Root finding over finite fields

Bruno Grenet
LIRMM
Université de Montpellier

Joris van der Hoeven & Grégoire Lecerf
CNRS – LIX
École polytechnique

GT MC2 — April 8., 2015

Root finding over finite fields

Given $f \in \mathbb{F}_q[X]$, compute its roots, that is $\{\alpha \in \mathbb{F}_q : f(\alpha) = 0\}$.

Root finding over finite fields

Given $f \in \mathbb{F}_q[X]$, compute its roots, that is $\{\alpha \in \mathbb{F}_q : f(\alpha) = 0\}$.

- Building block for many algorithms in computer algebra: root finding over $\mathbb{Z}$, factorization, sparse interpolation, ...
- Applications in cryptography, error correcting codes, ...
- Derandomization
- Sparse interpolation: bottleneck in practice

[van der Hoeven & Lecerf, 2014]

$\mathbb{F}_q$: field with q elements, $q = p^r$ for some prime number p

- $\mathbb{F}_p \simeq \mathbb{Z}/p\mathbb{Z}$; $+, -, \times$ and / **modulo p**
- $\mathbb{F}_q \simeq \mathbb{F}_p[\lambda]/\langle\phi\rangle$ ($\phi \in \mathbb{F}_p[\lambda]$ irreducible of degree r); $+, -, \times$ and / **modulo p and ϕ**

$\mathbb{F}_q$: field with q elements, $q = p^r$ for some prime number p

- $\mathbb{F}_p \simeq \mathbb{Z}/p\mathbb{Z}$; $+, -, \times$ and / **modulo p**
- $\mathbb{F}_q \simeq \mathbb{F}_p[\lambda]/\langle\phi\rangle$ ($\phi \in \mathbb{F}_p[\lambda]$ irreducible of degree r); $+, -, \times$ and / **modulo p and ϕ**

- $\mathbb{F}_3 = \{0, 1, 2\}$:

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

and

×	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

;

$\mathbb{F}_q$: field with q elements, $q = p^r$ for some prime number p

- $\mathbb{F}_p \simeq \mathbb{Z}/p\mathbb{Z}$; $+, -, \times$ and / **modulo p**
- $\mathbb{F}_q \simeq \mathbb{F}_p[\lambda]/\langle\phi\rangle$ ($\phi \in \mathbb{F}_p[\lambda]$ irreducible of degree r); $+, -, \times$ and / **modulo p and ϕ**

- $\mathbb{F}_3 = \{0, 1, 2\}$:

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

and

×	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

;

- $\mathbb{F}_4 = \mathbb{F}_2[\lambda]/\langle\lambda^2 + \lambda + 1\rangle = \{0, 1, \lambda, \lambda + 1\}$:

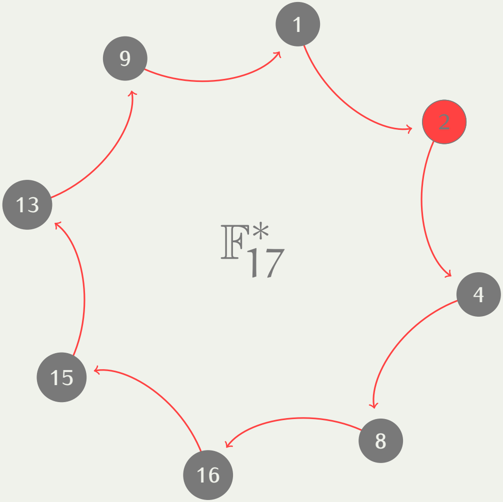
+	1	λ	$\lambda+1$
1	0	$\lambda+1$	λ
λ	$\lambda+1$	0	1
$\lambda+1$	λ	1	0

and

×	λ	$\lambda+1$
λ	$\lambda+1$	1
$\lambda+1$	1	λ

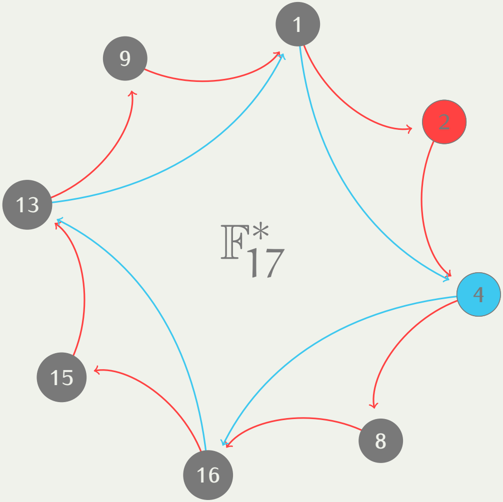
.

Multiplicative structure



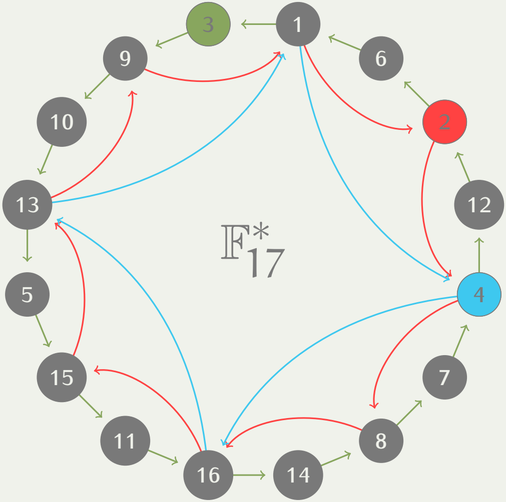
2 is a primitive root of unity of order 8

Multiplicative structure



2 is a primitive root of unity of order 8

4 is a primitive root of unity of order 4

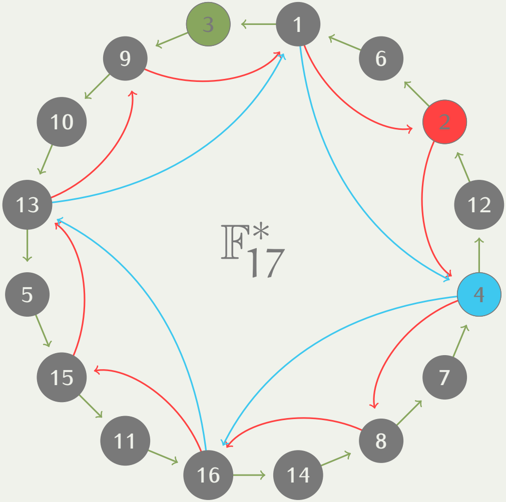


2 is a primitive root of unity of order 8

4 is a primitive root of unity of order 4

3 is a primitive element of $\mathbb{F}_{17}^*$: $\mathbb{F}_{17}^* = \{3^i : 0 \leqslant i < 16\}$

Multiplicative structure



2 is a primitive root of unity of order 8

4 is a primitive root of unity of order 4

3 is a primitive element of $\mathbb{F}_{17}^*$: $\mathbb{F}_{17}^* = \{3^i : 0 \leqslant i < 16\}$

$$\alpha^{q-1} = \zeta^{i(q-1)} = 1$$

$$X^{q-1} - 1 = \prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha)$$

Theorem

The roots of $f \in \mathbb{F}_q[X]$ can be computed in **deterministic time** $\mathrm{poly}(q, d_f)$.

