Hyperelliptic Discrete Logarithms

P. Gaudry

gaudry@lix.polytechnique.fr

http://www.lix.polytechnique.fr/Labo/Pierrick.Gaudry/

ECC 2000, Essen, October 4, 2000.

Overview

- Cryptographical motivation
- Hyperelliptic curves and their Jacobians
- History of hyperelliptic index calculus methods
- Discrete log algorithm for “small” genus
- Improvements, tricks, practical considerations
- Conclusion, future work

Cryptographical motivation

Public Key Cryptosystems: what key size?

Maximal number of operations, in a reasonable amount of time: 2^{50}–2^{60}.

Secure means: unbreakable with less than 2^{80} operations.

1. for any system, key size > 80 bits. (Try all keys)
2. for a system based on Dlog in a group, key size > 160 bits. (Pollard's Rho)
3. for a system based on factorization, key size > 1024 bits. (NFS)

Elliptic curves. "optimal" for Dlog based cryptosystems:

- Group order can be computed quickly (Schoof–Elkies–Atkin, Satoh–ECPC).
- In general, the best known attack is Pollard's Rho.

Higher genus curves. Same kind of behaviour:

+ Operations are in a smaller base field (no multiprecision?)

− Group law is more complicated (more base field operations)

− There exists sub-exponential attacks when genus is “high”

Question: What is “high” genus ? In practice ?

We will concentrate on (imaginary) hyperelliptic curves

- Easiest group law

- First high genus attack was designed for these curves

- Extends to the case of curves with a smooth affine model with a unique place at infinity ($\mathcal{C}_{ab}$, superelliptic curves)

Hyperelliptic curves

Def: Curves with equation $y^2 + h(x)y = f(x)$ over $\mathbb{F}_q$, where $\deg f(x) = 2g + 1$ and $\deg h(x) \leq g$. The integer g is called the genus.

Rem: Case $g = 1$ is elliptic curve in Weierstraß form.

If $g > 1$ the curve is not a group $\Longrightarrow$ go to its Jacobian.

Def: The Jacobian of a curve is Jac$(\mathcal{C}) = \mathrm{Div}_0(\mathcal{C})/\mathrm{Pr}(\mathcal{C})$.

Thm (Mumford, Cantor): In each divisor class, there is a unique representative $D = \sum_i P_i - r\infty$, such that $r \leq g$ and no two P_i's are opposites. This representative is called a reduced divisor.

Opposite of $P_i = (x_i, y_i)$ is $P_i' = (x_i, -y_i - h(x_i))$.

Def: D is defined over $\mathbb{F}_q$ iff it is Galois invariant.

Rem: D defined over $\mathbb{F}_q$ does not imply P_i defined over $\mathbb{F}_q$.

Mumford representation, Cantor's algorithm

Prop: A reduced divisor $D = \sum_i P_i - r\infty$ can be represented by a pair of polynomials $\langle u(x), v(x)\rangle$, defined by $u(x) = \prod(x - x_i)$, $v(x_i) = y_i$, $\deg v < \deg u \leq g$ and the condition $u(x) \mid v(x)^2 - h(x)v(x) - f(x)$.

Rem: If D is defined over $\mathbb{F}_q$ then $u(x)$ and $v(x)$ are also defined over $\mathbb{F}_q$.

The group law can be computed by geometry $\Longrightarrow$ slow.

Cantor's algorithm: operates on reduced divisors in Mumford representation.

2 phases: composition, reduction. Consist of multplications, divisions, GCD's of polynomials.

$\Longrightarrow$ Complexity = polynomial in $g \log q$.

Improvements: Spallek, Harley, Nagao.

Ex: in genus 2, one operation costs 30 mult + 2 inv in base field.

