Polynomial Interpolation And Identity testing

Gorav Jindal

Saarland University

PhD Application Talk, 7 October 2013

Outline

Outline

1. Introduction
   - Polynomial Interpolation
   - Previous work
   - New Black-box model
2. Uni-variate interpolation
   - Preliminaries
   - Good primes
   - Goodg primes
   - Final algorithm
3. Applications and Other work
4. Questions?

Polynomial Interpolation

- Black-box Model
  - $\mathcal{R}$ is the underlying ring
  - $P(x_1, x_2, \ldots, x_n) \in \mathcal{R}[x_1, x_2, \ldots, x_n]$,

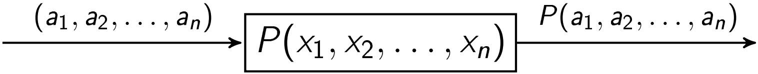

Figure: Traditional Black-Box Model

- Ask value of P at some set of points and output P as a list of coefficients along with corresponding monomials

Outline

1. Introduction
   - Polynomial Interpolation
   - Previous work
   - New Black-box model
2. Uni-variate interpolation
   - Preliminaries
   - Good primes
   - Goodg primes
   - Final algorithm
3. Applications and Other work
4. Questions?

Previous work

- Lot of previous research in Black-box polynomial interpolation.
  - Randomized algorithm by Zippel [Zip79].
  - Technique for deterministic algorithm by Grigoriev and Karpinski [GK87] .
  - Deterministic algorithm by Ben-Or and Tiwari [BO88], using the technique of [GK87].
    - Makes $2m$ queries to the given black box.

Over finite fields

- Studied extensively in [Wer94, GKS90, CDGK91].
- NC algorithm for interpolating m-sparse polynomials over finite fields [GKS90].
  - $O(\log^3(nm))$ Boolean parallel time.
  - $O(n^2 m^6 \log^2(nm))$ processors.
- Polynomial interpolation over fields of large characteristic by Klivans and Spielman [KS01].
- Interpolation over integers.
  - Known algorithms take time polynomial in d.

Outline

1. Introduction
   - Polynomial Interpolation
   - Previous work
   - New Black-box model
2. Uni-variate interpolation
   - Preliminaries
   - Good primes
   - Goodg primes
   - Final algorithm
3. Applications and Other work
4. Questions?

Our New Black-Box Model

- Works over Integers.
- Uses access to the black-box in a new way.

Figure: Our Black-Box Model

Our New Black-Box Model

- Works over Integers.
- Uses access to the black-box in a new way.

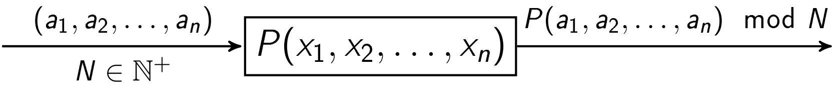

Figure: Our Black-Box Model

Why this model makes sense?

- No Extra information.
- May help in designing algorithms running in time sub-linear in d.
  - Traditional black-box model output will have $\Omega(d)$ bits.
- Generalized version of arithmetic circuits over integers.

Our contribution

Theorem

In the new black-box model, there is an algorithm to interpolate m-sparse polynomials in time $\text{poly}(m, n, \log d, \log H)$.

- First algorithm with sub-linear dependence on degree d.
- Running time is polynomial in output size.
  - Output size $= m \cdot (n \log d + \log H)$.

Outline

Interpolating Modulo prime p

- $F(x) = \sum_{i=1}^{m} c_i x^{\alpha_i}$, $|c_i| \leq H$ and $\alpha_i \leq d$.
- Interpolating $F(x)$ modulo prime p.
  - Ask value of $F(x)$ mod p at $\{0, 1, 2, \ldots, p-1\}$
  - Interpolate to obtain $F_p(x) = \sum_{i=1}^{m} (c_i \bmod p) x^{\alpha_i \bmod (p-1)}$
- Want all coefficients, but choice of p may bad.
  - If some coefficient vanishes modulo p
  - Or $\alpha_i \equiv \alpha_j \bmod (p-1)$.