- **Algo:** Test each $\alpha \in \mathbb{F}_q$, sequentially.

Theorem

The roots of $f \in \mathbb{F}_q[X]$ can be computed in **deterministic time** $\mathrm{poly}(q, d_f)$.

- **Algo:** Test each $\alpha \in \mathbb{F}_q$, sequentially.

⚠ Input size $(1 + d_f) \log q$: exponential time!

Theorem

The roots of $f \in \mathbb{F}_q[X]$ can be computed in **deterministic time** $\mathrm{poly}(q, d_f)$.

- **Algo:** Test each $\alpha \in \mathbb{F}_q$, sequentially.

⚠ Input size $(1 + d_f)\log q$: exponential time!

- Randomization: expected time $\dfrac{d_f}{d_f + 1} q$.

Objectives

Obtain **fast** algorithms for polynomial root finding in finite fields.

- Deterministic, probabilistic, heuristic; in practice or in theory.

Objectives

Obtain **fast** algorithms for polynomial root finding in finite fields.

- Deterministic, probabilistic, heuristic; in practice or in theory.
- **Assumption:** f has d_f **distinct** and **nonzero** roots.

  (easy reduction: $f \leftarrow \gcd(f, X^{q-1} - 1)$)

Objectives

Obtain **fast** algorithms for polynomial root finding in finite fields.

- Deterministic, probabilistic, heuristic; in practice or in theory.
- **Assumption:** f has d_f **distinct** and **nonzero** roots.

  (easy reduction: $f \leftarrow \gcd(f, X^{q-1} - 1)$)
- **Extra input:** A primitive element ζ, or a primitive root of unity ξ.

Objectives

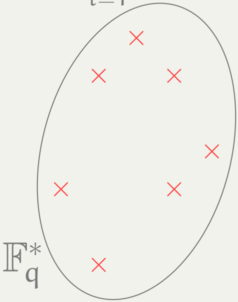
Obtain **fast** algorithms for polynomial root finding in finite fields.

- Deterministic, probabilistic, heuristic; in practice or in theory.
- **Assumption:** f has d_f **distinct** and **nonzero** roots.

  (easy reduction: $f \leftarrow \gcd(f, X^{q-1} - 1)$)
- **Extra input:** A primitive element ζ, or a primitive root of unity ξ.
- **Smooth cardinality:**
  - $q = \rho\pi_1 \cdots \pi_m + 1$, where ρ, π_1, …, π_m are *small*;
  - Practical purpose: $q = M2^m + 1$ is a *FFT prime*.

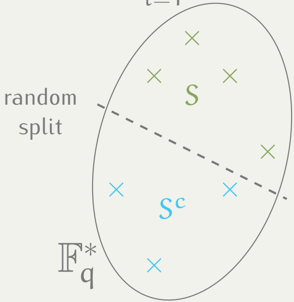
A (slow) recursive algorithm

$$f = \prod_{i=1}^{d} (X - \alpha_i)$$



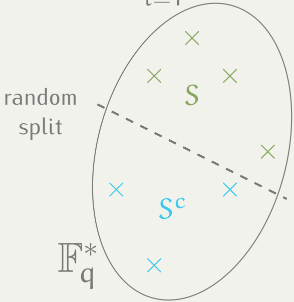
A (slow) recursive algorithm

$$f = \prod_{i=1}^{d} (X - \alpha_i)$$



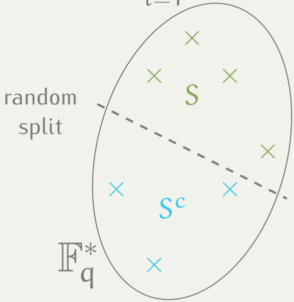
A (slow) recursive algorithm

$$f = \prod_{i=1}^{d}(X - \alpha_i) \qquad u = \prod_{\alpha\in S}(X-\alpha)$$



A (slow) recursive algorithm

$$f = \prod_{i=1}^{d} (X - \alpha_i) \qquad u = \prod_{\alpha \in S} (X - \alpha)$$

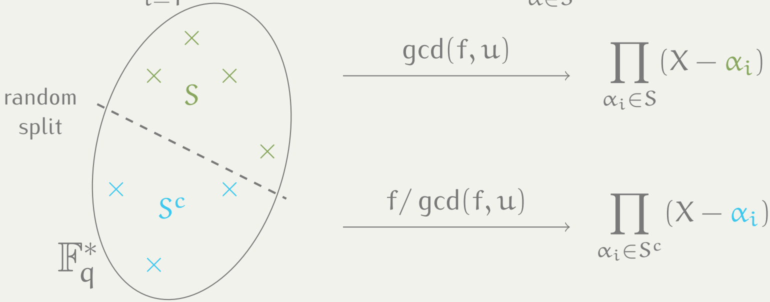


$$\xrightarrow{\gcd(f,u)} \prod_{\alpha_i \in S} (X - \alpha_i)$$

$$\xrightarrow{f/\gcd(f,u)} \prod_{\alpha_i \in S^c} (X - \alpha_i)$$

A (slow) recursive algorithm

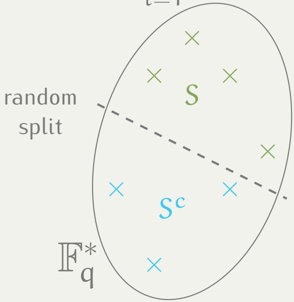
$$f = \prod_{i=1}^{d} (X - \alpha_i) \qquad u = \prod_{\alpha \in S} (X - \alpha)$$



$$\mathbb{P}[\gcd(f, u) \in \{1, f\}] = \mathbb{P}[\forall i, \alpha_i \in S] + \mathbb{P}[\forall i, \alpha_i \in S^c] = 1/2^{d-1}$$