Cardinality

Thm (Hasse–Weil bound):

$$(\sqrt{q}-1)^{2g} \leq \#\text{Jac}(\mathcal{C}) \leq (\sqrt{q}+1)^{2g}.$$

We have $\#\text{Jac} \approx q^g$. To obtain $\#\text{Jac} \approx 2^{160}$, we need $q \approx 2^{\frac{160}{g}}$.

Rem: In general, it is difficult to compute the exact value of $\#\text{Jac}(C) \Longrightarrow$ use curves with special properties.

- Koblitz curves
- Curves with complex multiplication
- Curves with real multiplication (modular curves)

See G.–Harley (ANTS-IV) for computations with random curves in genus 2.

The discrete log problem

Input: A hyperelliptic curve $\mathcal{C}$, a divisor $D_1 \in \mathrm{Jac}(\mathcal{C})$, its order n, and another divisor $D_2 \in \langle D_1 \rangle$.

Output: An integer λ modulo n such that $D_2 = \lambda.D_1$

Special cases:

- Supersingular curves : Frey-Rück. Cost: dlog computation in $\mathbb{F}_{q^k}$.
- Group of order $n = p^k$: Rück. Cost: polynomial time.
- Large genus: Adleman–DeMarrais–Huang, Flassenberg–Paulus, Enge, Müller–Stein–Thiel. Cost: $L_{q^g}(1/2, c)$.

General case: Pollard's Rho algorithm, Cost $O(q^{g/2})$. Our algorithm: à la Haffner–McCurley. Cost: $O(q^2 + qg!)$.

The Adleman–DeMarrais–Huang algorithm (1994)

Presented for odd characteristic $\Longrightarrow$ assume $h(x) = 0$.

Index-calculus in the function field of the curve, using the following **smoothness notion**:

Def: A divisor over $\mathbb{F}_q$ is said to be S-smooth if all its points are defined over an extension $\mathbb{F}_{q^k}$, wih $k \leq S$.

Prop: If $D = \langle u(x), v(x) \rangle$ is in Mumford representation, then D is S-smooth iff the irreducible factors of $u(x)$ have degree $\leq S$.

Def: A divisor is said to be prime if $u(x)$ is irreducible.

Prop: A divisor $D = \langle u(x), v(x) \rangle$ is equal to the sum of the prime divisors $\langle u_i(x), v_i(x) \rangle$, where the $u_i(x)$ are the irreducible factors of $u(x)$.

$\Longrightarrow$ $\mathrm{Div}(\mathcal{C})$ inherits the "unique factorization theorem" for polynomials.

Algorithm

1. Choose a smoothness bound S, and build the factor basis = set of prime divisors of degree at most S.

2. Find relations between those primes.

3. Solve for the group structure (SNF).

4. Express D_1 and D_2 on the basis.

5. Compute the discrete log.

Step 2: choose random polynomial $A(x)$ and $B(x)$, construct the divisor of the function $A(x)y + B(x)$, test its smoothness ($\Leftrightarrow$ polynomial $A(x)^2 f(x) - B(x)^2$ is S-smooth).

Complexity:

Choose $S = \log_q L_{q^g}(\frac{1}{2}, a)$. Under certain heuristics, gives a total runtime

$$L_{q^g}\left(\frac{1}{2}, c\right).$$

Heuristics: Factor basis contains a generating set, the norm of a random function is a random polyomial.

Rem: S has to be an integer. If genus is low, the rounding for S will change the complexity.

For example, if $g = 1$ is fixed and q tends to infinity, then $S = 1$, and the factor basis contains all the $O(q)$ points on the elliptic curve $\Rightarrow$ exponential complexity.

Flassenberg and Paulus variant (1999)

Idea: Instead of chosing random $A(x)$ and $B(x)$, select the good ones by a sieve.

Based on

$$p(x)|A(x)^2 f(x) - B(x)^2 \iff p(x)|A'(x)^2 f(x) - B'(x)^2,$$

where $A'(x) = A(x) + r(x)p(x)$ and $B'(x) = B(x) + s(x)p(x)$.

