclusion

- We have successfully built two optimized σ-chains below 100 elements in length. The first one guarantees to find all primes below 2^{32} and has 99 elements and the second one guarantees to find all primes of the subset RSA200 and has 97 elements.
- FDECM algorithm's running time is approximately: 0.41 seconds for a 100-digit input number, 7.51 seconds for a 1000-digit input number, and 294 seconds for a 10000-digit input number, which is extremely small comparatively to trial-division for which times are 8.09 seconds, 56.3 seconds, and 5549 seconds respectively.
- The σ-chains can still be further-optimised to be reduced in length, since there exists in theory a huge super-σ that would find all primes below 2^{32} in a single curve!