A (slow) recursive algorithm

$$f = \prod_{i=1}^{d}(X - \alpha_i) \qquad u = \prod_{\alpha \in S}(X - \alpha)$$



$$\xrightarrow{\gcd(f, u)} \prod_{\alpha_i \in S}(X - \alpha_i)$$

$$\xrightarrow{f/\gcd(f, u)} \prod_{\alpha_i \in S^c}(X - \alpha_i)$$

$$\mathbb{P}[\gcd(f, u) \in \{1, f\}] = \mathbb{P}[\forall i, \alpha_i \in S] + \mathbb{P}[\forall i, \alpha_i \in S^c] = 1/2^{d-1}$$

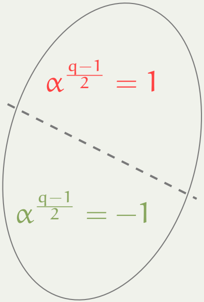
Good and bad news

The expected number of calls is $2d$, but the complexity is $\tilde{O}(q)$.

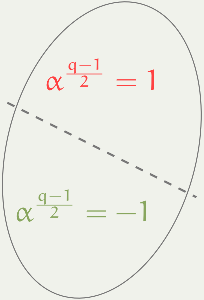
- $\prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha) = X^{q-1} - 1$

- $\prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha) = X^{q-1} - 1 = (X^{\frac{q-1}{2}} - 1)(X^{\frac{q-1}{2}} + 1)$ (q odd)

- $\prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha) = X^{q-1} - 1 = (X^{\frac{q-1}{2}} - 1)(X^{\frac{q-1}{2}} + 1)$ (q odd)



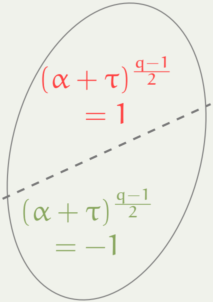
- $\prod_{\alpha\in\mathbb{F}_q^*}(X-\alpha) = X^{q-1}-1 = (X^{\frac{q-1}{2}}-1)(X^{\frac{q-1}{2}}+1)$ (q odd)



- With some luck, $\gcd(f, X^{\frac{q-1}{2}}-1) \notin \{1, f\}$.

Cantor-Zassenhaus' algorithm

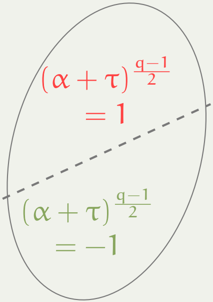
- $\prod_{\alpha\in\mathbb{F}_q^*}(X-\alpha) = X^{q-1}-1 = (X^{\frac{q-1}{2}}-1)(X^{\frac{q-1}{2}}+1)$ (q odd)



- With some luck, $\gcd(f, X^{\frac{q-1}{2}}-1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X+\tau)^{\frac{q-1}{2}}-1)$ for some **random** $\tau \in \mathbb{F}_q$

Cantor-Zassenhaus' algorithm

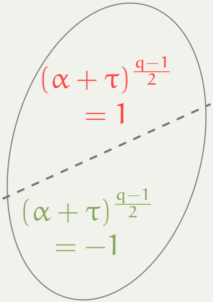
- $\prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha) = X^{q-1} - 1 = (X^{\frac{q-1}{2}} - 1)(X^{\frac{q-1}{2}} + 1)$ (q odd)



- With some luck, $\gcd(f, X^{\frac{q-1}{2}} - 1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X + \tau)^{\frac{q-1}{2}} - 1)$ for some **random** $\tau \in \mathbb{F}_q$

$$\mathbb{P}_{\tau \in \mathbb{F}_q}\left[\gcd(f, X + \tau)^{\frac{q-1}{2}} - 1) \notin \{1, f\}\right] = \frac{q-1}{2q}$$

- $\prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha) = X^{q-1} - 1 = (X^{\frac{q-1}{2}} - 1)(X^{\frac{q-1}{2}} + 1)$ (q odd)



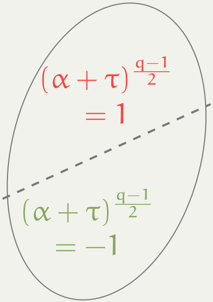
- With some luck, $\gcd(f, X^{\frac{q-1}{2}} - 1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X + \tau)^{\frac{q-1}{2}} - 1)$ for some **random** $\tau \in \mathbb{F}_q$

$$\mathbb{P}_{\tau \in \mathbb{F}_q} \left[\gcd(f, X + \tau)^{\frac{q-1}{2}} - 1) \notin \{1, f\} \right] = \frac{q-1}{2q}$$

- $\gcd(f, (X + \tau)^{\frac{q-1}{2}} - 1)$ in time $\tilde{O}(d \log q)$

Cantor-Zassenhaus' algorithm

- $\prod_{\alpha \in \mathbb{F}_q^*} (X - \alpha) = X^{q-1} - 1 = (X^{\frac{q-1}{2}} - 1)(X^{\frac{q-1}{2}} + 1)$ (q odd)



- With some luck, $\gcd(f, X^{\frac{q-1}{2}} - 1) \notin \{1, f\}$.
- Push your luck: $\gcd(f, (X + \tau)^{\frac{q-1}{2}} - 1)$ for some **random** $\tau \in \mathbb{F}_q$

$$\mathbb{P}_{\tau \in \mathbb{F}_q}\left[\gcd(f, X + \tau)^{\frac{q-1}{2}} - 1) \notin \{1, f\}\right] = \frac{q-1}{2q}$$

- $\gcd(f, (X + \tau)^{\frac{q-1}{2}} - 1)$ in time $\tilde{O}(d \log q)$

Randomized algorithm

The roots of $f \in \mathbb{F}_q[X]$ can be computed in time $\tilde{O}(d \log^2 q)$.