Experiments: Curve of genus 12 over $\mathbb{F}_{11}$ and curve of genus 6 over $\mathbb{F}_{101}$, in a few hours.

Crossover?: It “seems” that it becomes faster than BSGS... but what consequence for cryptography?

Müller–Stein–Thiel, Enge ($\approx$ 1999-2000)

Index-calculus in the Jacobian instead of in the function field.

They get proofs of subexponential complexity.

Extensions to real hyperelliptic curves (MST), and even characteristic (Enge).

Complexity:

$$L_{q^g}\left(\frac{1}{2}, \frac{5}{2\sqrt{3}}\right).$$

Tools: Proven complexity of SNF, bound for a generating set.

Interesting point: Enge gives an expression of the complexity depending on the ratio $\frac{g}{\log q}$.

Theoretical work: still does not tell the consequences in cryptography.

Variant that we will describe (Eurocrypt 2000)

We restrict to cryptographical context:

- We assume that $\#$Jac is known and almost prime.

In theory, we can find $\#$Jac in polynomial time, and algorithm applies if we know (or can guess) the group structure.

$\Longrightarrow$ do not need a HNF computation

$\Longrightarrow$ sparse methods can be used (Lanczos).

- consider g fixed and q tending to infinity.

We want to answer the question "Are the genus g curves secure?".

$\Longrightarrow$ exponential complexity but easy to analyze and to compare to Rho method

- Always take $S = 1$.

Algorithm

- Build the factor basis $\mathcal{F} = \{P_j\}$ (points of the curve).
- Initiate a pseudo-random walk similar to Pollard's Rho method. Start with a divisor R_0, compute R_{i+1} from R_i by adding one among 20 precomputed divisors: $R_{i+1} = R_i + T^{(\mathcal{H}(R_i))}$, where $\mathcal{H}$ is a hash function sending Jac$(\mathcal{C})$ to $\{1, \ldots, 20\}$. (see Teske, Sattler–Schnorr)
  Maintain a representation $R_i = \alpha_i D_1 + \beta_i D_2$.
- Retain only the smooth R_i's, and store them in a matrix $M = (m_{ij})$ of size $(\#\mathcal{F} + 1) \times \#\mathcal{F}$, each row defined by
$$R_i = \sum m_{ij} P_j.$$
- Compute a non zero vector $(\gamma_i) \in \mathrm{Ker}({}^t M)$.
- Return $\lambda = -\left(\sum \alpha_i \gamma_i\right) / \left(\sum \beta_i \gamma_i\right) \bmod n$.

Numerical example

Curve of genus 3 over $\mathbb{F}_{19}$ defined by

$$y^2 = x^7 + 18x^5 + 16x^4 + 18x^3 + 6x^2 + 2x + 11.$$

One has $\#\mathrm{Jac}(\mathcal{C}) = n = 6073$.

Let

$$\begin{aligned} D_1 &= \langle x^3 + 3x^2 + 5x + 18,\ \ 11x^2 + 18x + 2\rangle \\ D_2 &= \langle x^3 + 14x + 17,\ \ 17x^2 + 16x + 5\rangle. \end{aligned}$$

Factor basis:

$$\mathcal{F} = \Big\{P_1 = \langle x+2, 3\rangle,\ P_2 = \langle x+5, 3\rangle,\ P_3 = \langle x+6, 15\rangle,\ P_4 = \langle x+7, 16\rangle,$$

$$P_5 = \langle x+8, 11\rangle,\ P_6 = \langle x+9, 9\rangle,\ P_7 = \langle x+16, 4\rangle,\ P_8 = \langle x, 12\rangle\Big\}$$

