Simplified Settings for Discrete Logarithms in Small Characteristic Finite Fields

Antoine Joux

CryptoExperts, Fondation UPMC, LIP6, INRIA/Ouragan

Joint work with Cécile Pierrot

PKC 2015
March 30th, Gaithersburg

The Discrete Logarithm Problem (DLP)

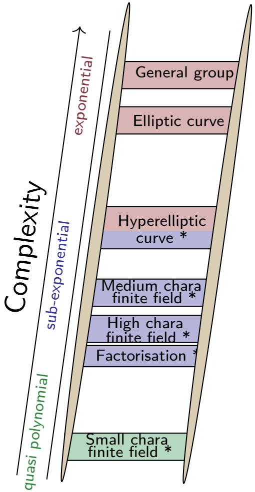
- Multiplicative group G generated by g: solving the discrete logarithm problem in G, is inverting the map $x \mapsto g^x$

The Discrete Logarithm Problem (DLP)

- Multiplicative group G generated by g: solving the discrete logarithm problem in G, is inverting the map $x \mapsto g^x$
- A hard problem in general, and used as such in cryptography.

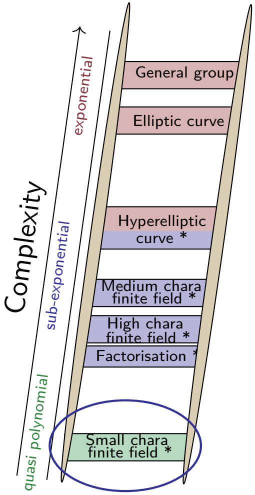
The Discrete Logarithm Problem (DLP)

- Multiplicative group G generated by g: solving the discrete logarithm problem in G, is inverting the map $x \mapsto g^x$
- A hard problem in general,
  and used as such in cryptography.
- Several groups in practice:



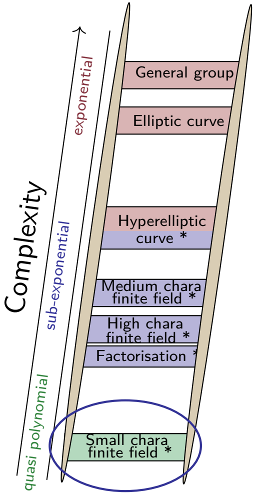
The Discrete Logarithm Problem (DLP)

- Multiplicative group G generated by g: solving the discrete logarithm problem in G, is inverting the map $x \mapsto g^x$
- A hard problem in general, and used as such in cryptography.
- Several groups in practice:



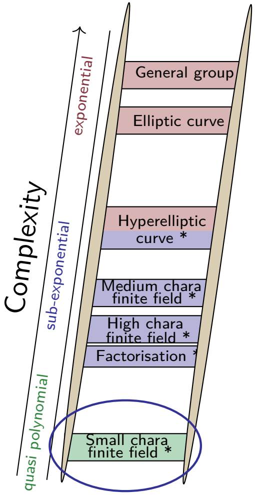
The Discrete Logarithm Problem (DLP)

- Multiplicative group G generated by g: solving the discrete logarithm problem in G, is inverting the map $x \mapsto g^x$
- A hard problem in general, and used as such in cryptography.
- Several groups in practice:
- Two algorithmic approaches:
    - Generic algorithms (Pollard's Rho, Pohlig-Hellman...)
    - Specific algorithms (Index Calculus)



The Discrete Logarithm Problem (DLP)

- Multiplicative group G generated by g: solving the discrete logarithm problem in G, is inverting the map $x \mapsto g^x$
- A hard problem in general, and used as such in cryptography.
- Several groups in practice:
- Two algorithmic approaches:
  - Generic algorithms (Pollard's Rho, Pohlig-Hellman...)
  - Specific algorithms (Index Calculus *)



Generic algorithms: Pohlig-Hellman

Generic algorithms: Pohlig-Hellman

- Given a multiplicative group G with generator g

Generic algorithms: Pohlig-Hellman

- Given a multiplicative group G with generator g
- Given $|G| = \prod_{i=1}^{k} p_i^{e_i}$

Generic algorithms: Pohlig-Hellman

- Given a multiplicative group G with generator g
- Given $|G| = \prod_{i=1}^{k} p_i^{e_i}$
- To compute dlogs in G, it suffices to compute dlogs in:

$$G_i = \langle g^{|G|/p_i} \rangle \quad \text{(Group of order } p_i\text{)}$$

Generic algorithms: $|G| = p$

Generic algorithms: $|G| = p$

- There exist algorithms with complexity $O(\sqrt{p})$ to solve:

$$y = g^n$$

Generic algorithms: $|G| = p$

- There exist algorithms with complexity $O(\sqrt{p})$ to solve:

$$y = g^n$$

- Baby-step giant-step (let $R = \lceil \sqrt{p} \rceil$):
  - Create list $y, y/g, \cdots, y/g^{R-1}$
  - Create list $1, h, h^2, \cdots, h^{R-1}$, where $h = g^R$
  - Find collision

Generic algorithms: $|G| = p$

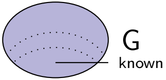
- There exist algorithms with complexity $O(\sqrt{p})$ to solve:

$$y = g^n$$

- Baby-step giant-step (let $R = \lceil \sqrt{p} \rceil$):
  - Create list $y, y/g, \cdots, y/g^{R-1}$
  - Create list $1, h, h^2, \cdots, h^{R-1}$, where $h = g^R$
  - Find collision
- Can be improved to memoryless algorithms using cycle finding techniques