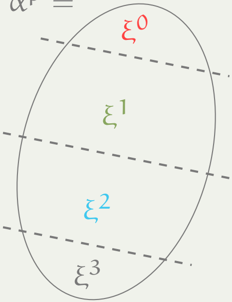
Modified Cantor-Zassenhaus' algorithm
(for smooth cardinality)

Let $q = \chi\rho + 1$. Then $X^{q-1} - 1 = \prod_{i=0}^{\chi-1}(X^{\rho} - \xi^{i})$, where $\xi^{\chi} = 1$.

Modified Cantor-Zassenhaus' algorithm
(for smooth cardinality)

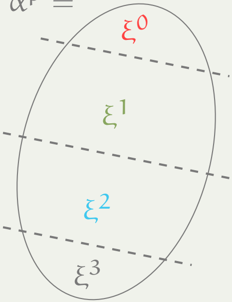
Let $q = \chi\rho + 1$. Then $X^{q-1} - 1 = \prod_{i=0}^{\chi-1}(X^\rho - \xi^i)$, where $\xi^\chi = 1$.

$\alpha^p =$



Let $q = \chi\rho + 1$. Then $X^{q-1} - 1 = \prod_{i=0}^{\chi-1}(X^\rho - \xi^i)$, where $\xi^\chi = 1$.

$\alpha^p =$



$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^0)$

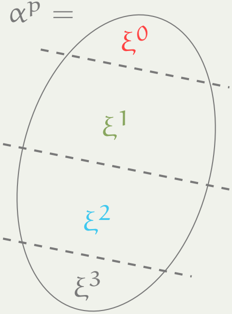
$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^1)$

$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^2)$

$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^3)$

Modified Cantor-Zassenhaus' algorithm
(for smooth cardinality)

Let $q = \chi\rho + 1$. Then $X^{q-1} - 1 = \prod_{i=0}^{\chi-1} (X^\rho - \xi^i)$, where $\xi^\chi = 1$.



$$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^0)$$

$$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^1)$$

$$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^2)$$

$$\longrightarrow \gcd(f, (X+\tau)^\rho - \xi^3)$$

If $\chi \ll \log q / \log d$, the **speed-up is approximately** $\log_2 \chi$.

The (generalized) Graeffe transform

Definition

The **Graeffe transform** of $g \in \mathbb{F}_q[X]$ is the unique polynomial $h \in \mathbb{F}_q[X]$ such that

$$h(X^2) = g(X)g(-X).$$

If $g(X) = \prod_i (\alpha_i - X)$, then $h(X) = \prod_i (\alpha_i^2 - X)$.

The (generalized) Graeffe transform

Definition

The **Graeffe transform** of $g \in \mathbb{F}_q[X]$ is the unique polynomial $h \in \mathbb{F}_q[X]$ such that

$$h(X^2) = g(X)g(-X).$$

If $g(X) = \prod_i(\alpha_i - X)$, then $h(X) = \prod_i(\alpha_i^2 - X)$.

The **generalized Graeffe transform** of $g \in \mathbb{F}_q[X]$ of order π is

$$G_\pi(g)(X) = (-1)^{\pi \deg g} \operatorname{res}_z(g(z), z^\pi - x).$$

If $g = \prod_i(\alpha_i - X)$, then $G_\pi(g)(X) = \prod_i(\alpha_i^\pi - X)$.

The (generalized) Graeffe transform

Definition

The **Graeffe transform** of $g \in \mathbb{F}_q[X]$ is the unique polynomial $h \in \mathbb{F}_q[X]$ such that

$$h(X^2) = g(X)g(-X).$$

If $g(X) = \prod_i (\alpha_i - X)$, then $h(X) = \prod_i (\alpha_i^2 - X)$.

The **generalized Graeffe transform** of $g \in \mathbb{F}_q[X]$ of order π is

$$G_\pi(g)(X) = (-1)^{\pi \deg g} \operatorname{res}_z(g(z), z^\pi - x).$$

If $g = \prod_i (\alpha_i - X)$, then $G_\pi(g)(X) = \prod_i (\alpha_i^\pi - X)$.

Note. $G_{\pi_1 \pi_2} = G_{\pi_1} \circ G_{\pi_2}$

Let $q = \rho\pi_1 \cdots \pi_m + 1$.

Let $q = \rho\pi_1 \cdots \pi_m + 1$.

Remark

$G_{q-1}(g)(X) = \pm \prod_i (X - \alpha_i^{q-1}) = \pm(X-1)^{\deg(g)}$

Let $q = \rho\pi_1 \cdots \pi_m + 1$.

Remark

$G_{q-1}(g)(X) = \pm\prod_i (X - \alpha_i^{q-1}) = \pm(X-1)^{\deg(g)}$

$$f \xrightarrow{G_\rho} h_0 \xrightarrow{G_{\pi_1}} h_1 \xrightarrow{G_{\pi_2}} \cdots \xrightarrow{G_{\pi_{m-1}}} h_{m-1} \xrightarrow{G_{\pi_m}} h_m$$

Let $q = \rho\pi_1 \cdots \pi_m + 1$.

Remark

$G_{q-1}(g)(X) = \pm \prod_i (X - \alpha_i^{q-1}) = \pm(X-1)^{\deg(g)}$

$$f \xrightarrow{G_\rho} h_0 \xrightarrow{G_{\pi_1}} h_1 \xrightarrow{G_{\pi_2}} \cdots \xrightarrow{G_{\pi_{m-1}}} h_{m-1} \xrightarrow{G_{\pi_m}} h_m$$

$$h_m \longmapsto \{1\}$$

Using Graeffe transforms

Let $q = \rho\pi_1 \cdots \pi_m + 1$.

Remark

$G_{q-1}(g)(X) = \pm \prod_i (X - \alpha_i^{q-1}) = \pm (X-1)^{\deg(g)}$

$$\begin{array}{ccccccccccc}
f & \xrightarrow{G_\rho} & h_0 & \xrightarrow{G_{\pi_1}} & h_1 & \xrightarrow{G_{\pi_2}} & \cdots & \xrightarrow{G_{\pi_{m-1}}} & h_{m-1} & \xrightarrow{G_{\pi_m}} & h_m \\
 & & & & & & & & & & \downarrow \\
Z(f) & \longleftarrow & Z_0 & \longleftarrow & Z_1 & \longleftarrow & \cdots & \longleftarrow & Z_{m-1} & \longleftarrow & \{1\}
\end{array}$$

Lemma

Let π divide $q-1$, and ξ a primitive root of unity of order π. Then

$$G_\pi(g)(X^\pi) = g(X)g(\xi X)\cdots g(\xi^{\pi-1}X).$$

Lemma

Let π divide $q-1$, and ξ a primitive root of unity of order π. Then

$$G_\pi(g)(X^\pi) = g(X)g(\xi X)\cdots g(\xi^{\pi-1}X).$$

Theorem

Given $g \in \mathbb{F}_q[X]$ and a primitive root of unity ξ of order π, $G_\pi(g)$ can be computed in $\tilde{O}(\pi d \log q)$ operations.