Relations:

$$\begin{array}{rcrcl}
4290D_1 & + & 266D_2 & = & P_8 - P_1 - P_7 \\
2179D_1 & + & 1076D_2 & = & P_2 - P_3 + P_4 \\
3260D_1 & + & 4395D_2 & = & P_7 - P_6 + P_5 \\
4345D_1 & + & 1253D_2 & = & P_1 - P_2 - P_3 \\
4132D_1 & + & 5748D_2 & = & P_3 - P_5 - P_6 \\
296D_1 & + & 1779D_2 & = & -P_1 + P_3 + P_5 \\
5750D_1 & + & 1098D_2 & = & P_2 - P_5 + P_6 \\
5390D_1 & + & 3436D_2 & = & P_8 + P_1 - P_5 \\
5459D_1 & + & 5868D_2 & = & P_8 + P_1 + P_7
\end{array}$$

We get the following matrix:

$$M = \begin{pmatrix}
-1 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\
0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 \\
1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & -1 & -1 & 0 & 0 \\
-1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & -1 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{pmatrix}$$

and we must compute its kernel modulo $n = 6073$.

One gets for example the vector

$$\gamma = [0, 0, 1, 1, 0, 1, 1, 1, -1].$$

And then

$$\begin{aligned} \lambda &= -\tfrac{3260+4345+296+5750+5390-5459}{4395+1253+1779+1098+3436-5868} \\ &= 3572 \mod 6073. \end{aligned}$$

Runtime of the algorithm

We count the number of elementary operations. Four kinds (polynomial runtime):

1) Additions in the Jacobian. Cost: $O(g^3 \log^2 q)$.

2) Operations in $\mathbb{F}_q$. Cost: $O(\log^3 q)$.

3) Operations in $\mathbb{Z}/n\mathbb{Z}$. Cost: $O(g^2 \log^2 q)$.

4) Operations on polynomials of degree $O(g)$ over $\mathbb{F}_q$. Cost: $O(g^2 \log^3 q)$.

- Building the factor basis: $O(q)$ equations of degree 2 in $\mathbb{F}_q$ to solve.
- Probability for a random divisor to be smooth: The Hasse-Weil bound gives $1/(g!)$.
- One step in the random walk + smoothness test: polynomial time $O(1)$.
- Linear algebra: Lanczos algorithm $O(q^2)$.

Overall complexity: $O(qg! + q^2)$.

Rem: for fixed g, one gets $O(q^2)$.

This approximation is reasonnable if $g! = O(q)$. For $g \log q \approx 160$, it is valid for $g \leq 9$.

