The Tangent-Graeffe root finding algorithm

Michael Monagan

Department of Mathematics,
Simon Fraser University

This is joint work with Joris van der Hoeven.

Let $f(x) \in \mathbb{F}_p[x]$ for p prime.
Suppose we know $f(x) = \prod_{i=1}^{d}(x - \alpha_i)$ with $\alpha_i \in \mathbb{F}_p$.

Problem 1: Compute the roots α_i of $f(x)$.
Using CZ (1981) – implemented in Maple by MBM and Magma by AS.
Using TG (2015) – requires $p = \sigma 2^k + 1$ with $\sigma \in O(d)$, e.g. $p = 5 \cdot 2^{55} + 1$.

Problem 2: Let $\beta_1, \beta_2, \ldots, \beta_d \in \mathbb{F}_p$.
Evaluate $f(\beta_i)$ for $1 \leq i \leq d$ (multi-point evalution).

Let $f(x) \in \mathbb{F}_p[x]$ for p prime.
Suppose we know $f(x) = \prod_{i=1}^{d}(x - \alpha_i)$ with $\alpha_i \in \mathbb{F}_p$.

Problem 1: Compute the roots α_i of $f(x)$.
Using CZ (1981) – implemented in Maple by MBM and Magma by AS.
Using TG (2015) – requires $p = \sigma 2^k + 1$ with $\sigma \in O(d)$, e.g. $p = 5 \cdot 2^{55} + 1$.

Problem 2: Let $\beta_1, \beta_2, \ldots, \beta_d \in \mathbb{F}_p$.
Evaluate $f(\beta_i)$ for $1 \leq i \leq d$ (multi-point evalution).

Evaluate	CZ	TG
$O(M(d) \log d)$	$O(M(d) \log d \log p)$	$O(M(d) \log p)$

Number of arithmetic operations in $\mathbb{F}_p$.

- CZ and TG are Las Vegas algorithms.
- TG is $O(\log d)$ times faster than CZ. Is TG really faster than CZ in practice?

Talk Outline

- What is a Las Vegas algorithm?
- The Graeffe transform
- The Tangent-Graeffe (TG) algorithm
- Improving the constant by a factor of 2
- Comparison of new C implementation with Magma's CZ implementation
- How big can the method go?
- Current work

What is a Las Vegas algorithm?

Input: 1: a problem instance X of size n from a set S
2: a sequence of k random bits where $k = f(n)$
3: a constant $0 < q < 1$

Output: a solution y with probability q or FAIL with probability $1 - q$

If $q = 0.5$, on average it will take 2 attempts to obtain a solution.

For $X = f(x) \in \mathbb{F}_p[x]$ k could depend on $\deg(f)$ and/or $\log p$.

The Graeffe Transform

Definition: Let $P(z) \in \mathbb{F}_p[z]$ of degree $d > 0$. The **Graeffe transform** of P is

$$\mathbf{G}(P) = P(z)P(-z)|_{z=\sqrt{z}} \in \mathbb{F}_p[z]$$

Lemma 1: If $P(z) = \prod_{i=1}^{d}(z - \alpha_i)$ then $\mathbf{G}(P) = \prod_{i=1}^{d}(z - \alpha_i^2)$.

Main idea: Let $p = \sigma 2^k + 1$. Pick $r = 2^N$ such that $s = (p-1)/r \in [2d, 4d)$.

1: Compute $\tilde{P} = \mathbf{G}^{(N)}(P)$. Then $\tilde{P} = \prod_{i=1}^{d}(z - \alpha_i^r)$.

Observe $s = (p-1)/r \implies p - 1 = rs \implies (\alpha_i^r)^s = 1$ by Fermat's theorem.

2: Pick ω with order s in $\mathbb{F}_p$. NB: $s \in O(d)$
Compute $\{\omega^i : \tilde{P}(\omega^i) = 0 \text{ for } 0 \le i < s\} = \{\alpha_i^r : 1 \le i \le d\}$ using multi-point evaluation.

