18.783 Elliptic Curves
Lecture 10

Andrew Sutherland

October 17, 2023

Lecture 9 recap: generic DLP bounds

Pohlig-Hellman: $O(n \log n + n\sqrt{p})$, where $n = \log N$, largest prime $p|N$.

Baby-steps giant-steps: $(2 + o(1))\sqrt{N}$ time, $(2 + o(1))\sqrt{N}$ space.

Pollard-ρ (Las Vegas): $(\sqrt{\pi/2} + o(1))\sqrt{N}$ expected time, $O(\log N)$ space.

Theorem (Shoup)

Let G be cyclic group of prime order N.

- *Every deterministic generic algorithm for the discrete logarithm problem in G uses at least $(\sqrt{2} + o(1))\sqrt{N}$ group operations.*
- *Every Las Vegas generic algorithm for the discrete logarithm problem in G expects to use at least $(\sqrt{2}/2 + o(1))\sqrt{N}$ group operations.*

Shoup's lower bounds match the best upper bounds to within a factor of 2.

Index calculus: a non-generic algorithm for the DLP

Let $G = \langle\alpha\rangle = (\mathbb{Z}/p\mathbb{Z})^\times$ and identify G with $[1, N] \cap \mathbb{Z}$, where $N = \#G = p - 1$. For $e \in \mathbb{Z}$ we can use the prime factorization $\alpha^e\beta^{-1} = \prod_i p_i^{e_i}$ to obtain a relation

$$e_1 \log_\alpha p_1 + \cdots + e_b \log_\alpha p_b + \log_\alpha \beta = e. \tag{1}$$

which would allow us to compute $\log_\alpha \beta$ if we knew the values of $\log_\alpha p_i$.

Our plan: Pick a smallish set of primes $S = \{p : p \leq B\} = p_1, \ldots p_b$ (the factor base), and generate relations as in (1) by picking random $e \in [1, N]$ and attempting to factor $\alpha^e\beta^{-1}$ over our factor base (e.g. by trial division, or something more clever).

How we win: Collect relations that uniquely determine $\log_\alpha p_1, \ldots, \log_\alpha p_b, \log_\alpha \beta$ and use linear algebra over the ring $\mathbb{Z}/N\mathbb{Z}$ to solve the system for $\log_\alpha \beta$.

When we expect to win: After about $\pi(B) \cdot N/\psi(N, B)$ attempts, where $\psi(N, B)$ is the number of B-smooth integers in $[1, N]$, those with all prime factors less than B.

Optimizing the smoothness bound B

Theorem (Canfield–Erdős-Pomerance)

As $u, x \to \infty$ *with* $u < (1-\epsilon)\log x/\log\log x$ *we have* $\psi(x, x^{1/u}) = xu^{-u+o(u)}$.

With trial division factoring takes $O(\pi(B)\mathsf{M}(\log N))$ time and we expect to need

$$O(\pi(B)u^u\pi(B)\mathsf{M}(\log N)) \approx B^2u^u = N^{2/u}u^u$$

time to get enough relations, where $u := \log N/\log B$ so that $N^{1/u} = B$.

To minimize $f(u) := \log(N^{2/u}u^u) = \frac{2}{u}\log N + u\log u$ we want to choose u so that

$$f'(u) = -2u^{-2}\log N + 2(uN)^{-1} + \log u + 1 = 0.$$

Ignoring $O(1)$ terms, we want $u^2\log u \approx 2\log N$, meaning $u \approx 2\sqrt{\log N/\log\log N}$.

Expected running time of our index calculus algorithm

Our choice of $u \approx 2\sqrt{\log N / \log\log N}$ yields the smoothness bound

$$B = N^{1/u} = \exp(u^{-1}\log N) = \exp(1/2\sqrt{\log N \log\log N}) = L_N[1/2, 1/2],$$

where we have used the standard subexponential asymptotic notation

$$L_N[a, c] := \exp((c + o(1))(\log N)^a(\log\log N)^{1-a}),$$

interpolating $L_N[0, c] = (\log N)^{c+o(1)}$ (polynomial), $L_N[1, c] = N^{c+o(1)}$ (exponential).