Index Calculus Algorithms

To compute Discrete Logs in G:



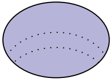
1. Collection of Relations
   → Create a lot of sparse multiplicative relations between some (small) specific elements = the factor base

$$\prod g_i^{e_i} = \prod g_i^{e'_i} \quad \Rightarrow \quad \sum (e_i - e'_i) \log(g_i) = 0$$

   → So a lot of sparse linear equations

Index Calculus Algorithms

To compute Discrete Logs in G:


G

1. Collection of Relations
   $\rightarrow$ Create a lot of sparse multiplicative relations between some (small) specific elements = the factor base

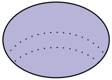
$$\prod g_i^{e_i} = \prod g_i^{e'_i} \quad \Rightarrow \quad \sum (e_i - e'_i) \log(g_i) = 0$$

   $\rightarrow$ So a lot of sparse linear equations

2. Linear Algebra
   $\rightarrow$ Recover the Discrete Logs of the factor base

Index Calculus Algorithms

To compute Discrete Logs in G:



G

1. Collection of Relations
   $\rightarrow$ Create a lot of sparse multiplicative relations between some (small) specific elements = the factor base

$$\prod g_i^{e_i} = \prod g_i^{e'_i} \quad \Rightarrow \quad \sum (e_i - e'_i) \log(g_i) = 0$$

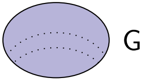
   $\rightarrow$ So a lot of sparse linear equations

2. Linear Algebra
   $\rightarrow$ Recover the Discrete Logs of the factor base

3. Extension Phase (for small characteristic finite fields)
   $\rightarrow$ Recover the Discrete Logs of the extended factor base

Index Calculus Algorithms

To compute Discrete Logs in G:



1. Collection of Relations

   $\rightarrow$ Create a lot of sparse multiplicative relations between some (small) specific elements = the factor base

$$\prod g_i^{e_i} = \prod g_i^{e'_i} \quad \Rightarrow \quad \sum (e_i - e'_i) \log(g_i) = 0$$

   $\rightarrow$ So a lot of sparse linear equations

2. Linear Algebra

   $\rightarrow$ Recover the Discrete Logs of the factor base

3. Extension Phase (for small characteristic finite fields)

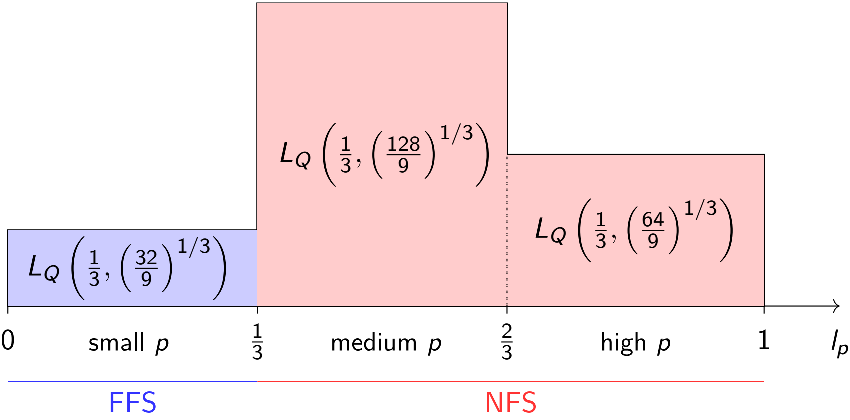
   $\rightarrow$ Recover the Discrete Logs of the extended factor base

4. Individual Logarithm Phase

   $\rightarrow$ Recover the Discrete Log of an arbitrary element

Complexity of Index calculus algorithms (before 2013)

$$L_Q(\beta, c) = \exp((c + o(1))(\log Q)^{\beta}(\log \log Q)^{1-\beta}).$$



Index Calculus for Small Characteristic Finite Fields

- $G = \mathbb{F}_{p^n}$ where p is small compared to n.

Index Calculus for Small Characteristic Finite Fields

- $G = \mathbb{F}_{p^n}$ where p is small compared to n.
- Asymptotic Complexities:

$$\left.\begin{array}{c}\text{Collection of Relations}\\ \text{Linear Algebra}\\ \text{Extension Phase}\end{array}\right\}\ \text{Polynomial time}$$

Index Calculus for Small Characteristic Finite Fields

- $G = \mathbb{F}_{p^n}$ where p is small compared to n.
- Asymptotic Complexities:

$$\left.\begin{array}{c}\text{Collection of Relations}\\ \text{Linear Algebra}\\ \text{Extension Phase}\end{array}\right\}\text{Polynomial time}$$

$$\text{Individual Logarithm Phase}\ \Big\}\ \text{Quasipolynomial time}$$

Index Calculus for Small Characteristic Finite Fields

- $G = \mathbb{F}_{p^n}$ where p is small compared to n.
- Asymptotic Complexities:

$$\left.\begin{array}{c}\text{Collection of Relations}\\ \text{Linear Algebra}\\ \text{Extension Phase}\end{array}\right\}\ \text{Polynomial time}$$

$$\text{Individual Logarithm Phase}\ \Big\}\ \text{Quasipolynomial time}$$

- In practice:
  Linear algebra and extension phase dominate.