Improved Graeffe transform computation

Theorem

Let $g \in \mathbb{F}_q[X]$ of degree d. For all $\delta > 0$ such that $d^{1+\delta} \leqslant q - 1$, $G_\pi(g)$ can be computed in time $(d \log q)^{1+\delta} + \tilde{O}(d \log q \log \pi)$.

Improved Graeffe transform computation

Theorem

Let $g \in \mathbb{F}_q[X]$ of degree d. For all $\delta > 0$ such that $d^{1+\delta} \leqslant q - 1$, $G_\pi(g)$ can be computed in time $(d \log q)^{1+\delta} + \tilde{O}(d \log q \log \pi)$.

Based on:

Theorem **[Kedlaya-Umans'11]**

Let $f, g, h \in \mathbb{F}_q[X]$ of degree d. For all $\delta > 0$, $(f \circ g \mod h)$ can be computed in time $d^{1+\delta}\tilde{O}(\log q)$.

Improved Graeffe transform computation

Theorem

Let $g \in \mathbb{F}_q[X]$ of degree d. For all $\delta > 0$ such that $d^{1+\delta} \leqslant q-1$, $G_\pi(g)$ can be computed in time $(d \log q)^{1+\delta} + \tilde{O}(d \log q \log \pi)$.

Based on:

Theorem [Kedlaya-Umans'11]

Let $f, g, h \in \mathbb{F}_q[X]$ of degree d. For all $\delta > 0$, $(f \circ g \mod h)$ can be computed in time $d^{1+\delta}\tilde{O}(\log q)$.

Corollary

Let $g \in \mathbb{F}_q[X]$ and $q = \rho\pi_1 \cdots \pi_m + 1$. For all δ, $G_\rho(g)$, $G_{\rho\pi_1}(g)$, $\ldots$, $G_{\rho\pi_1 \cdots \pi_{m-1}}(g)$ can be computed in time $(d \log^2 q)^{1+\delta}$.

Let $q = \rho\pi_1 \cdots \pi_m + 1 = \rho\chi + 1$ and $g = G_\rho(f) = \prod_i (\alpha_i - X)$

$$\prod_{i=1}^{r} (\alpha_i - X) \xrightarrow{G_\pi} \prod_{i=1}^{r} (\alpha_i^\pi - X)$$

Let $q = \rho\pi_1 \cdots \pi_m + 1 = \rho\chi + 1$ and $g = G_\rho(f) = \prod_i (\alpha_i - X)$

$$\prod_{i=1}^{r} (\alpha_i - X) \xrightarrow{G_\pi} \prod_{i=1}^{r} (\alpha_i^\pi - X)$$

ξ: primitive root of unity of order χ

$$\prod_{i=1}^{r} (\xi^{e_i} - X) \xrightarrow{G_\pi} \prod_{i=1}^{r} (\xi^{f_i} - X)$$

Let $q = \rho\pi_1 \cdots \pi_m + 1 = \rho\chi + 1$ and $g = G_\rho(f) = \prod_i (\alpha_i - X)$

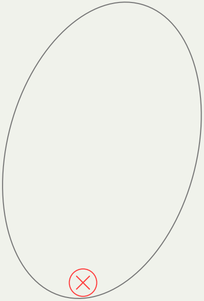
$$\prod_{i=1}^{r} (\alpha_i - X) \xrightarrow{G_\pi} \prod_{i=1}^{r} (\alpha_i^\pi - X)$$

ξ: primitive root of unity of order χ

$$\prod_{i=1}^{r} (\xi^{e_i} - X) \xrightarrow{G_\pi} \prod_{i=1}^{r} (\xi^{f_i} - X)$$

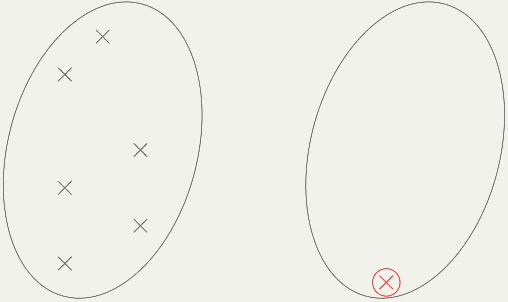
$$\begin{aligned} & \forall i, (\xi^{e_i})^\pi = \xi^{f_i} \\ \iff & \forall i, \pi e_i = f_i \mod \chi \\ \iff & \forall i, e_i \in \left\{ \frac{f_i + j\chi}{\pi} : 0 \leqslant j \leqslant \pi - 1 \right\} \end{aligned}$$

A deterministic algorithm

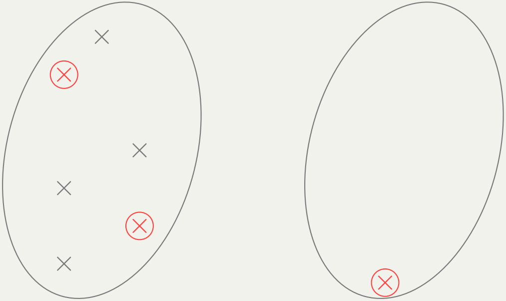


$Z_m = \{\xi^0\}$

A deterministic algorithm



$\{\xi^{\frac{j\chi}{\pi_m}} : 0 \leqslant j \leqslant \pi_m\}$ $\quad$ $Z_m = \{\xi^0\}$