Okay so how to we get α_i from α_i^r ?

The Tangent Graeffe transform.

Lemma 2: Let $\tilde{P}(z) = P(z+\epsilon) \mod \epsilon^2 \in \mathbb{F}_p[\epsilon, z]/(\epsilon^2)$. Then

1. $\tilde{P}(z) = P(z) + P'(z)\epsilon$
2. $\mathbf{G}(\tilde{P}(z)) = \underbrace{P(z)P(-z)|_{z=\sqrt{z}} + (P(z)P'(-z) + P(-z)P'(z))|_{z=\sqrt{z}}\ \epsilon}_{\text{three polynomial multiplications}}$
3. $\mathbf{G}^{(N)}(\tilde{P}(z)) = A(z) + B(z)\epsilon$ where $A(z) = \mathbf{G}^{(N)}(P)$

Lemma 3: If $A(\beta) = 0$ and $A'(\beta) \neq 0$ then $\alpha = \dfrac{r\beta A'(\beta)}{B(\beta)}$ is a root of $P(z)$.

Compute $\mathbf{G}^{(N)}(P(z+\epsilon)) = A(z) + B(z)\epsilon$ with $3N$ multiplications
Compute $A(\omega^i), A'(\omega^i), B(\omega^i)$ for $0 \leq i < s$ and apply Lemma 3.

What's going on with the roots under G^N ?

Recap: $A(z) = G^N(P) = \prod_{i=1}^{d}(z - \alpha_i^r)$ where $r = 2^N$.
How many of the roots α_i^r are single roots of $G^N(P)$?

Example: Let $p = 41$ and $\alpha = [7, 10, 20, 21, 30, 35]$ so $d = 6$

What happens when we square these roots $N = 1, 2, 3$ times?

N	$G^{(N)}(\alpha)$	s		$e^{-d/s}$
1	[8, 18, 31, 31, 39, 36]	20	$2d \le s < 4d$	0.741
2	[23, 37, 18, 18, 4, 25]	10	$d \le s < 2d$	0.549
3	[37, 16, 37, 37, 16, 10]	5	$d/2 \le s < d$	0.301

Problem: if $\alpha = [1, -1, 2, -2, 3, -3]$ we get $G(\alpha) = [1, 1, 4, 4, 9, 9]$.

What's going on with the roots under G^N ?

Recap: $A(z) = G^N(P) = \prod_{i=1}^{d}(z - \alpha_i^r)$ where $r = 2^N$.
How many of the roots α_i^r are single roots of $G^N(P)$?

Example: Let $p = 41$ and $\alpha = [7, 10, 20, 21, 30, 35]$ so $d = 6$
What happens when we square these roots $N = 1, 2, 3$ times?

N	$G^{(N)}(\alpha)$	s		$e^{-d/s}$
1	[8, 18, 31, 31, 39, 36]	20	$2d \le s < 4d$	0.741
2	[23, 37, 18, 18, 4, 25]	10	$d \le s < 2d$	0.549
3	[37, 16, 37, 37, 16, 10]	5	$d/2 \le s < d$	0.301

Problem: if $\alpha = [1, -1, 2, -2, 3, -3]$ we get $G(\alpha) = [1, 1, 4, 4, 9, 9]$.

Solution: Pick $\tau \in \mathbb{F}_p$ at random and set $P = P(z + \tau)$.

The Tangent Graeffe Algorithm

Input: $P \in \mathbb{F}_p[z]$ of degree d with d distinct roots in $\mathbb{F}_p$ and $p = \sigma 2^k + 1$ with $2^k > 4d$.
Output: the set $\{\alpha_1, \ldots, \alpha_d\}$ of roots of P.