Index Calculus for Small Characteristic Finite Fields

- $G = \mathbb{F}_{p^n}$ where p is small compared to n.
- Asymptotic Complexities:

$$\left.\begin{array}{c}\text{Collection of Relations}\\ \text{Linear Algebra}\\ \text{Extension Phase}\end{array}\right\}\text{Polynomial time}$$

$$\text{Individual Logarithm Phase}\ \}\ \text{Quasipolynomial time}$$

- In practice:
  Linear algebra and extension phase dominate.
- In this talk:
  Simplified description of algorithms + additional ideas
  $\Rightarrow$ Improve the complexity of the polynomial phases.

Frobenius Representation Algorithms

- Our goal: solve the DLP in $\mathbb{F}_{p^n}$ with small characteristic.

- Our goal: solve the DLP in $\mathbb{F}_{p^n}$ with small characteristic.
- How ? Represent $\mathbb{F}_{p^n}$ as

  $\mathbb{F}_{q^k}$

  $\mathbb{F}_{p^{m.k}} = \mathbb{F}_{(p^m)^k} = \mathbb{F}_{q^k} \simeq \mathbb{F}_q[X]/(I(X))$ where
  $I(X)$ is an irreducible polynomial of degree k such that:

  $$I(X) | h_1(X)X^q - h_0(X)$$

  where h_0 and h_1 are polynomials of low degrees.

Frobenius Representation Algorithms

- Our goal: solve the DLP in $\mathbb{F}_{p^n}$ with small characteristic.
- How ? Represent $\mathbb{F}_{p^n}$ as

  $\mathbb{F}_{p^{m.k}} = \mathbb{F}_{(p^m)^k} = \mathbb{F}_{q^k} \simeq \mathbb{F}_q[X]/(I(X))$ where $I(X)$ is an irreducible polynomial of degree k such that:

  $$I(X) | h_1(X)X^q - h_0(X)$$

  where h_0 and h_1 are polynomials of low degrees.
- Why ? To have two equations in the finite field:

$$\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \quad \text{and} \quad \underbrace{X^q = \frac{h_0(X)}{h_1(X)}}_{\text{Frobenius Representation}}$$

$\mathbb{F}_{q^k}$

Frobenius Representation Algorithms

- Our goal: solve the DLP in $\mathbb{F}_{p^n}$ with small characteristic.
- How ? Represent $\mathbb{F}_{p^n}$ as

  $\mathbb{F}_{q^k}$

  $\mathbb{F}_{p^{m.k}} = \mathbb{F}_{(p^m)^k} = \mathbb{F}_{q^k} \simeq \mathbb{F}_q[X]/(I(X))$ where
  $I(X)$ is an irreducible polynomial of degree k such that:

  $$I(X) | h_1(X)X^q - h_0(X)$$

  where h_0 and h_1 are polynomials of low degrees.
- Why ? To have two equations in the finite field:

  $$\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \quad \text{and} \quad \underbrace{X^q = \frac{h_0(X)}{h_1(X)}}_{\text{Frobenius Representation}}$$

- What choice do we have ? Degree of h_0 and h_1.

Frobenius Representation Algorithms

- Our goal: solve the DLP in $\mathbb{F}_{p^n}$ with small characteristic.
- How ? Represent $\mathbb{F}_{p^n}$ as

  $\mathbb{F}_{q^k}$

  $\mathbb{F}_{p^{m.k}} = \mathbb{F}_{(p^m)^k} = \mathbb{F}_{q^k} \simeq \mathbb{F}_q[X]/(I(X))$ where
  $I(X)$ is an irreducible polynomial of degree k such that:

  $$I(X) | h_1(X) X^q - h_0(X)$$

  where h_0 and h_1 are polynomials of low degrees.
- Why ? To have two equations in the finite field:

$$\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \qquad \text{and} \qquad \underbrace{X^q = \frac{h_0(X)}{h_1(X)}}_{\text{Frobenius Representation}}$$

- What choice do we have ? Degree of h_0 and h_1.
- Simplest choice: To take

  h_0 : deg 1 polynomial
  h_1 : deg 2 polynomial

Frobenius Representation Algorithms

- Our goal: solve the DLP in $\mathbb{F}_{p^n}$ with small characteristic.
- How ? Represent $\mathbb{F}_{p^n}$ as

  $\mathbb{F}_{q^k}$

  $\mathbb{F}_{p^{m.k}} = \mathbb{F}_{(p^m)^k} = \mathbb{F}_{q^k} \simeq \mathbb{F}_q[X]/(I(X))$ where $I(X)$ is an irreducible polynomial of degree k such that:

$$I(X) | h_1(X)X^q - h_0(X) \quad \text{or} \quad I(X) | h_1(X^q)X - h_0(X^q)$$

  where h_0 and h_1 are polynomials of low degrees.
- Why ? To have two equations in the finite field:

$$\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \quad \text{and} \quad \underbrace{X^q = \frac{h_0(X)}{h_1(X)}}_{\text{Frobenius Representation}} \quad \text{or} \quad \underbrace{X = \frac{h_0(X^q)}{h_1(X^q)}}_{\text{Dual Frob. Rep.}}$$

- What choice do we have ? Degree of h_0 and h_1.
- Simplest choice: To take

h_0 :	deg 1 polynomial	or	h_0 :	deg 2 polynomial	useful variant
h_1 :	deg 2 polynomial		h_1 :	deg 1 polynomial	

Creation of Relations

Our goal: multiplicative relation between small degree polynomials.

Creation of Relations

Our goal: multiplicative relation between small degree polynomials.

Main idea : start from $\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \quad (**)$.

Creation of Relations

Our goal: multiplicative relation between small degree polynomials.

Main idea : start from $\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \quad (**)$.

