Sparse polynomial interpolation and computing roots of polynomials over finite fields

Michael Monagan

Department of Mathematics, Simon Fraser University, British Columbia

Computing roots of polynomials over finite fields.

Input: a polynomial $f(x) \in \mathbb{F}_p[x]$ of degree d where $\mathbb{F}_p$ is a prime field.
In our application we know that $f(x)$ has d roots in $\mathbb{F}_p$.
Output: the d roots in $\mathbb{F}_p$.

```
> p := 5*2^25+1;
                                    p := 167772161

> Factor(x^4+1) mod p;

        (x + 137365239) (x + 96278553) (x + 30406922) (x + 71493608)

> Roots(x^4+1) mod p;

        [[137365239, 1], [71493608, 1], [96278553, 1], [30406922, 1]]
```

What is Maple doing?

Let $f(x) = \prod_{i=1}^{d}(x - \alpha_i)$, $\alpha_i \in \mathbb{F}_p$, $p \gg d$.

Problem 1: Compute the roots α_i of f.
Using CZ (1981) – implemented in Maple in 1992 by MBM.
Using TG (2015) – requires $p = \sigma 2^k + 1$ with $\sigma \leq 4d$, 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
$O(M(d) \log d)$	$O(M(d) \log d \log p)$	$O(M(d) \log p)$

TG is $O(\log d)$ times faster than CZ.
Is TG practical?

Is TG practical?

A comparison of our C code implementation of TG with FFT based algorithms with Maple and Magma. Maple and Magma both use CZ but Maple's $M(d) \in O(d^2)$ and Magma's $M(d) \in O(d \log d \log\log d)$

d	Maple	Magma	TG in C
100	0.15s	1.16s	0.00s
10^3	1.95s	2.31s	0.01s
10^4	51.86s	18.36s	1.71s
10^5	2927.4s	298.68s	5.52s
10^6			18.80s
10^7			115.59s

Table: CPU timings in seconds for $p = 5 \times 2^{55} + 1$

How can we integrate the C code into Maple?

```
#define LONG long long int
int roots64s( LONG A[], LONG d, LONG R[], LONG p );

gcc -O3 -shared -o TG9.so -fPIC fftutil10.c  ...  polyalg3.c TG9.c

> TG := define_external('roots64s',
>   LL::ARRAY(0..nn,integer[8]), nn::integer[4],
>   RR::ARRAY(1..nn,integer[8]), pp::integer[8], LIB="TG9.so",
>   RETURN::integer[4]):
> p := 5*2^25+1;
                              p := 167772161

> F := Array(0..4,[1,0,0,0,1],datatype=integer[8]):
> R := Array(1..4,datatype=integer[8]): # for output roots
> TG(F,4,R,p);
                                     4
> R;
                  [137365239, 96278553, 30406922, 71493608]
```

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 $\mathbb{F}_p$.
Compute $\{\omega^i : \tilde{P}(\omega^i) = 0 \leq i < s\} = \{\alpha_i^r : 1 \leq i \leq d\}$ 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. $\mathbf{G}(\tilde{P}(z)) = \underbrace{P(z)P(-z) + (P(z)P'(-z) + P(-z)P'(z))\,\epsilon}_{\text{three polynomial multiplications}}$
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 $O(NM(d) + M(d)\log d + M(s)) = O(M(d)\log(p/s) + M(d)\log d)$.

How big a polynomial can TG factor?

Can we factor $P(z) = z^{10^9} + \ldots$ 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.

Yes! time = 4000s, space = 121 GB
Used an Intel E5 2680 CPU with 10 cores and 128 GB RAM.

See [4] for details.

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: $3s + 1.2s = 84GB$.

References

J. von zur Gathen and J. Gerhard.
Modern Computer Algebra.
3rd ed., Cambridge University Press, 2013.

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.

Joris van der Hoven and Michael Monagan.
Computing one billion roots using the tangent Graeffe method.
Submitted July 2020 to *Communications in Computer Algebra*.
HAL archive: https://hal.archives-ouvertes.fr/hal-02525408/