Assuming the linear algebra is negligible (it is), the total expected time is

$$B^2 u^u = L_N[1/2, 1/2]^2 \cdot L_N[1/2, 1] = L_N[1/2, 2].$$

With ECM, smoothness testing becomes negligible and we can achieve $L_N[1/2, \sqrt{2}]$.
More sophisticated techniques (NFS) heuristically yield $L_N[1/3, (64/9)^{1/3}]$.

Current state of the art

For finite fields $\mathbb{F}_{p^n} \simeq \mathbb{F}_p[x]/(f)$ the function field sieve uses a factor base of low degree polynomials in $\mathbb{F}_p[x]$ representing elements of $\mathbb{F}_{p^n}$ to obtain an $L_N[1/3, c]$ bound.

In 2013 Joux found an $L_N[1/4, c]$-time algorithm for $\mathbb{F}_{p^n}^\times$ for suitable p and n.

Joux and collaborators improved these techniques rapidly, eventually leading to a DLP algorithm for $\mathbb{F}_{p^n}^\times$ with $p = O(n)$ that runs in time $n^{\log n}$, which is better than $L_N[\epsilon, c]$ for any $\epsilon, c > 0$ (quasi-polynomial time).

Instant poll

For $E(\mathbb{F}_p)$ the only DLP algorithms we know are generic. For prime fields $\mathbb{F}_q$ we have a subexponential-time algorithm for DLP in $\mathbb{F}_q^\times$, and for suitable prime powers q we have a quasi-polynomial time algorithm. Based on this information, what do you think the current records are for solving DLP in $E(\mathbb{F}_p)$, $\mathbb{F}_p^\times$, $\mathbb{F}_q^\times$?

A. 64 bits, 192 bits, 4,096 bits

B. 117 bits, 795 bits, 30,750 bits

C. 161 bits, 2,011 bits, 86,117 bits

D. No idea, “subexponential” and “quasi-polynomial” mean nothing to me.

E. No idea, it really depends on the constant factors.

The Pollard $p-1$ factorization method

Algorithm

Given an integer N and a smoothness bound B, attempt to factor N as follows:

1. Pick a random integer $a \in [1, N-1]$; if $\gcd(a, N) = d \neq 1$ return $(d, N/d)$.
2. Set $b = a$ and for increasing primes $\ell \leq B$:
   - 2.1 Replace b with b^{ℓ^e} where $\ell^{e-1} < N \leq \ell^e$. If $b = 1$ then give up.
   - 2.2 if $\gcd(b-1, N) = d \neq 1$ then return $(d, N/d)$.

Theorem

Let $p, q|N$ be primes. If $p-1$ is ℓ-smooth but $q-1$ is not for some prime $\ell \leq B$ then the algorithm succeeds with probability at least $1 - {}^{1}\!/_{(\ell+1)}$.

Proof. When we reach ℓ in 2.2 we will have $b = a^m \equiv 1 \bmod p$, since $(p-1)|m$. But some prime $\ell' > \ell$ divides $q-1$ but not m, so $\Pr[b \not\equiv 1 \bmod q] \geq 1 - {}^{1}\!/_{(\ell+1)}$.

Robbing a random bank

If $\#(\mathbb{Z}/N\mathbb{Z})^\times$ has a B-smooth prime factor then Pollard's algorithm is very likely to succeed, but this is unlikely for any particular $N = pq$ a product of two large primes.

For random pq in $[N, 2N]$ we expect the probability is u^{-u}, where $u = \log N / \log B$. That is small, but only subexponentially so; if we try u^u random pq we should succeed.

If we let $u = \sqrt{2 \log N / \log\log N}$, then $B = N^{1/u} = L_N[1/2, 1/\sqrt{2}]$ and we should expect to factor a random pq in $[N, 2N]$ in time $N^{1/u} u^u = L_N[1/2, \sqrt{2}]$.

Key point: By varying pq we vary the group $(\mathbb{Z}/pq\mathbb{Z})^\times$. But what if pq is fixed?

Lenstra: We can vary the group by picking a random elliptic curve "modulo pq".