Let A and B be 2 small polynomials in $\mathbb{F}_q[X]$ (i.e. of degree $\leqslant D$).

$$B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X)) \underset{\text{thanks to } (**)}{=} A(X)^q B(X) - A(X) B(X)^q$$

$$\underset{\text{Frob. linearity}}{=} A(X^q) B(X) - A(X) B(X^q)$$

$$\underset{\text{Frob. Rep.}}{=} \underbrace{A\left(\frac{h_0(X)}{h_1(X)}\right) B(X) - A(X) B\left(\frac{h_0(X)}{h_1(X)}\right)}_{\frac{[A,B]_D}{h_1(X)^D}}$$

Creation of Relations

Our goal: multiplicative relation between small degree polynomials.

Main idea : start from $\prod_{\alpha \in \mathbb{F}_q} (X - \alpha) = X^q - X \quad (**)$.

Let A and B be 2 small polynomials in $\mathbb{F}_q[X]$ (i.e. of degree $\leqslant D$).

$$B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X)) \underset{\text{thanks to } (**)}{=} A(X)^q B(X) - A(X) B(X)^q \underset{\text{Frob. linearity}}{=} A(X^q) B(X) - A(X) B(X^q) \underset{\text{Frob. Rep.}}{=} \underbrace{A\left(\frac{h_0(X)}{h_1(X)}\right) B(X) - A(X) B\left(\frac{h_0(X)}{h_1(X)}\right)}_{\frac{[A,B]_D}{h_1(X)^D}}$$

We finally get:

$$\underbrace{h_1(X)^D B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{Product of small polynomials !!}} = [A, B]_D(X)$$

Properties and simplification of $[A, B]_D(X)$

- $[A, B]_D$ is bilinear

Properties and simplification of $[A, B]_D(X)$

- $[A, B]_D$ is bilinear
- $[A, A]_D = 0$.

Properties and simplification of $[A, B]_D(X)$

- $[A, B]_D$ is bilinear
- $[A, A]_D = 0$.
- Thus, A and B can be assumed monic.

Properties and simplification of $[A, B]_D(X)$

- $[A, B]_D$ is bilinear
- $[A, A]_D = 0$.
- Thus, A and B can be assumed monic.
- Since $[A, B]_D = [A, B - A]_D$, we may also assume $\deg B < \deg A$.

Properties and simplification of $[A, B]_D(X)$

- $[A, B]_D$ is bilinear
- $[A, A]_D = 0$.
- Thus, A and B can be assumed monic.
- Since $[A, B]_D = [A, B - A]_D$, we may also assume $\deg B < \deg A$.
- Assume $\deg A = D$ and $\deg B = D - 1$. Then, using bilinearity, one may reduce the coefficient of X^{D-1} in A to 0.

Properties and simplification of $[A, B]_D(X)$

- $[A, B]_D$ is bilinear
- $[A, A]_D = 0$.
- Thus, A and B can be assumed monic.
- Since $[A, B]_D = [A, B - A]_D$, we may also assume $\deg B < \deg A$.
- Assume $\deg A = D$ and $\deg B = D - 1$. Then, using bilinearity, one may reduce the coefficient of X^{D-1} in A to 0.
- In the sequel, we assume:

$$\begin{aligned} A(X) &= X^D + A_{D-2}(X) \text{ and} \\ B(X) &= X^{D-1} + B_{D-2}(X). \end{aligned}$$

A Small Factor Base

We have: $$\underbrace{h_1(X)^D B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{polynomials of degree} \leqslant D} = [A, B]_D(X)$$

- A natural Factor Base: Irreducible poly in $\mathbb{F}_q[X]$ of deg $\leqslant D$.

A Small Factor Base

We have: $$\underbrace{h_1(X)^D B(X) \prod_{\alpha\in\mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{polynomials of degree} \leqslant D} = [A, B]_D(X)$$

- A natural Factor Base: Irreducible poly in $\mathbb{F}_q[X]$ of deg $\leqslant D$.
- $D \searrow \Rightarrow$ size of the factor base $\searrow \Rightarrow$ complexity of Linear Algebra $\searrow$. The smaller, the better.

A Small Factor Base

We have: $$\underbrace{h_1(X)^D B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{polynomials of degree} \leqslant D} = [A, B]_D(X)$$

- A natural Factor Base: Irreducible poly in $\mathbb{F}_q[X]$ of deg $\leqslant D$.
- $D \searrow \Rightarrow$ size of the factor base $\searrow \Rightarrow$ complexity of Linear Algebra $\searrow$. The smaller, the better.
- What is simple ? Irreducible poly in $\mathbb{F}_q[X]$ of degree $\leqslant 2$.
- Yet, lowering D rises 2 problems:
  1. Need to generate enough good equations = equations where $[A, B]_2$ splits in terms of degree $\leqslant 2$. Pb: the probability $\mathcal{P}$ to have good equations is too small w.r.t the number of equations required (need $\mathcal{P} > 1/2$).
  2. Need to be able to descend large polynomials to degree 2 ones.

A Small Factor Base: Systematic factors of $[A, B]_D$

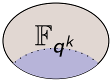
- First goal, solving pb 1: i.e. improve the probability $\mathcal{P}$.
- How ? $[A, B]_2$ is a degree 6 polynomial. The prob that it factors into degree 2 polynomials is too low.

A Small Factor Base: Systematic factors of $[A, B]_D$

- First goal, solving pb 1: i.e. improve the probability $\mathcal{P}$.
- How ? $[A, B]_2$ is a degree 6 polynomial. The prob that it factors into degree 2 polynomials is too low.
  Yet, $[A, B]_D$ has a systematic factor of degree 3 ! Namely $X\, h_1(X) - h_0(X)$.
- A degree 3 polynomial factors into terms of degree at most 2 with prob $\mathcal{P} > 2/3 > 1/2$.