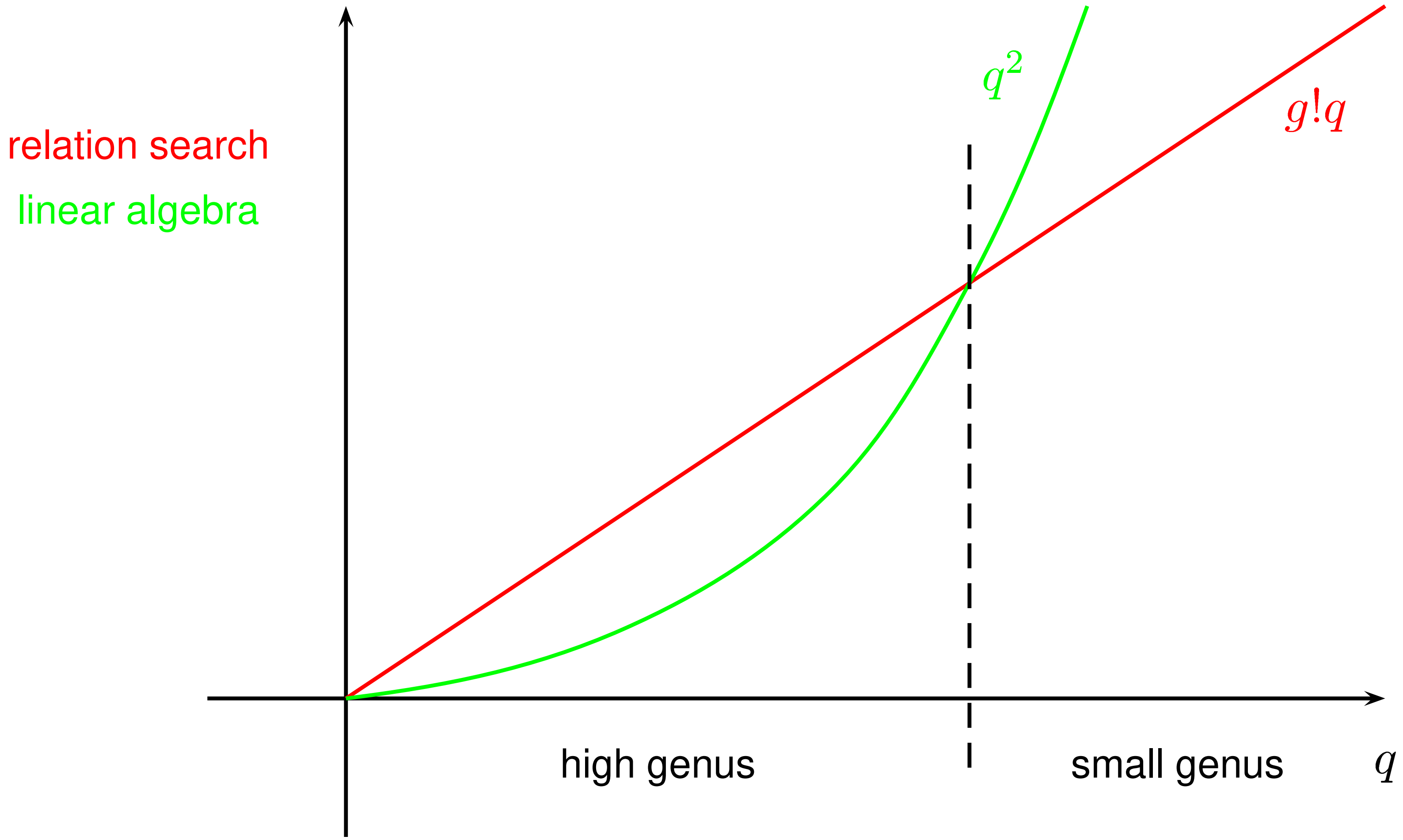
Comparison with Pollard's Rho method

Group size: $n \approx q^g$.

Complexity of Rho: $O(\sqrt{n}) = O(q^{g/2})$.

Complexity of index calculus: $O(q^2)$ (for fixed g).

g	1	2	3	4	5	6	7
Rho	$q^{1/2}$	q	$q^{3/2}$	q^2	$q^{5/2}$	q^3	$q^{7/2}$
Index	q^2	q^2	q^2	q^2	q^2	q^2	q^2


relation search
linear algebra
q^2
$g!q$
high genus
small genus
q

Using automorphisms of the curve

Prop: If σ is an automorphism of $\mathcal{C}$ of order m, then it can be extended to Jac$(\mathcal{C})$ by linearity.

Prop: σ is compatible with the smoothness notion:

$$D \text{ smooth} \Longleftrightarrow D^{\sigma} \text{ smooth}$$

$\Longrightarrow$ keep only one representative for each orbit in the factor basis.

Advantage: reduce factor basis by a factor m. Complexity:

$$O\left(\frac{g!q}{m} + \frac{q^2}{m^2}\right)$$

Drawback: The entries in the matrix are roots of unity instead of 1's.

Duursma–G.–Morain (Asiacrypt ’99)

In the context of Rho method, speed-up by a factor $\sqrt{m}$.

Rem: Not only Frobenius automorphism (Koblitz curves) but every “easy computable” automorphism.

$\Longrightarrow$ new examples coming from Complex Multiplication.

Rem: Almost every hyperelliptic curves proposed for cryptography have automorphisms.