Interpolating Modulo prime p

- $F(x) = \sum_{i=1}^{m} c_i x^{\alpha_i}$, $|c_i| \leq H$ and $\alpha_i \leq d$.
- Interpolating $F(x)$ modulo prime p.
  - Ask value of $F(x)$ mod p at $\{0, 1, 2, \ldots, p-1\}$
  - Interpolate to obtain $F_p(x) = \sum_{i=1}^{m} (c_i \bmod p) x^{\alpha_i \bmod (p-1)}$
- Want all coefficients, but choice of p may bad.
  - If some coefficient vanishes modulo p
  - Or $\alpha_i \equiv \alpha_j \bmod (p-1)$.

How to avoid bad primes?

- Avoiding primes modulo which some coefficient vanishes
  - A number cannot have too many distinct prime divisors
  - At most $m \log H$ bad primes
- Avoiding primes modulo which two monomials merge
  - p is bad when $(p-1) \mid (\alpha_i - \alpha_j)$
  - Difficult to bound the number of primes p such that $(p-1)$ divides an integer

How to avoid bad primes?

- Avoiding primes modulo which some coefficient vanishes
  - A number cannot have too many distinct prime divisors
  - At most $m \log H$ bad primes
- Avoiding primes modulo which two monomials merge
  - p is bad when $(p-1) \mid (\alpha_i - \alpha_j)$
  - Difficult to bound the number of primes p such that $(p-1)$ divides an integer

Primes in AP

- For $k \in \mathbb{N}^+$
  - Consider primes in AP $1+k, 1+2k, 1+3k, \ldots$
  - $P(k) =$ Smallest prime in above AP

Theorem (Linnik's Theorem)

$\exists k_0, L \in \mathbb{N}^+$ *such that* $\forall k \geq k_0 : P(k) \leq k^L$

- Interpolate modulo $p = P(q)$ for prime q
  - Makes sure that $p - 1$ cannot divide an integer for too many such p

Primes in AP

- For $k \in \mathbb{N}^+$
  - Consider primes in AP $1 + k, 1 + 2k, 1 + 3k, \ldots$
  - $P(k)$ =Smallest prime in above AP

Theorem (Linnik's Theorem)

$\exists k_0, L \in \mathbb{N}^+$ *such that* $\forall k \geq k_0 : P(k) \leq k^L$

- Interpolate modulo $p = P(q)$ for prime q
  - Makes sure that $p - 1$ cannot divide an integer for too many such p

Interpolating Modulo $P(q)$

- Want $P(q_1) \neq P(q_2)$ for distinct primes q_1 and q_2.
  - May not be always true.
  - But $P(q)$ can not be same for too many distinct primes q.

Lemma (Lemma 2 in [BHLV09])

Let $k_0 < q_1 < q_2 < \ldots < q_v$ and $\forall i \in [v] : P(q_i) = p$. Then $v \leq 5$.

Interpolating Modulo $P(q)$

- Want $P(q_1) \neq P(q_2)$ for distinct primes q_1 and q_2.
  - May not be always true.
  - But $P(q)$ can not be same for too many distinct primes q.

Lemma (Lemma 2 in [BHLV09])

Let $k_0 < q_1 < q_2 < \ldots < q_v$ and $\forall i \in [v] : P(q_i) = p$. Then $v \leq 5$.

Outline

1. Introduction
   - Polynomial Interpolation
   - Previous work
   - New Black-box model
2. Uni-variate interpolation
   - Preliminaries
   - Good primes
   - Goodg primes
   - Final algorithm
3. Applications and Other work
4. Questions?

Interpolating Modulo $P(q)$

Definition (Bad number (or prime))

A number (or prime) q is Bad for a polynomial $F(x)$ if $P(q) \mid c_i$ or $(P(q) - 1) \mid (\alpha_j - \alpha_k)$.

- How many Bad primes?
  - At most $5m \log H$ bad for coefficients
  - At most $\binom{m}{2} \log d$ bad for monomial merging
- At most $b = \binom{m}{2} \log d + 5m \log H$ Bad primes.

Definition (Good number (or prime))

A number (or prime) q is called Good for a polynomial $F(x)$ if it is not Bad.

Interpolating Modulo $P(q)$

Definition (Bad number (or prime))

A number (or prime) q is Bad for a polynomial $F(x)$ if $P(q) \mid c_i$ or $(P(q)-1) \mid (\alpha_j - \alpha_k)$.