A Small Factor Base: Systematic factors of $[A, B]_D$

- First goal, solving pb 1: i.e. improve the probability $\mathcal{P}$.
- How ? $[A, B]_2$ is a degree 6 polynomial. The prob that it factors into degree 2 polynomials is too low.
  Yet, $[A, B]_D$ has a systematic factor of degree 3 ! Namely $X\,h_1(X) - h_0(X)$.
- A degree 3 polynomial factors into terms of degree at most 2 with prob $\mathcal{P} > 2/3 > 1/2$.



$\Rightarrow$ Linear Algebra permits to recover the DLogs of the factor base in $O((\underbrace{\#\text{ factor base}}_{q^2})^2(\underbrace{\#\text{ of entries}}_{q})) \approx O(q^5)$ operations.

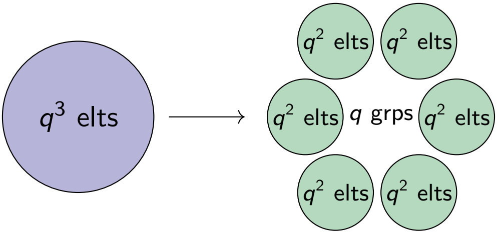
Extend the Factor Base to Degree 3

Second goal: Solving pb 2 i.e. extend the factor base to degree 3 BUT without performing linear algebra on a matrix of dim q^3.

Extend the Factor Base to Degree 3

Second goal: Solving pb 2 i.e. extend the factor base to degree 3 BUT without performing linear algebra on a matrix of dim q^3.

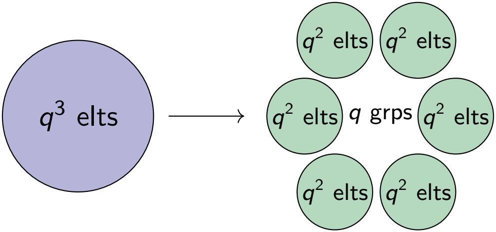
1. Divide the deg. 3 monic polynomials into groups.



Extend the Factor Base to Degree 3

Second goal: Solving pb 2 i.e. extend the factor base to degree 3 BUT without performing linear algebra on a matrix of dim q^3.

1. Divide the deg. 3 monic polynomials into groups.

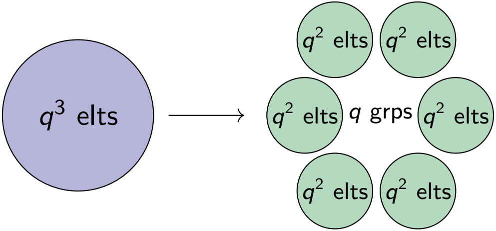


What is simple ? To consider that 2 polynomials belongs to the same group if they have the same constant coefficient.

Extend the Factor Base to Degree 3

Second goal: Solving pb 2 i.e. extend the factor base to degree 3 BUT without performing linear algebra on a matrix of dim q^3.

1. Divide the deg. 3 monic polynomials into groups.

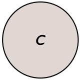


What is simple ? To consider that 2 polynomials belongs to the same group if they have the same constant coefficient.

2. Given q^2, generate equations involving only poly in q^2 and degree 1 and 2 polynomials (whose logs are already known).

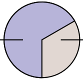
Extend the Factor Base to Degree 3

- An example: let $\textcircled{c} = \{(X^3 + c) + \alpha X^2 + \beta X | (\alpha, \beta) \in \mathbb{F}_q{}^2\}$.



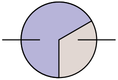
Extend the Factor Base to Degree 3

- An example: let $\textcircled{c} = \{(X^3 + c) + \alpha X^2 + \beta X | (\alpha, \beta) \in {\mathbb{F}_q}^2\}$.

Reducible  Irreducible $\Rightarrow$ new unknowns

Extend the Factor Base to Degree 3

- An example: let $\textcircled{c} = \{(X^3 + c) + \alpha X^2 + \beta X | (\alpha, \beta) \in \mathbb{F}_q{}^2\}$.

Reducible  Irreducible $\Rightarrow$ new unknowns

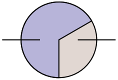
As for degree 2: set $A(X) = (X^3 + c) + \alpha X^2$ and $B(X) = (X^3 + c) + \beta X$ and create relations of the form:

$$\underbrace{h_1(X)^3 B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{all belongs to } \textcircled{c} \text{ !!}} \quad = \quad \underbrace{[A, B]_3(X)}$$

deg 8 with these A and B
$+$ deg 3 systematic factor
$+$ divisible by X

Extend the Factor Base to Degree 3

- An example: let (c) $= \{(X^3 + c) + \alpha X^2 + \beta X | (\alpha, \beta) \in \mathbb{F}_q{}^2\}$.

Reducible  Irreducible $\Rightarrow$ new unknowns

As for degree 2: set $A(X) = (X^3 + c) + \alpha X^2$ and $B(X) = (X^3 + c) + \beta X$ and create relations of the form:

$$\underbrace{h_1(X)^3 \, B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{all belongs to (c) !!}} = \underbrace{[A, B]_3(X)}_{}$$

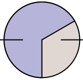
all belongs to (c) !!

deg 8 with these A and B
+ deg 3 systematic factor
+ divisible by X

Prob that $[A, B]_3$ factors into deg $\leqslant 2 \Rightarrow 41\%$. Enough !