Due to the difficulty to compute group order for random curves.

Exceptions:

- Curves of Spallek: CM-curves, but the unit group is torsion free.
- Modular curves (Frey–Müller): cryptographical size?
- Curves coming from Weil-Descent

Author	Curve equation	Field	Automorphisms	m
Koblitz	$Y^2 + Y = X^5 + X^3$	$\mathbb{F}_{2^n}$	Frob + $\begin{cases} X \mapsto X+1 \\ Y \mapsto Y + X^2 \end{cases}$	$4n$
	$Y^2 + Y = X^5 + X^3 + X$	$\mathbb{F}_{2^n}$	Frobenius	$2n$
	$Y^2 + Y = X^{2g+1} + X$	$\mathbb{F}_{2^n}$	Frobenius	$2n$
	$Y^2 + Y = X^{2g+1}$	$\mathbb{F}_{2^n}$	Frobenius	$2n$
Buhler–Koblitz Chao et al.	$Y^2 + Y = X^{2g+1}$ (and twists)	$\mathbb{F}_p$ with $p \equiv 1\ (2g+1)$	multiplication by ζ_{2g+1}	$2(2g+1)$
Sakai–Sakurai	$Y^2 + Y = X^{13} + X^{11} + X^9 + X^5 + 1$	$\mathbb{F}_{2^{29}}$	Frob + $\begin{cases} X \mapsto X+1 \\ Y \mapsto Y + X^6 + X^5 + X^4 + X^3 + X^2 \end{cases}$	4×29
Duursma–Sakurai	$Y^2 = X^p - X + 1$	$\mathbb{F}_{p^n}$	Frob + $\begin{cases} X \mapsto X+1 \\ Y \mapsto Y \end{cases}$	$2np$

Example: curve of genus 6 proposed by Koblitz

field	$\mathbb{F}_{5026243}$
equation	$y^2 + y = x^{13}$
genus	6
$\#J$	$13^3 \times 7345240503856807663632202049344834001 \approx 10^{40}$

Rem: $\zeta_{13} \ : \ (x, y) \mapsto (\zeta_{13} x, y)$ is an automorphism on the curve.

We keep in the factor basis only one representative under the action of $\zeta_{13} \Longrightarrow$ reduce the size of G.

Size of factor basis: $2, 511, 392$ vs $193, 185$.

Runtime on a Pentium II 450 MHz, 128 Mb (used 60Mb).

size of the factor basis	$193,485$
time for building the basis	1638 sec
number of random steps	$201,426,284$
number of early abort by Swan	$100,721,873$
number of relations collected	$281,200$
proportion of smooths $(g!)$	$716.3\ (720)$
total time for collecting the relations	$513,870$ sec $=$ **6 days**
time for building the matrix	$8,822$ sec
preprocessing the matrix	1218 sec
size of the reduced matrix	$165,778 \times 165,779$
total time for Lanczos	$780,268$ sec $=$ **9 days**

Solving the linear algebra in practice

Pretraitement of the matrix

- Eliminate empty columns
- Eliminate columns with only one entry and the corresponding rows
- If there is more rows than columns, eliminate one arbitrary row (use heuristics!)
- Begin a Gaussian elimination where the pivot is chosen in order to minimize the increase of the weight.

Clique algorithm by Cavalar may be useful (but here the matrix is not a binary matrix).

Rem: Proportion of empty columns: e^{-g}, with one entry: ge^{-g}.

Lanczos algorithm

Iterative method: main operation is (matrix $\times$ vector).

$\Longrightarrow$ during the process, the matrix is not modified. Remains sparse.

Implementation: (see Denny–Weber, Lercier–Joux)

- use compact representation
- lazy reduction technique
- Montgomery representation

Parallelization: By Coppersmith's Block Wiedemann algorithm. Need some experiments. Fast sequential stage by Thomé. Bottleneck: each processor must have the whole matrix in central memory.