- How many Bad primes?
  - At most $5m \log H$ bad for coefficients
  - At most $\binom{m}{2} \log d$ bad for monomial merging
- At most $b = \binom{m}{2} \log d + 5m \log H$ Bad primes.

Definition (Good number (or prime))

A number (or prime) q is called Good for a polynomial $F(x)$ if it is not Bad.

Interpolating Modulo $P(q)$

Definition (Bad number (or prime))

A number (or prime) q is Bad for a polynomial $F(x)$ if $P(q) \mid c_i$ or $(P(q) - 1) \mid (\alpha_j - \alpha_k)$.

- How many Bad primes?
  - At most $5m \log H$ bad for coefficients
  - At most $\binom{m}{2} \log d$ bad for monomial merging
- At most $b = \binom{m}{2} \log d + 5m \log H$ Bad primes.

Definition (Good number (or prime))

A number (or prime) q is called Good for a polynomial $F(x)$ if it is not Bad.

Finding a Good prime

- Interpolating modulo $P(q')$ for Bad prime q'.
  - We get less than m monomials in $F_{P(q')}(x)$.
- Interpolating modulo $P(q)$ for Good prime q.
  - We get exactly m monomials in $F_{P(q)}(x)$.
- Interpolate modulo $b+1$ primes $P(p_1), P(p_2), \ldots, P(p_{b+1})$ for distinct primes $p_1 < p_2 < \ldots < p_{b+1}$.
  - p_i is Good if $F_{P(p_i)}(x)$ has maximum number of monomials.

Finding a Good prime

- Interpolating modulo $P(q')$ for Bad prime q'.
  - We get less than m monomials in $F_{P(q')}(x)$.
- Interpolating modulo $P(q)$ for Good prime q.
  - We get exactly m monomials in $F_{P(q)}(x)$.
- Interpolate modulo $b+1$ primes $P(p_1), P(p_2), \ldots, P(p_{b+1})$ for distinct primes $p_1 < p_2 < \ldots < p_{b+1}$.
  - p_i is Good if $F_{P(p_i)}(x)$ has maximum number of monomials.

Finding many Good primes

- $t = \max\{\lceil \log H \rceil + 1, \lceil \log d \rceil\}$
- Enough to find t Good primes.
  - Above method can find t Good primes
- Use Chinese remaindering after interpolation modulo Good primes?
  - Order of coefficients/powers unknown.

Finding many Good primes

- $t = \max\{\lceil \log H \rceil + 1, \lceil \log d \rceil\}$
- Enough to find t Good primes.
  - Above method can find t Good primes
- Use Chinese remaindering after interpolation modulo Good primes?
  - Order of coefficients/powers unknown.

Outline

1. Introduction
   - Polynomial Interpolation
   - Previous work
   - New Black-box model
2. Uni-variate interpolation
   - Preliminaries
   - Good primes
   - Goodg primes
   - Final algorithm
3. Applications and Other work
4. Questions?

Finding many Goodg primes

- q_0 = Good prime already found
  - $g = P(q_0) - 1$

Definition (Badg prime)

Prime q is Badg for a polynomial $F(x)$ if gq is Bad number for $F(x)$.

Definition (Goodg prime)

A prime q is called Goodg for a polynomial $F(x)$ if it is not Badg.

Finding many Goodg primes

- $q_0 =$ Good prime already found
  - $g = P(q_0) - 1$

Definition (Badg prime)

Prime q is Badg for a polynomial $F(x)$ if gq is Bad number for $F(x)$.

Definition (Goodg prime)

A prime q is called Goodg for a polynomial $F(x)$ if it is not Badg.

Finding t Goodg primes

- At most $b = \binom{m}{2} \log d + 5m \log H$ Badg primes.
- Try $b + t$ primes
  - Pick t Goodg primes.
- Why Goodg primes are better than Good primes?
  - Can use Chinese remaindering.

Finding t Goodg primes

- At most $b = \binom{m}{2} \log d + 5m \log H$ Badg primes.
- Try $b + t$ primes
  - Pick t Goodg primes.
- Why Goodg primes are better than Good primes?
  - Can use Chinese remaindering.