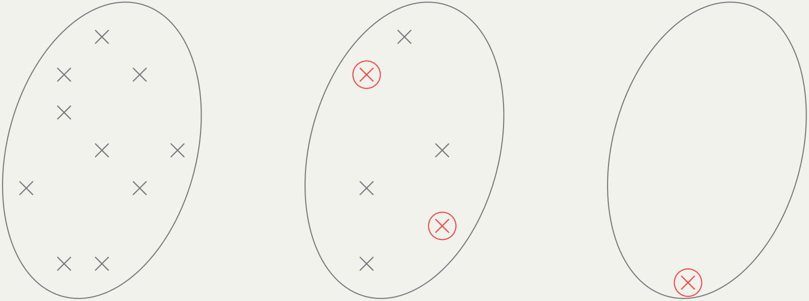


$\{\xi^{\frac{j\chi}{\pi_m}} : 0 \leqslant j \leqslant \pi_m\}$

$h_{m-1}(\xi^e) = 0$

$Z_m = \{\xi^0\}$

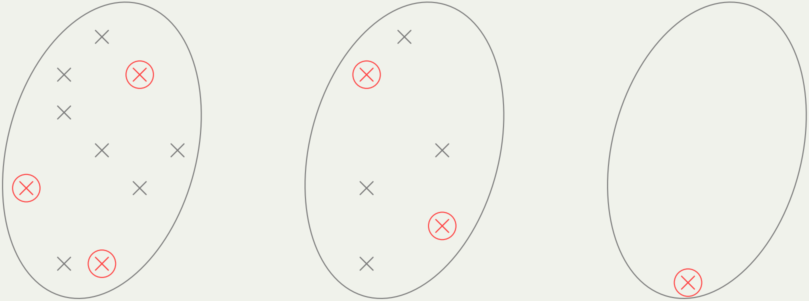
A deterministic algorithm



$\{\xi^{\frac{e+j\chi}{\pi_{m-1}}} : 0 \leqslant j \leqslant \pi_{m-1}\}$ $\{\xi^{\frac{j\chi}{\pi_m}} : 0 \leqslant j \leqslant \pi_m\}$ $Z_m = \{\xi^0\}$

$h_{m-1}(\xi^e) = 0$

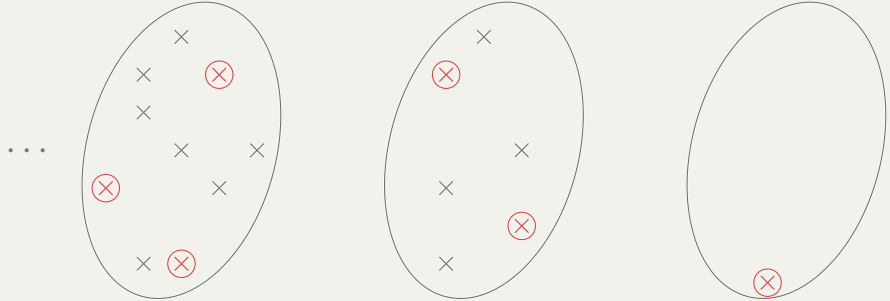
A deterministic algorithm



$\{\xi^{\frac{e+j\chi}{\pi_{m-1}}} : 0 \leqslant j \leqslant \pi_{m-1}\}$ $\{\xi^{\frac{j\chi}{\pi_m}} : 0 \leqslant j \leqslant \pi_m\}$ $Z_m = \{\xi^0\}$

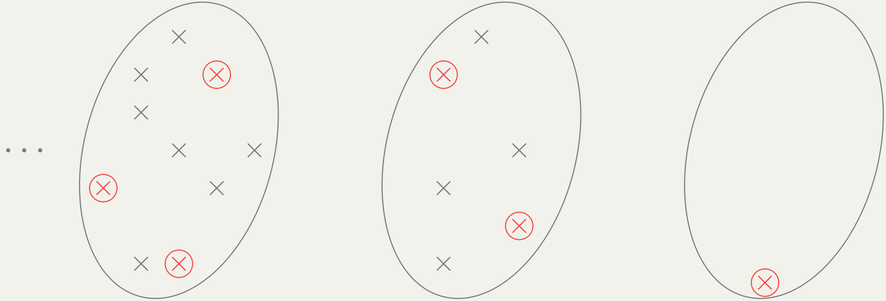
$h_{m-2}(\xi^e) = 0$ $h_{m-1}(\xi^e) = 0$

A deterministic algorithm



$\{\xi^{\frac{e+j\chi}{\pi_{m-1}}} : 0 \leqslant j \leqslant \pi_{m-1}\}$ $\{\xi^{\frac{j\chi}{\pi_m}} : 0 \leqslant j \leqslant \pi_m\}$ $Z_m = \{\xi^0\}$

$h_{m-2}(\xi^e) = 0$ $h_{m-1}(\xi^e) = 0$



$\{\xi^{\frac{e+j\chi}{\pi_{m-1}}} : 0 \leqslant j \leqslant \pi_{m-1}\}$ $\quad$ $\{\xi^{\frac{j\chi}{\pi_m}} : 0 \leqslant j \leqslant \pi_m\}$ $\quad$ $Z_m = \{\xi^0\}$

$h_{m-2}(\xi^e) = 0$ $\quad$ $h_{m-1}(\xi^e) = 0$

Theorem

If $\rho, \max_i \pi_i = O(\log q)$, the algorithm runs in time $\tilde{O}(d \log^3 q)$.

Lemma

Given $h = G_\pi(g)$, and $\{a_1, \ldots, a_l\}$ its roots, one can compute the roots of g in time $\tilde{O}(\sqrt{\pi} d \log q) + (d \log q)^{1+\delta}$ for all $\delta > 0$.

Lemma

Given $h = G_\pi(g)$, and $\{a_1, \dots, a_l\}$ its roots, one can compute the roots of g in time $\tilde{O}(\sqrt{\pi} d \log q) + (d \log q)^{1+\delta}$ for all $\delta > 0$.

Theorem

Given $f \in \mathbb{F}_q[X]$ with $\deg(f)$ distinct roots in $\mathbb{F}_q^*$ and a primitive element of $\mathbb{F}_q^*$, the roots of f can be computed in time

$$\tilde{O}(\sqrt{S_1(q-1)} d \log^2 q) + (d \log^2 q)^{1+\delta}$$

where $S_1(q-1)$ is the largest factor of $q-1$.

Lemma

Given $h = G_\pi(g)$, and $\{a_1, \dots, a_l\}$ its roots, one can compute the roots of g in time $\tilde{O}(\sqrt{\pi} d \log q) + (d \log q)^{1+\delta}$ for all $\delta > 0$.