The elliptic curve factorization method (ECM)

Algorithm

Given $N \in \mathbb{Z}$, a smoothness bound B, and a prime bound M, attempt to factor N:

1. Pick random $a, x_0, y_0 \in [0, N-1]$ and set $b = y_0^2 - x_0^3 - ax_0$.
2. if $d = \gcd(4a^3 + 27b^2, N) \neq 1$ return $(d, N/d)$ if $d < N$ but give up if $d = N$.
3. Let $Q = (x_0 : y_0 : 1)$ and for increasing primes $\ell \leq B$:
   - **3.1** Replace Q with $\ell^e Q \bmod N$ where $\ell^{e-1} \leq (\sqrt{M}+1)^2 < \ell^e$. Give up if $Q_z = 0$.
   - **3.2** If $d = \gcd(Q_z, N) \neq 1$ then return $(d, N/d)$.

Theorem

Let P_1 and P_2 be the reductions of $(x_0 : y_0 : 1)$ modulo distinct $p_1, p_2 | N$ with $p_1 \leq M$. If $|P_1|$ is ℓ-smooth and $|P_2|$ is not for some $\ell \leq B$ then the algorithm succeeds in 3.2.

Heuristic complexity of ECM

The Hasse interval $[p+1-2\sqrt{p},\ p+1+2\sqrt{p}]$ is too narrow to apply CEP bounds. We can prove $\#E(\mathbb{F}_p) \in [p+1-\sqrt{p},\ p+1+\sqrt{p}]$ with probability at least 1/2, and roughly uniformly distributed over this interval (Sato-Tate on average).

If we heuristically assume integers in $[p+1-\sqrt{p},\ p+1+\sqrt{p}]$ are as likely to be smooth as integers in $[p, 2p]$ we can compute the optimal choice of $B = L_M[1/2, 1/\sqrt{2}]$. We generally don't know what M should be, so start small and double it. This yields

$$L_p[1/2, \sqrt{2}]\mathsf{M}(\log N),$$

where p is the smallest prime factor of N. We can then use ECM to test whether a given integer N is $L_N[1/2, c]$-smooth in expected time

$$\begin{aligned} L_{L_N[1/2,c]}\left[1/2, \sqrt{2}\right] &\approx \exp\left(\sqrt{2\log(\exp(c\sqrt{\log N \log\log N})\log\log(\exp(c\sqrt{\log N \log\log N})}\right) \\ &= L_N[1/4, \sqrt{c}]. \end{aligned}$$

Montgomery curves

Definition

A Montgomery curve is an elliptic curve defined by an equation of the form

$$By^2 = x^3 + Ax^2 + x$$

with $B \neq 0$ and $A \neq \pm 2$. Put $v = Bx + AB/3$, $w = B^2y$, $x = u/B$, $y = w/B^2$ to get

$$w^2 = v^2 + (B^2 - A^2B^2/3)v + (2A^3B^3/27 - AB^3/3),$$

an elliptic curve in Weierstrass form.

To compute $(x_3, y_3) = (x_1, y_1) + (x_2, y_2)$ we use

$$x_3 = Bm^2 - (A + x_1 + x_2), \qquad y_3 = m(x_1 - x_3) - y_1,$$

where $m = (y_2 - y_1)/(x_1 - x_2)$ or $m = (3x_1^2 + 2Ax_1 + 1)/(2By_1)$.

Montgomery ladder

Let $(x_4, y_4) = (x_1, y_1) - (x_2, y_2)$. In projective coordinates we have

$$x_3 = z_4 \left[(x_1 - z_1)(x_2 + z_2) + (x_1 + z_1)(x_2 - z_2)\right]^2,$$
$$z_3 = x_4 \left[(x_1 - z_1)(x_2 + z_2) - (x_1 + z_1)(x_2 - z_2)\right]^2.$$

This allow us to compute $P_1 + P_2$ using 6 multiplications, assuming we know $P_1 - P_2$.

Algorithm (Montgomery Ladder)

Input: A point $P = (x_1 : z_1)$ on a Montgomery curve and a positive integer m.
Output: The point $mP = (x_m : z_m)$.

1. Let $m = \sum_{i=0}^{k} m_i 2^i$ be the binary representation of m.
2. Set $Q[0] = P$ and compute $Q[1] = 2P$ (note that $P = Q[1] - Q[0]$).
3. For $i = k - 1$ down to 0: $Q[1 - m_i] \leftarrow Q[1] + Q[0]$, $Q[m_i] \leftarrow 2Q[0]$.
4. Return $Q[0]$.