Extend the Factor Base to Degree 3

- An example: let $\textcircled{c} = \{(X^3 + c) + \alpha X^2 + \beta X | (\alpha, \beta) \in \mathbb{F}_q{}^2\}$.

Reducible  Irreducible $\Rightarrow$ new unknowns

As for degree 2: set $A(X) = (X^3 + c) + \alpha X^2$ and $B(X) = (X^3 + c) + \beta X$ and create relations of the form:

$$\underbrace{h_1(X)^3 B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X))}_{\text{all belongs to } \textcircled{c} \text{ !!}} = \underbrace{[A, B]_3(X)}_{\substack{\text{deg 8 with these } A \text{ and } B \\ + \text{ deg 3 systematic factor} \\ + \text{ divisible by } X}}$$

Prob that $[A, B]_3$ factors into deg $\leqslant 2 \Rightarrow 41\%$. Enough !

- Complexity to recover the Dlogs of all degree 3 polynomials:

$$O((\underbrace{\# \textcircled{c}}_{q})(\underbrace{\# \text{ factor base}}_{q^2})^2(\underbrace{\# \text{ of entries}}_{q})) \approx O(q^6) \text{ ops.}$$

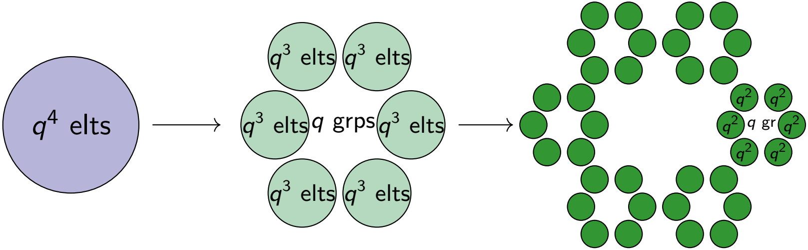
Extend the Factor Base to Degree 4

Third goal: extend the factor base to degree 4
by performing smaller linear algebra steps.

Extend the Factor Base to Degree 4

Third goal: extend the factor base to degree 4
by performing smaller linear algebra steps.

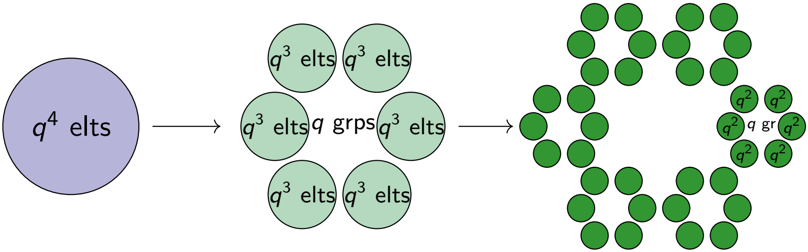
1



Extend the Factor Base to Degree 4

Third goal: extend the factor base to degree 4
by performing smaller linear algebra steps.

1



What is simple ? To consider that:

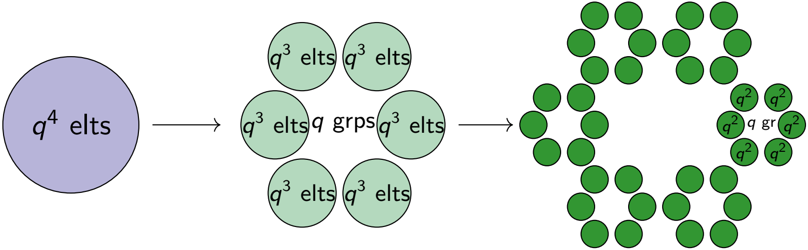
2 poly belongs to the same q^3 if same constant coefficient.

AND 2 poly belongs to the same q^2 if same coeff before X.

Extend the Factor Base to Degree 4

Third goal: extend the factor base to degree 4

1. by performing smaller linear algebra steps.



What is simple ? To consider that:

2 poly belongs to the same (q^3) if same constant coefficient.

AND 2 poly belongs to the same (q^2) if same coeff before X.

2. Given (q^2), generate equations involving only poly in it and degree 1, 2 and 3 polynomials.

Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.

Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE q^3:

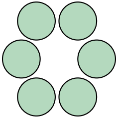
$$O((\underbrace{\# \; q^2 \text{ in } q^3}_{q}) \cdot (\underbrace{\# \; q^2}_{q^2})^2 \, \underbrace{(\#\text{entries})}_{q}) = O(q^6) \text{ ops.}$$

Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE q^3:

$$O((\underbrace{\#\ q^2 \text{ in } q^3}_{q}) \cdot (\underbrace{\#\ q^2}_{q^2})^2 \underbrace{(\#\text{entries})}_{q}) = O(q^6) \text{ ops.}$$

- Final complexity dominated by the first q^3 computation:

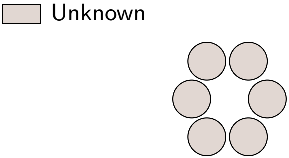


Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE $\textcircled{q^3}$:

$$O((\underbrace{\# \textcircled{q^2} \text{ in } \textcircled{q^3}}_{q}) \cdot (\underbrace{\# \textcircled{q^2}}_{q^2})^2 \underbrace{(\#\text{entries})}_{q}) = O(q^6) \text{ ops.}$$

- Final complexity dominated by the first $\textcircled{q^3}$ computation:

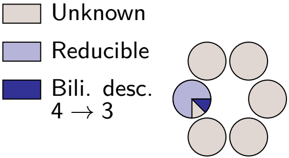


Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE (q^3):

$$O((\underbrace{\#\ (q^2)\ \text{in}\ (q^3)}_{q}) \cdot (\underbrace{\#\ (q^2)}_{q^2})^2\ \underbrace{(\#\text{entries})}_{q}) = O(q^6)\ \text{ops.}$$

- Final complexity dominated by the first (q^3) computation:

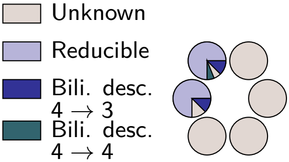


Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE q^3:

$$O((\underbrace{\# \; q^2 \text{ in } q^3}_{q}) \cdot (\underbrace{\# \; q^2}_{q^2})^2 \, \underbrace{(\#\text{entries})}_{q}) = O(q^6) \text{ ops.}$$

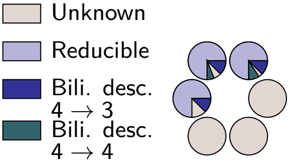
- Final complexity dominated by the first q^3 computation:



- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE q^3:

$$O((\underbrace{\#\ q^2\ \text{in}\ q^3}_{q}) \cdot (\underbrace{\#\ q^2}_{q^2})^2 \underbrace{(\#\text{entries})}_{q}) = O(q^6)\ \text{ops}.$$

- Final complexity dominated by the first q^3 computation:

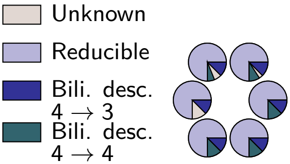


Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE q^3:

$$O((\underbrace{\# \; q^2 \text{ in } q^3}_{q}) \cdot (\underbrace{\# \; q^2}_{q^2})^2 \, \underbrace{(\#\text{entries})}_{q}) = O(q^6) \text{ ops.}$$

- Final complexity dominated by the first q^3 computation:



⇒ Final complexity of extension to deg 4 in $O(q^6)$ operations.

Extend the Factor Base to Degree 4

- How ? Previous techniques (bilinear descent from 4 to 3) + additional equations + systematic factors of $[A, B]_4$.
- Complexity of DLogs computation of ONE q^3:

$$O((\underbrace{\# \; q^2 \text{ in } q^3}_{q}) \cdot (\underbrace{\# \; q^2}_{q^2})^2 \underbrace{(\#\text{entries})}_{q}) = O(q^6) \text{ ops.}$$

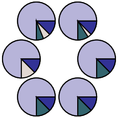
- Final complexity dominated by the first q^3 computation:

Unknown

Reducible

Bili. desc. $4 \rightarrow 3$

Bili. desc. $4 \rightarrow 4$

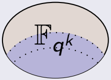


$\Rightarrow$ Final complexity of extension to deg 4 in $O(q^6)$ operations.

Main Result

Final asymptotic complexity of the three first phases:

$O(q^6)$ operations – to be compared with previous $O(q^7)$.



Individual Logarithms (Descent strategies)

- Continued fractions (high degrees)

Individual Logarithms (Descent strategies)

- Continued fractions (high degrees)
- Classical descent (for high to mid degrees, need subfield)

Individual Logarithms (Descent strategies)

- Continued fractions (high degrees)
- Classical descent (for high to mid degrees, need subfield)
- Bilinear descent (for mid to low degrees)

Individual Logarithms (Descent strategies)

- Continued fractions (high degrees)
- Classical descent (for high to mid degrees, need subfield)
- Bilinear descent (for mid to low degrees)
- Quasi-polynomial descent (all degrees)

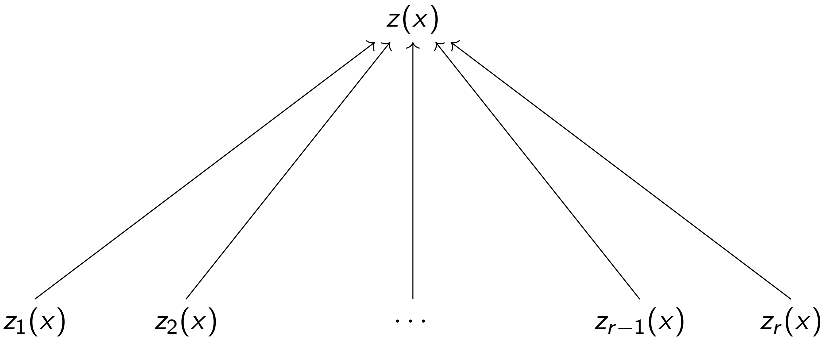
Individual Logarithms (Descent strategies)

- Continued fractions (high degrees)
- Classical descent (for high to mid degrees, need subfield)
- Bilinear descent (for mid to low degrees)
- Quasi-polynomial descent (all degrees)
- ZigZag descent (all even degrees)

General principle

- Given target $z(x)$ in finite field, write:

$$z(x) = \prod_i z_i(x)^{e_i}, \quad \text{with smaller } z_i\text{s}$$



Classical descent

- Need two variables x and y

Classical descent

- Need two variables x and y
- If $q = p^{\ell}$, let:

$$\begin{aligned} y &= x^{p^{\ell_1}} \qquad \text{then} \\ y^{p^{\ell_2}} &= x^{p^{\ell}} = \frac{h_0(x)}{h_1(x)}. \end{aligned}$$

Classical descent

- Need two variables x and y
- If $q = p^\ell$, let:

$$\begin{aligned} y &= x^{p^{\ell_1}} \qquad \text{then} \\ y^{p^{\ell_2}} &= x^{p^\ell} = \frac{h_0(x)}{h_1(x)}. \end{aligned}$$