Theorem

Given $f \in \mathbb{F}_q[X]$ with $\deg(f)$ distinct roots in $\mathbb{F}_q^*$ and a primitive element of $\mathbb{F}_q^*$, the roots of f can be computed in time

$$\tilde{O}(\sqrt{S_1(q-1)} d \log^2 q) + (d \log^2 q)^{1+\delta}$$

where $S_1(q-1)$ is the largest factor of $q-1$.

- Best known bound for *smooth* q;
- If $q = M2^m + 1$, $M = O(\log q)$, complexity $\tilde{O}(d \log^2 q)$.

Definition

The **tangent Graeffe transform of order π** of $g \in \mathbb{F}_q[X]$ is

$$G_\pi(g(X+\varepsilon)) \in (\mathbb{F}_q[\varepsilon]/\langle\varepsilon^2\rangle)[X].$$

Remark. $G_\pi(g(X+\varepsilon)) = h(X) + \varepsilon\overline{h}(X)$ where $h = G_\pi(g)$.

Definition

The **tangent Graeffe transform of order π** of $g \in \mathbb{F}_q[X]$ is

$$G_\pi(g(X+\varepsilon)) \in (\mathbb{F}_q[\varepsilon]/\langle\varepsilon^2\rangle)[X].$$

Remark. $G_\pi(g(X+\varepsilon)) = h(X) + \varepsilon\overline{h}(X)$ where $h = G_\pi(g)$.

Lemma

A nonzero root β of h is a **simple root** iff $\overline{h}(\beta) \neq 0$. The corresponding root of g is $\alpha = \pi\beta h'(\beta)/\overline{h}(\beta)$.

Proof. $\overline{h}(\alpha^\pi) = \pi\alpha^{\pi-1}h'(\alpha^\pi)$.

Goal: Ensure many **simple roots**.

Goal: Ensure many **simple roots**.

- Replace f by $f_\tau(X) = f(X - \tau)$ for a random $\tau \in \mathbb{F}_q$.

Goal: Ensure many **simple roots**.

- Replace f by $f_\tau(X) = f(X-\tau)$ for a random $\tau \in \mathbb{F}_q$.

Lemma

If $q = \rho\chi + 1$ with $\chi \geqslant d(d-1)$,

$$\mathbb{P}_{\tau\in\mathbb{F}_q}[G_\rho(f_\tau) \text{ has multiple roots}] \leqslant \frac{1}{2}.$$

Goal: Ensure many **simple roots**.

- Replace f by $f_\tau(X) = f(X-\tau)$ for a random $\tau \in \mathbb{F}_q$.

Lemma

If $q = \rho\chi + 1$ with $\chi \geqslant d(d-1)$,

$$\mathbb{P}_{\tau\in\mathbb{F}_q}[G_\rho(f_\tau) \text{ has multiple roots}] \leqslant \frac{1}{2}.$$

Proof. Given $\alpha_i \neq \alpha_j$,

$$\#\left\{\tau \in \mathbb{F}_q : (\tau+\alpha_i)^\rho = (\tau+\alpha_j)^\rho\right\} \leqslant \rho.$$

$\Longrightarrow$ $G_\rho(f_\tau)$ has multiple roots for at most $\frac{d(d-1)}{2}\rho$ values of τ.

A randomized algorithm

- $q = M \cdot 2^m + 1$

A randomized algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant d(d-1)$

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant d(d-1)$

$$f(X-\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_l + \varepsilon\overline{h}_l \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_m + \varepsilon\overline{h}_m$$

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant d(d-1)$

$$f(X-\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_l + \varepsilon\overline{h}_l \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_m + \varepsilon\overline{h}_m$$

$$h_m + \varepsilon\overline{h}_m \longmapsto Z_m \subset \{\xi^e : 0 \leqslant e < M\}$$

A randomized algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant d(d-1)$

$$
\begin{array}{ccccccccc}
f(X - \tau + \varepsilon) & \xrightarrow{G_2} & \cdots & \xrightarrow{G_2} & h_l + \varepsilon\overline{h}_l & \xrightarrow{G_2} & \cdots & \xrightarrow{G_2} & h_m + \varepsilon\overline{h}_m \\
 & & & & & & & & \downarrow \\
 & & & & Z_l & \longleftarrow & \cdots & \longleftarrow & Z_m \\
 & & & & & & & & \cap \\
 & & & & & & & & \{\xi^e : 0 \leqslant e < M\}
\end{array}
$$

A randomized algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant d(d-1)$

$$f(X-\tau+\varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_l + \varepsilon \overline{h}_l \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_m + \varepsilon \overline{h}_m$$

$$\downarrow$$

$$Z_0 \xleftarrow{\cdots} Z_l \longleftarrow \cdots \longleftarrow Z_m$$

Only simple roots

$$Z_m \subset \{\xi^e : 0 \leqslant e < M\}$$

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant d(d-1)$

$$f(X - \tau + \varepsilon) \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_l + \varepsilon\overline{h}_l \xrightarrow{G_2} \cdots \xrightarrow{G_2} h_m + \varepsilon\overline{h}_m$$

$$h_m + \varepsilon\overline{h}_m \longmapsto Z_m$$

$$Z_0 \longleftarrow \cdots \longleftarrow Z_l \longleftarrow \cdots \longleftarrow Z_m$$

Only simple roots

$$Z_m \subset \{\xi^e : 0 \leqslant e < M\}$$

- Recursive call with $f / \prod_{\alpha \in Z_0}(X - \alpha)$.

Theorem

If $q = M2^m + 1$ with $M = O(\log q)$, the randomized algorithm runs in expected time $\tilde{O}(d \log^2 q)$.

Theorem

If $q = M2^m + 1$ with $M = O(\log q)$, the randomized algorithm runs in expected time $\tilde{O}(d \log^2 q)$.

- Same asymptotic as Cantor-Zassenhaus' algorithm;
- Better efficiency in practice.

Theorem

If $q = M2^m + 1$ with $M = O(\log q)$, the randomized algorithm runs in expected time $\tilde{O}(d \log^2 q)$.

- Same asymptotic as Cantor-Zassenhaus' algorithm;
- Better efficiency in practice.

Heuristic

Let $q = \rho\chi + 1$ and $f \in \mathbb{F}_q[X]$ with $d = \deg(f)$ roots in $\mathbb{F}_q^*$. If $\chi \geqslant 4d$, $G_\rho(f(X+\tau))$ has $\geqslant d/3$ simple roots with probability at least $1/2$, for a random $\tau \in \mathbb{F}_q$.