1. If $d = 0$ then return ϕ.
2. Let $s \in [2d, 4d)$ such that $s|(p-1)$ and set $r := (p-1)/s = 2^N$.
3. Pick $\tau \in \mathbb{F}_p$ at random and compute $P^* := P(z+\tau) \in \mathbb{F}_p[z]$ . . . . $O(M(d))$.
4. Compute $\tilde{P} := P^*(z) + P^*(z)'\epsilon$. // $= P^*(z+\epsilon) \mod \epsilon^2$.
5. For $i = 1, \ldots, N$ set $\tilde{P} := \mathbf{G}(\tilde{P})(z) \mod \epsilon^2$ . . . . $3NM(d)$.
6. Let ω have order s in $\mathbb{F}_p$. Let $\tilde{P}(z) = A(z) + B(z)\epsilon$.
   Evaluate $A(\omega^i), A'(\omega^i)$ and $B(\omega^i)$ for $0 \le i < s$ using Bluestein . . . . $3O(M(s))$.
7. If $P(\tau) = 0$ then set $S := \{\tau\}$ else set $S := \phi$.
8. For $\beta \in \{1, \omega, \ldots, \omega^{(s-1)}\}$
   if $A(\beta) = 0$ and $A'(\beta) \neq 0$ set $S := S \cup \{r\beta A'(\beta)/B(\beta) + \tau\}$.
9. Compute $Q := \prod_{\alpha \in S}(z - \alpha)$ and set $R = P/Q$ . . . . $O(M(d)\log d)$.
10. Recursively determine the set of roots S' of R and return $S \cup S'$.

For $s \in [2d, 4d)$, on average, we get at least $e^{-1/2} = 61\%$ of the roots.
Total cost $O(NM(d) + M(d)\log d + M(s)) = O(M(d)\log(p/s) + M(d)\log d)$.

Improving the constant in $\mathbf{G}(P)$ and $\mathbf{G}^{(N)}(P)$

$$\mathbf{G}(P) = P(z)P(-z)|_{z=\sqrt{z}} \text{ and } d = \deg P$$

Theorem

We can compute $\mathbf{G}(P)$ in $F(2d) + F(d) = 1/2M(d)$. Note: $M(d) = 3F(2d) + O(d)$.
We can compute $\mathbf{G}^{(N)}(P)$ in $(2N+1)F(d) = (1/3N + 1/6)M(d)$.

This compares with $2/3M(d)$ and $2/3NM(d)$ in [GHL 2015].

In the FFT, if $\omega^n = 1$ and $n = 2^k$ then $\omega^{n/2+i} = -\omega^i$ so

$$\begin{aligned} FFT(P(z)) &= [P(1), P(\omega), P(\omega^2), \ldots, P(-1), P(-\omega), P(-\omega^2), \ldots] \\ FFT(P(-z)) &= [P(-1), P(-\omega), P(-\omega^2), \ldots, P(1), P(\omega), f(\omega^2), \ldots] \end{aligned}$$

Also $FFT(H := P(z)P(-z))$ is

$$[H(1), H(\omega), H(\omega^2), \ldots, H(1), H(\omega), H(\omega^2), \ldots]$$

We can compute the inverse FFT with an FFT of size d.
Cost of $\mathbf{G}(P)$: $F(2d) + 0 + F^{-1}(d) < 1.5F(2d) < 1/2M(d)$.

Tangent-Graeffe v. Cantor-Zassenhaus

We implemented TG in C using the FFT for $\mathbf{G}(P)$ and for arithmetic in $\mathbb{F}_p[z]$.

