Discrete logarithm in finite fields of small characteristic

September 18, 2017

CONTENTS

Introduction

CONTEXT

- $G = \langle g \rangle$ cyclic group generated by g
- $N = \operatorname{Card} G$.

We have an *isomorphism*

$$\begin{array}{rccc} \exp_g : & (\mathbb{Z}/N\mathbb{Z}, +) & \to & (G, \times) \\ & n & \mapsto & g^n \end{array}$$

- $\log_g := \exp_g^{-1}$
- Compute g^n from n: **easy** (polynomial in $\log n$)
- Compute n from g^n: **hard** (discrete logarithm problem)

DEFINITIONS

Two families of algorithms:

- The *generic* algorithms (complexity: $O(\sqrt{N})$)
- The *index calculus* algorithms, using group structure
  - from now on: $G = \mathbb{F}_{q^n}^\times$

Terminology:

- *small characteristic*: $\mathbb{F}_{q^n}$ with $q \ll q^n$
- *quasi-polynomial* complexity: $\ell^{O(\log \ell)}$ where $\ell = \log(q^n)$
- *Notation:*

$$L_N(\alpha) = \exp((c + o(1))(\log N)^\alpha (\log \log N)^{1-\alpha})$$

$$L_N(0) = (\log N)^{c+o(1)} \qquad L_N(1) = N^{c+o(1)}$$

HISTORICAL BACKGROUND

- First appearance in [Diffie, Hellman '76]
- First sub-exponential algorithm in $\mathbb{Z}/p\mathbb{Z}$ [Adleman '79]: $L_N(1/2)$
- Between 1984 and 2006: algorithms in $L_N(1/3)$

And more recently, in finite fields of small characteristic:

- New algorithm with $L_N(1/4)$ complexity [Joux '13]
- *Quasi-polynomial* algorithm [Barbulescu, Gaudry, Joux, Thomé '14]
- Second quasi-polynomial algorithm [Granger, Kleinjung, Zumbrägel '14]

SOME PRACTICAL RECORDS

Date	Field	Bitsize	Algorithm	Authors
2013/02	2^{1778}	1778	$L(1/4)$	Joux
2013/02	2^{1991}	1991	GKZ	Göloglu, Granger, McGuire, Zumbrägel
2013/05	2^{6168}	6168	$L(1/4)$	Joux
2014/01	$3^{6\cdot 137}$	1303	$L(1/4)$, GKZ	Adj, Menezes, Oliveira, Rodríguez-Henríquez
2014/01	2^{9234}	9234	$L(1/4)$, GKZ	Granger, Kleinjung, Zumbrägel
2014	$3^{6\cdot 509}$	4034	$L(1/4)$, GKZ	Adj, Menezes, Oliveira, Rodríguez-Henríquez

- Runtime of the last computation: 220 CPU years

INDEX CALCULUS

Overview

Goal: find $\log_g(h)$

0. first choose $F \subset G$ with $\langle F \rangle = G$
1. find multiplicative relations between elements in F
2. solve the associated linear system for $\{\log_g(f) \mid f \in F\}$
3. express h as a product of elements in F

Steps 1 and 3 depend on the structure of the finite field, different fields give different complexities for the index calculus.

AN EXAMPLE: ADLEMAN

Context:

- $G = \mathbb{F}_p^\times = \langle g \rangle$ for a prime p and $N = |G| = p - 1$
- $F = \{f \mid f \leq B,\ f \text{ prime}\}$ for a chosen integer B

Step 1: relations generation

- randomly choose $e \in \mathbb{Z}/N\mathbb{Z}$
- test if g^e is B-smooth
- if it is the case, it yields a relation in G:

$$g^e = \prod_{f \in F} f^{e_f}, e_f \in \mathbb{N}$$

that can be written

$$e = \sum_{f \in F} e_f \log_g(f).$$

AN EXAMPLE

Step 2: linear algebra: solve the linear system
Step 3: express h as a function of the elements in F:

- randomly choose $e \in \mathbb{Z}/N\mathbb{Z}$
- test if hg^e is B-smooth
- if it is the case, it yields a relation:

$$\log_g(h) = \sum_{f \in F} e_f \log_g(f) - e$$

Depends on B:

- B large: easier to find relations
- B large: need more relations to solve the system

Complexity: $L_N(1/2)$

QUASI-POLYNOMIAL ALGORITHMS

COMPARISON WITH ADLEMAN

Algorithm	Adleman	Quasi-polynomial
Field	$\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$	$\mathbb{F}_{q^n} \cong \mathbb{F}_q[X]/(P)$
Elements repr. by	integers	polynomials
Subset $F \subset G$	primes $\leq B$	irreducibles of degree $\leq B$

Goal: find polynomials that factor into irreducible polynomials of small degree.