Justification: holds for a random f rather than $f(X+\tau)$.

A heuristic algorithm

- $q = M \cdot 2^m + 1$

A heuristic algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant 4d$

A heuristic algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant 4d$

$$f(X - \tau + \varepsilon) \xrightarrow{\quad G_{2^l} \quad} h_l + \varepsilon \overline{h}_l$$

A heuristic algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant 4d$

$$f(X - \tau + \varepsilon) \xrightarrow{\quad G_{2^l} \quad} h_l + \varepsilon \overline{h}_l$$

$$h_l + \varepsilon \overline{h}_l \longmapsto Z_l$$

$$Z_l \subset \{\xi^e : 0 \leqslant e < M2^l\}$$

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant 4d$

$$
\begin{array}{ccc}
f(X - \tau + \varepsilon) & \xrightarrow{\quad G_{2^l} \quad} & h_l + \varepsilon \overline{h}_l \\
 & & \big\downarrow \\
Z_0 & \xleftarrow[\text{only simple roots}]{\qquad\qquad} & Z_l \\
 & & \cap \\
 & & \{\xi^e : 0 \leqslant e < M2^l\}
\end{array}
$$

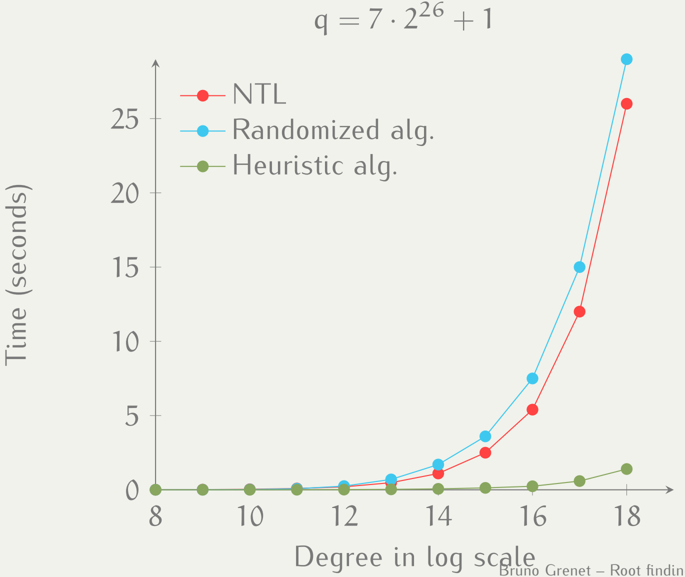
A heuristic algorithm

- $q = M \cdot 2^m + 1$
- Find the largest l s.t. $M2^{m-l} \geqslant 4d$

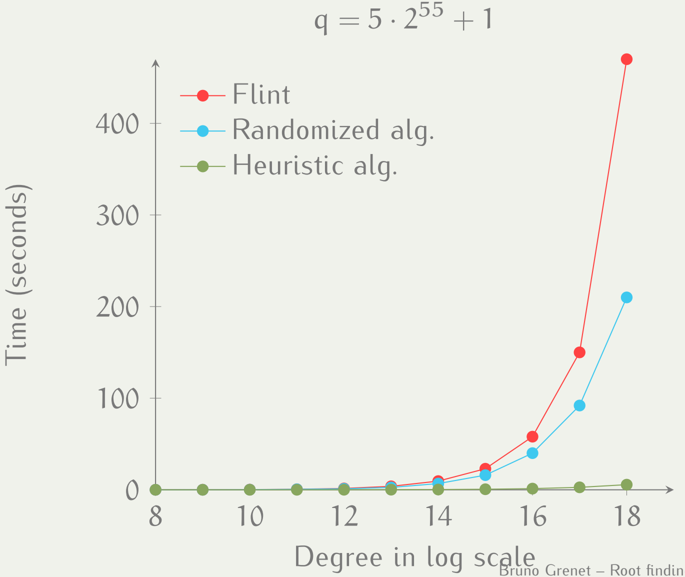
$$
\begin{array}{ccc}
f(X - \tau + \varepsilon) & \xrightarrow{\quad G_{2^l} \quad} & h_l + \varepsilon \overline{h}_l \\
 & & \downarrow \\
Z_0 & \xleftarrow[\text{only simple roots}]{} & Z_l \\
 & & \cap \\
 & & \{\xi^e : 0 \leqslant e < M2^l\}
\end{array}
$$

- Recursive call with $f / \prod_{\alpha \in Z_0} (X - \alpha)$.

- Algorithms implemented in Mathemagix (`http://mathemagix.org/`);
- Heuristic algorithm faster than flint and ntl by factors up to 80;
- Modification of Cantor-Zassenhaus algorithm: gain for large q only.



$q = 7 \cdot 2^{26} + 1$
NTL
Randomized alg.
Heuristic alg.
Time (seconds)
0
5
10
15
20
25
8
10
12
14
16
18
Degree in log scale



$q = 5 \cdot 2^{55} + 1$
Flint
Randomized alg.
Heuristic alg.
Time (seconds)
0
100
200
300
400
8
10
12
14
16
18
Degree in log scale

Conclusion

- Revisit classical algorithms for finite fields of smooth cardinality;

Conclusion

- Revisit classical algorithms for finite fields of smooth cardinality;
- New approach using Graeffe transforms:

Conclusion

- Revisit classical algorithms for finite fields of smooth cardinality;
- New approach using Graeffe transforms:
  - Good deterministic complexity bounds;
  - Good probabilistic complexity bounds;
  - Good computation times.

Conclusion

- Revisit classical algorithms for finite fields of smooth cardinality;
- New approach using Graeffe transforms:
  - Good deterministic complexity bounds;
  - Good probabilistic complexity bounds;
  - Good computation times.
- Open questions:
  - Deterministic alg.: use of tangent Graeffe transforms;
  - Heuristic alg.: Graeffe transform of order 2^l is the bottleneck;
  - Prove the heuristic!

- Revisit classical algorithms for finite fields of smooth cardinality;
- New approach using Graeffe transforms:
  - Good deterministic complexity bounds;
  - Good probabilistic complexity bounds;
  - Good computation times.
- Open questions:
  - Deterministic alg.: use of tangent Graeffe transforms;
  - Heuristic alg.: Graeffe transform of order 2^l is the bottleneck;
  - Prove the heuristic!

Thank you!