Outline

Determining the order

- We have t Goodg primes $q_1, q_2, \ldots, q_t$.
  - Also $F_{P(gq_1)}(x), F_{P(gq_2)}(x), \ldots, F_{P(gq_t)}(x)$
- $F_{P(gq_i)}(x) = \sum_{j=1}^{m} c_{ij} x^{\alpha_{ij}}$

Lemma

$u \neq v \in [t]$, *and* $s = \gcd(P(gq_u) - 1, P(gq_v) - 1)$. *Then* $\forall j \in [m]$, *there exists a unique* $j' \in [m]$ *such that* $\alpha_{uj} \equiv \alpha_{vj'} \mod s$.

Determining the order

- We have t Goodg primes $q_1, q_2, \ldots, q_t$.
    - Also $F_{P(gq_1)}(x), F_{P(gq_2)}(x), \ldots, F_{P(gq_t)}(x)$
- $F_{P(gq_i)}(x) = \sum_{j=1}^{m} c_{ij} x^{\alpha_{ij}}$

Lemma

$u \neq v \in [t]$, and $s = \gcd(P(gq_u) - 1, P(gq_v) - 1)$. Then $\forall j \in [m]$, there exists a unique $j' \in [m]$ such that $\alpha_{uj} \equiv \alpha_{vj'} \bmod s$.

Completing the Interpolation

- For $j = 1$ to m
  - For $i = 2$ to t
    - $s_i = \gcd(P(gq_1) - 1, P(gq_i) - 1)$.
    - Find $k_{ij} \in [m]$ such that $\alpha_{ik_{ij}} \equiv \alpha_{1j} \bmod s_i$.
  - Compute α_j using CRT from $\alpha_{1j}, \alpha_{2k_{2j}}, \ldots, \alpha_{tk_{tj}}$.
  - Compute c_j using CRT from $c_{1j}, c_{2k_{2j}}, \ldots, c_{tk_{tj}}$.
- Runs in time $\text{poly}(m, \log d, \log H)$
  - Polynomial in output size.

Completing the Interpolation

- For $j = 1$ to m
  - For $i = 2$ to t
    - $s_i = \gcd(P(gq_1) - 1, P(gq_i) - 1)$.
    - Find $k_{ij} \in [m]$ such that $\alpha_{ik_{ij}} \equiv \alpha_{1j} \bmod s_i$.
  - Compute α_j using CRT from $\alpha_{1j}, \alpha_{2k_{2j}}, \ldots, \alpha_{tk_{tj}}$.
  - Compute c_j using CRT from $c_{1j}, c_{2k_{2j}}, \ldots, c_{tk_{tj}}$.
- Runs in time $\mathrm{poly}(m, \log d, \log H)$
  - Polynomial in output size.

Applications and Other work

- Easily adaptable to Multivariate interpolation.
- Can interpolate polynomials represented by arithmetic circuits.
- Other work
  - Polynomial Identity Testing
    - Faster deterministic algorithm.
    - Randomness efficient randomized algorithm.
    - Optimal randomness efficient randomized algorithm in a special case.

Questions?

Thank you

Questions?

Thank you

Chinese remaindering

Theorem (Generalized Chinese Remainder Theorem[BS96])

$m_1, m_2, \ldots, m_k$ be positive integers. Define $m = m_1 m_2 \ldots m_k$, and $m' = \operatorname{lcm}(m_1, m_2, \ldots, m_k)$. The system S of congruences

$$x \equiv x_i \bmod m_i, 1 \leq i \leq k$$

has a solution iff $x_i \equiv x_j \pmod{\gcd(m_i, m_j)}$ for all $i \neq j$. If the solution exists, it is unique $\pmod{m'}$.
We can determine if S has a solution, using $O((\log m)^2)$ bit operations, and if so, we can find the unique solution $\pmod{m'}$, using $O((\log m)^2)$ bit operations.

Multivariate interpolation

- Use the Kronecker substitution
  - Substitute $x_i \mapsto X^{(d+1)^{i-1}}$ to convert to uni-variate.
- Interpolate the uni-variate polynomial of degree at most $(d+1)^n - 1$
- Convert back to multivariate.
- Runs in time $\text{poly}(m, \log((d+1)^n - 1), \log H) = \text{poly}(m, n, \log d, \log H)$.

Polynomial identity testing

$$(a_1, a_2, \ldots, a_n) \in \mathbb{R}^n \longrightarrow \boxed{P(x_1, x_2, \ldots, x_n)} \longrightarrow P(a_1, a_2, \ldots, a_n) == 0$$

- $P(x_1, x_2, \ldots, x_n) = m$-sparse polynomial of unbounded degree over reals.
- Want to test whether $P(x_1, x_2, \ldots, x_n)$ is a zero polynomial.