Table: Sequential timings in CPU seconds for $p = 3 \cdot 29 \cdot 2^{56} + 1$ and using $s \in [2d, 4d)$. Intel Xeon E5 2660 CPU, 8 cores, 2.2 GHz base, 3.0 GHz turbo, 64 gigabytes RAM

	Our sequential TG implementation in C						Magma CZ timings	
d	total	first	%roots	$\mathbf{G}^{(N)}$	step6	step9	V2.25-3	V2.25-5
$2^{12}-1$	0.11s	0.07s	69.8%	0.04s	0.02s	0.01s	23.22s	8.43
$2^{13}-1$	0.22s	0.14s	69.8%	0.09s	0.03s	0.01s	56.58s	18.94
$2^{14}-1$	0.48s	0.31s	68.8%	0.18s	0.07s	0.02s	140.76s	44.07
$2^{15}-1$	1.00s	0.64s	69.2%	0.38s	0.16s	0.04s	372.22s	103.5
$2^{16}-1$	2.11s	1.36s	68.9%	0.78s	0.35s	0.10s	1494.0s	234.2
$2^{17}-1$	4.40s	2.85s	69.2%	1.62s	0.74s	0.23s	6108.8s	534.5
$2^{18}-1$	9.16s	5.91s	69.2%	3.33s	1.53s	0.51s	NA	1219.
$2^{19}-1$	19.2s	12.4s	69.2%	6.86s	3.25s	1.13s	NA	2809.
$2^{20}-1$	39.7s	25.7s	69.2%	14.1s	6.77s	2.46s	NA	6428.

Conclusion: TG is a lot (100 times) faster than CZ.

How big can the method go?

Can we factor $P(z) = z^{10^9} + \dots$ in $\mathbb{F}_p[z]$ for $p = 5 \cdot 2^{55} + 1$?
Note: we need 8 gigabytes for the input and 8 gigabytes for the output.

How big can the method go?

Can we factor $P(z) = z^{10^9} + \dots$ in $\mathbb{F}_p[z]$ for $p = 5 \cdot 2^{55} + 1$?
Note: we need 8 gigabytes for the input and 8 gigabytes for the output.

Succeeded in June 2020: time = 3,715 secs, space = 121 GB
Used an Intel E5 2680 CPU with 10 cores and 128 GB RAM.
Parallel implemenation in Cilk C.

To evaluate $A(\omega^i), A'(\omega^i), B(\omega^i)$ for $0 \le i < s = 5 \cdot 2^{30}$
Space: $3s + 3n = 504GB$ with $n = 2^k > 2s$ for $M(s)$ using Bluestein.
Use $s \in [2d, 4d)$ instead of $s \in [4d, 8d)$.
For $s = 5 \cdot 2^{29}$, a DFT$(5 \cdot 2^{29})$ can be done using $5F(2^{29}) + 2^{29}F(5) + O(s)$.
Space: $3s + 1.2s = 84GB.$

Current work.

We are trying to determine the constants in the complexities assuming the FFT model in order to determine how much faster CZ is than TG.

Tangent-Graeffe cost for $s \in [\lambda d, 2\lambda d)$.

$\mathbf{G}^{(N)}(P)$	$Q := \prod_{\alpha \in S} (z - \alpha)$
$< \frac{1}{3} e^{1/\lambda} M(d) \log_2 \frac{p}{\lambda d} + \ldots$	$< \frac{1}{4} M(d) \log_2 d + \ldots$

Cantor-Zassenhaus cost

$h := (z + \alpha)^{(p-1)/2} \bmod P(z)$	$g := \gcd(h(z) - 1, P(z))$
$< \frac{7}{6} M(d) \log_2 \frac{p}{2d} \log_2 d + \ldots$	$< \frac{5}{12} M(d) \log_2^2 d + \ldots$

References

David G. Cantor and Hans Zassenhaus.
A new algorithm for factoring polynomials over finite fields.
Math. Comp. **36**(154): 587–592, 1981.

Bruno Grenet, Joris van der Hoeven and Gregoire Lecerf.
Randomized Root Finding over Finite FFT-fields using Tangent Graeffe Transforms.
In Proceedings of ISSAC 2015, pp. 197–204, ACM, 2015.

Joris van der Hoven and Michael Monagan.
Implementing the tangent Graeffe root finding algorithm.
In Proceedings of ICMS 2020, LNCS **12097**, 482–492, Springer, 2020.
Preprint available on the HAL achive.