Reducing the factor basis (Harley, Smart)

Problem: How to balance both phases when genus is very small?

Sol: keep in the factor basis a proportion $Q = \alpha q$ of the primes divisors ($\alpha < 1$).

- Probablity of finding a smooth divisor is reduced by a factor $\frac{1}{\alpha^g}$.
- Number of relations needed is reduced by a factor $\frac{1}{\alpha}$.
- Linear algebra is speeded-up by a factor α^2.

Cost of building the matrix: $g!\frac{q}{\alpha^{g-1}}$.

Cost of linear algebra: $\alpha^2 q^2$.

Choose $\alpha = \left(\frac{q}{g!}\right)^{-\frac{1}{g+1}}$.

Complexity: $O(q^{\frac{2g}{g+1}})$

Rem: For $g = 4$, the complexity is $O(q^{\frac{8}{5}})$, which is better than $O(q^2)$ of Rho method.

$\Longrightarrow$ genus 4 curves may be weaker than genus 1,2,3 curves for a given key size.

Choice of the primes to keep: (example)

Over $\mathbb{F}_{2^d}$, polynomial base representation. Fix $i_0 < d$.

A base field element $x = \sum_{0 \leq i \leq d-1} x_i t^i$ is distinguished if $x_i = 0$ for $i > i_0$.

Keep in the basis those primes (x, y) for which x is distinguished.

Prop: If a divisor is smooth on this reduced basis, then the sub-leading coefficient of $u(x)$ in the Mumford representation is distinguished.

(sum of distinguished elements)

$\Longrightarrow$ Very fast smoothness test.

Construction of the matrix

Rem. This phase is very easy to parallelize.

Modifying the random walk:

With the classical walk, one step = one “generic” addition in Jac.

Rem: Adding a point is faster than adding a weight g divisor.

1. Start with R_i in the classical random walk.
2. Choose a random element P in the basis.
3. While R_i is not smooth, add P to R_i.
4. Put a row of the form $R_i = -kP + \sum a_j P_j$ in the matrix.

$\Longrightarrow$ Gain: add an element of weight 1 instead of g.

Smoothness test:

- Usual Smoothness test for $D = \langle u(x), v(x) \rangle$:

  Compute $\mathrm{GCD}(x^q - x, u(x))$. Return smooth if the degree = $\deg u(x)$.

- Swan's theorem:

  Let r be the number of irred. factors of $u(x)$. Then

$$r \equiv \deg u(x) \mod 2 \text{ iff } \mathrm{disc}(u(x)) \text{ is a square}$$

$\Longrightarrow$ obtain a speed-up in char $\neq 2$ (early abort).

Index calculus in genus 2

Claim: the linear algebra phase can be done in linear time.

Building a graph:

- vertices are the $O(q)$ elements of the factor basis = points on the curve.
- Each time one finds a relation $\alpha D_1 + \beta D_2 = P_i + P_j$, put an edge between P_i and P_j, labelled by α and β.

If one has a cycle, the discrete log follows (with proba $\frac{1}{2}$).

With Union–Find algorithm one detects cycles in (almost) linear time in the number of edges.

Thm (Flajolet–Knuth–Pittel): first cycle in a random graph appears after κq edges.

Experimentally, $\kappa \approx 0.23$ (F-K-P give constants but not for our model).

$\Longrightarrow$ complexity in $\approx 0.46q$ group operations and central memory. (Rho $\approx 0.88q$).

Conclusion, perspectives

- very simple algorithm, simple to analyze, faster than Rho for $g \geq 4$.
- easy to implement, but ∞^{ty} of tricks can (have to) be played to speed up, depending on the given problem
- application to Weil Descent: new parameters

Future work:

- study the case of genus 3.
- study for more general curves (see Arita).
- is the $g!$ factor inevitable? can we fill in the matrix with a new method?