MAIN IDEAS

1. Use homographies:
   - Q polynomial that factors nicely
   - $m(X) = \frac{aX+b}{cX+d}$ an homography

   $m \cdot Q = (cX+d)^{\deg Q} Q(\frac{aX+b}{cX+d})$ factors nicely too

2. Use Idea 1 on $X^q - X$ (factors into linear polynomials)

3. Take advantage of the freedom for the defining polynomial P of $\mathbb{F}_{q^n} = \mathbb{F}_q[X]/(P)$ to simplify X^q
   - We choose $P \mid h_1 X^q - h_0$ with h_0, h_1 polynomials of small degree (existence of h_0, h_1 **heuristic**)
   - We obtain $X^q \equiv \frac{h_0(X)}{h_1(X)}$ in $\mathbb{F}_q[X]/(P)$

QUASI-POLYNOMIAL COMPLEXITY

Goal: $\mathbb{F}_{q^n}$ field, find $\log Q$ for $Q \in \mathbb{F}_{q^n}$

- not able to express $\log Q$ as elements in F in one step

Proposition

Let Q be an element of a field $\mathbb{F}_{q^n}$. There exists a ***heuristic*** *algorithm whose complexity is polynomial in $\ell = \log(q^n)$ which returns an expression of $\log Q$ as a linear combination of at most $\mathcal{N}$ logarithms $\log Q_j$ with $\deg Q_j \leq \lceil \frac{1}{2} \deg Q \rceil$.*

- Barbulescu, Gaudry, Joux, Thomé: n even and $\mathcal{N} = O(q^2 \frac{n}{2})$
- Granger, Kleinjung, Zumbrägel: $\mathcal{N} = q + 2$

The descent

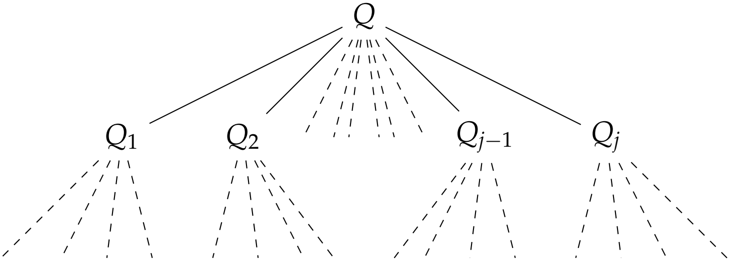

Figure: The descent tree

- Each node: one application of the Proposition (polynomial time algorithm), number of nodes = arity$^{\text{depth}}$
- arity $= \mathcal{N} = O(\ell^{O(1)})$, depth $= O(\log(\ell))$
- Complexity: $\ell^{O(\log \ell)}$

Barbulescu, Gaudry, Joux and Thomé

Context:

- $\mathbb{K} = \mathbb{F}_{q^{2m}} = \mathbb{F}_{q^2}[X]/(P)$ with P irreducible polynomial of degree m dividing $h_1 X^q - h_0$, $\deg h_i \leq 2$.
- $F = \{\text{degree one polynomials}\} \bigcup \{h_1\}$

Descent: based on the equation

$$X^q Y - X Y^q = Y \prod_{a \in \mathbb{F}_q} (X - aY) \tag{1}$$

- Substitute X by $aQ + b$ and Y by $cQ + d$ in (1)
- Combine these equations to express Q as polynomials of degree $\leq \lceil \frac{1}{2} \deg Q \rceil$ (**linear algebra**)

GRANGER, KLEINJUNG AND ZUMBRÄGEL

Context:

- $\mathbb{K} = \mathbb{F}_{q^n} = \mathbb{F}_q[X]/(P)$ with P irreducible polynomial of degree n dividing $h_1 X^q - h_0$.

Descent: $Q \in \mathbb{F}_q[X]$ irreducible of degree $2d$

- Factor Q in $\mathbb{F}_{q^d}$: $Q = \prod_{j=0}^{d-1} Q_j$ with $\deg Q_j = 2$
- Express each Q_j as $q+2$ linear polynomials $R_{j,k}$ in $\mathbb{F}_{q^d}[X]$ (this is called “on-the-fly” elimination)
- Q is expressed as $q+2$ polynomials $N_k = \prod_{j=0}^{d-1} R_{j,k}$

Fact: The polynomials N_k are of degree d in $\mathbb{F}_q[X]$ (not $\mathbb{F}_{q^d}[X]$).

"On the fly" elimination

Input:

- $Q \in \mathbb{F}_{q^m}[X]$ and $\deg Q = 2$

Output:

- $Q \rightsquigarrow q+2$ polynomials Q_i with $\deg Q_i = 1$ and $Q_i \in \mathbb{F}_{q^m}[X]$

Ideas

- Find (a, b, c) such that
  1. $\Delta = X^{q+1} + aX^q + bX + c$ splits into linear factors in $\mathbb{F}_{q^m}[X]$
  2. $Q \mid (X+a)h_0 + (bX+c)h_1$ (degree 3)

  (can be reduced to **root finding**)
- $\Delta = \frac{1}{h_1}((X+a)h_0 + (bX+c)h_1) \mod P$
- $h_1\Delta = QL \mod P$

EXPERIMENTAL RESULTS

Implementation in Julia (based on Nemo/Flint):

Algorithm	BGJT	GKZ
Bottleneck	linear algebra	root finding
arity ($\mathbb{F}_{17^{2\cdot 17}}$)	794	291
descent runtime ($\mathbb{F}_{17^{2\cdot 17}}$) (seconds)	9.6	34
arity ($\mathbb{F}_{23^{2\cdot 23}}$)	1477	531
descent runtime ($\mathbb{F}_{23^{2\cdot 23}}$) (seconds)	80	140
arity ($\mathbb{F}_{29^{2\cdot 29}}$)	2360	843
descent runtime ($\mathbb{F}_{29^{2\cdot 29}}$) (seconds)	570	400

Open problems:

- practical improvements
- proofs of heuristics
- polynomial (heuristic) algorithm