Our contribution

- Improved deterministic algorithm running time from $\tilde{O}(m^3 n^3)$[BE11] to $\tilde{O}(m^2 n)$.
- Want randomized algorithm running in time poly$(n, \log m)$.
  - Lower bound of $\Omega(\log m)$ random bits known.
  - Upper bound of $O(\log^2 m)$ random bits known [BE11].
  - Improved upper bound to $O\left(\frac{\log^2 m}{\log \log m}\right)$ random bits.
- In case $P(x_1, x_2, \ldots, x_n)$ has degree bounded by poly(m) and coefficients are rationals.
  - Achieved upper bound of $O(\log m)$ random bits.

Our contribution

- Improved deterministic algorithm running time from $\tilde{O}(m^3 n^3)$[BE11] to $\tilde{O}(m^2 n)$.
- Want randomized algorithm running in time $\text{poly}(n, \log m)$.
  - Lower bound of $\Omega(\log m)$ random bits known.
  - Upper bound of $O(\log^2 m)$ random bits known [BE11].
  - Improved upper bound to $O\left(\frac{\log^2 m}{\log \log m}\right)$ random bits.
- In case $P(x_1, x_2, \ldots, x_n)$ has degree bounded by $\text{poly}(m)$ and coefficients are rationals.
  - Achieved upper bound of $O(\log m)$ random bits.

Our contribution

- Improved deterministic algorithm running time from $\tilde{O}(m^3 n^3)$[BE11] to $\tilde{O}(m^2 n)$.
- Want randomized algorithm running in time $\text{poly}(n, \log m)$.
  - Lower bound of $\Omega(\log m)$ random bits known.
  - Upper bound of $O(\log^2 m)$ random bits known [BE11].
  - Improved upper bound to $O\left(\frac{\log^2 m}{\log\log m}\right)$ random bits.
- In case $P(x_1, x_2, \ldots, x_n)$ has degree bounded by $\text{poly}(m)$ and coefficients are rationals.
  - Achieved upper bound of $O(\log m)$ random bits.

Markus Bläser and Christian Engels.
Randomness Efficient Testing of Sparse Black Box Identities of Unbounded Degree over the Reals.
In Thomas Schwentick and Christoph Dürr, editors, *28th International Symposium on Theoretical Aspects of Computer Science (STACS 2011)*, volume 9 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 555–566, Dagstuhl, Germany, 2011. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Markus Bläser, Moritz Hardt, Richard J. Lipton, and Nisheeth K. Vishnoi.
Deterministically testing sparse polynomial identities of unbounded degree.
Information Processing Letters, 109(3):187 – 192, 2009.

Michael Ben-Or.
A deterministic algorithm for sparse multivariate polynomial interpolation.
In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 301–309, New York, NY, USA, 1988. ACM.

E. Bach and J.O. Shallit.
Algorithmic Number Theory: Efficient Algorithms.
Number v. 1 in Algorithmic Number Theory. The Mit Press, 1996.

Michael Clausen, Andreas Dress, Johannes Grabmeier, and Marek Karpinski.
On zero-testing and interpolation of k-sparse multivariate polynomials over finite fields.

Theoretical Computer Science, 84(2):151 – 164, 1991.

Dima Grigoriev and Marek Karpinski.
The matching problem for bipartite graphs with polynomially bounded permanents is in NC (extended abstract).
In *FOCS*, pages 166–172, 1987.

Dima Yu. Grigoriev, Marek Karpinski, and Michael F. Singer.
Fast parallel algorithms for sparse multivariate polynomial interpolation over finite fields.
SIAM J. COMPUT, 19(6):1059–1063, 1990.

Adam R. Klivans and Daniel Spielman.
Randomness efficient identity testing of multivariate polynomials.

In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 216–223, New York, NY, USA, 2001. ACM.

Kai Werther.
The complexity of sparse polynomial interpolation over finite fields.
Applicable Algebra in Engineering, Communication and Computing, 5(2):91–103, 1994.

Richard Zippel.
Probabilistic algorithms for sparse polynomials.
In EdwardW. Ng, editor, *Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer Berlin Heidelberg, 1979.