- Let $F(x, y)$ be a (low degree) bivariate polynomial in $\mathbb{F}_q[x, y]$, then:

$$F(x, x^{p^{\ell_1}})^{p^{\ell_2}} = F(x^{p^{\ell_2}}, h_0(x)/h_1(x)) \quad \text{in } \mathbb{F}_{q^k}.$$

Classical descent

- Need two variables x and y
- If $q = p^{\ell}$, let:

$$\begin{aligned} y &= x^{p^{\ell_1}} \qquad \text{then} \\ y^{p^{\ell_2}} &= x^{p^{\ell}} = \frac{h_0(x)}{h_1(x)}. \end{aligned}$$

- Let $F(x, y)$ be a (low degree) bivariate polynomial in $\mathbb{F}_q[x, y]$, then:

$$F(x, x^{p^{\ell_1}})^{p^{\ell_2}} = F(x^{p^{\ell_2}}, h_0(x)/h_1(x)) \quad \text{in } \mathbb{F}_{q^k}.$$

- Force $z(x)$ as divisor of $F(x, x^{p^{\ell_1}})$ or $F(x^{p^{\ell_2}}, h_0(x)/h_1(x))$ (linear algebra)

Classical descent

- Need two variables x and y
- If $q = p^\ell$, let:

$$\begin{aligned} y &= x^{p^{\ell_1}} \qquad \text{then} \\ y^{p^{\ell_2}} &= x^{p^\ell} = \frac{h_0(x)}{h_1(x)}. \end{aligned}$$

- Let $F(x, y)$ be a (low degree) bivariate polynomial in $\mathbb{F}_q[x, y]$, then:

$$F(x, x^{p^{\ell_1}})^{p^{\ell_2}} = F(x^{p^{\ell_2}}, h_0(x)/h_1(x)) \quad \text{in } \mathbb{F}_{q^k}.$$

- Force $z(x)$ as divisor of $F(x, x^{p^{\ell_1}})$ or $F(x^{p^{\ell_2}}, h_0(x)/h_1(x))$ (linear algebra)
- Low arity in descent but can't go very low

Modern descent strategies

- Remember basic Equation:

$$h_1(X)^D B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X)) = [A, B]_D(X)$$

Modern descent strategies

- Remember basic Equation:

$$h_1(X)^D B(X) \prod_{\alpha \in \mathbb{F}_q} (A(X) - \alpha B(X)) = [A, B]_D(X)$$

- Make $z(x)$ appear on the right or left
  - On the right: bilinear descent
  - On the left: quasi-polynomial
  - On the right (powers of two): ZigZag descent [GKZ14]

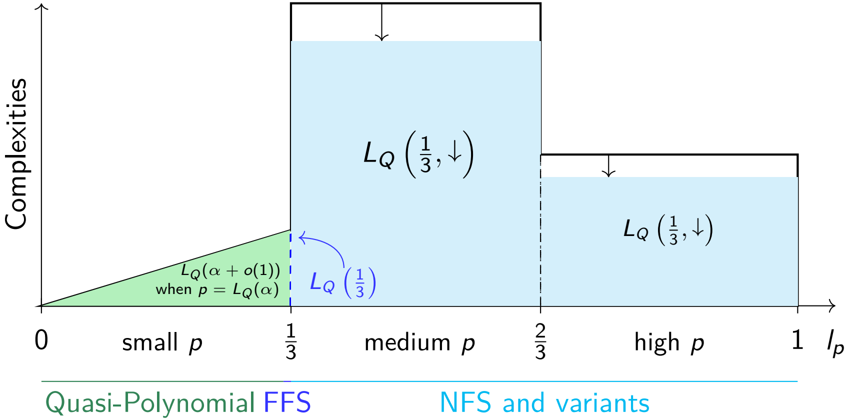
Descent Tree

- Continued fractions, **at most one application**
- Classical descent, **many levels possible**
- Bilinear descent (or [GKZ14]), **in practice 4-5 levels max.**
- Quasi-polynomial descent **in practice 2 levels max.**

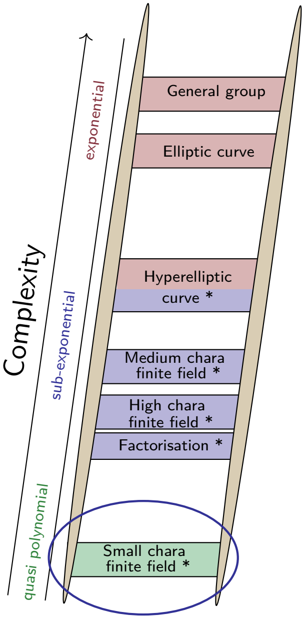
Practical application

- Record in characteristic 3 on $\mathbb{F}_{3^{5 \cdot 479}}$, a finite field of cardinality a 3796-bit integer.
  - Not a special extension field such as Kummer extension !
  - Make use of the Dual Frobenius Representation combined with the useful variant (both not presented here).
- To be compared with previous record in characteristic 3 by Adj, Menezes, Oliveira and Rodriguez-Henriquez on a 1551-bit finite field.
- Time : 8600 CPU-hours $\approx$ 1 CPU-year

Complexities of Index Calculus Algorithms



Questions ?


Complexity
exponential
sub-exponential
quasi polynomial
General group
Elliptic curve
Hyperelliptic
curve *
Medium chara
finite field *
High chara
finite field *
Factorisation *
Small chara
finite field *