Complexity of Shifted-Lacunary Polynomial Interpolation

Daniel S. Roche

Symbolic Computation Group
School of Computer Science
University of Waterloo

December 13, 2007

This is joint work
with Mark Giesbrecht.

First presented at the
MACIS conference
in Paris, France,
December 5–7, 2007.

Outline

Complexity of Shifted-Lacunary Polynomial Interpolation

General Problem

Determining a function from its values.

Goals

- Find the simplest possible formula.
- Don’t take too long.

Necessities

- What type of function? (output type)
- How big can it be? (output size)

Complexity of Shifted-Lacunary Polynomial Interpolation

Example

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

Suppose we can evaluate $f(x)$ at any chosen point.

- Can we find a formula for $f(x)$?

Complexity of Shifted-Lacunary Polynomial Interpolation

Example

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

Suppose we can evaluate $f(x)$ at any chosen point.

- Can we find a simple formula for $f(x)$?

Complexity of Shifted-Lacunary Polynomial Interpolation

Example

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

Suppose we can evaluate $f(x)$ at any chosen point.

- Can we find a simple formula for $f(x)$ in a reasonable amount of time?

Complexity of Shifted-Lacunary Polynomial Interpolation

Dense Methods

Definition (Dense Representation)

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n,$$

where $n = \deg(f)$ and $a_0, a_1, \ldots, a_n \in \mathsf{R}$

- Studied by Newton (1711), Waring (1779), ...
- Highly efficient implementations available

Complexity of Shifted-Lacunary Polynomial Interpolation

Dense Methods

Definition (Dense Representation)

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n,$$

where $n = \deg(f)$ and $a_0, a_1, \ldots, a_n \in \mathsf{R}$

Example

For $f(x) = (x-3)^{107} - 485(x-3)^{54}$, we will have

$$f(x) = x^{107} - 321x^{106} + 51039x^{105} - 5359095x^{104} + \cdots$$
$$+ 402009927496590798548375851523117926743035901448193 73x$$
$$- 1127130637840908780976768693419860197828458989848152$$

This is way too big! (twice exponential in the desired size)

Complexity of Shifted-Lacunary Polynomial Interpolation

Confusing Nomenclature

$$\text{Lacunary} \subseteq \text{Supersparse} \neq \text{Sparse}$$

- Default representation in Maple, Mathematica, etc.
- Some things are hard (Plaisted 1977, 1984)
- Some things aren't: Interpolation, finding low-degree factors
- Some things are unknown!

Complexity of Shifted-Lacunary Polynomial Interpolation

Sparse Methods

Definition (Lacunary Representation)

$$f(x) = b_1 x^{d_1} + b_2 x^{d_2} + \cdots + b_s x^{d_s},$$

where $d_1 < d_2 < \cdots < d_s = n$ and $b_1, \ldots, b_s \in \mathsf{R} \setminus \{0\}$

- Baron de Prony (1795), Ben-Or & Tiwari (1988), Kaltofen, Lakshman, Wiley, Lee, Lobo, ...
- Need to choose evaluation points
- R must have a high-order element and a fast logarithm.

Complexity of Shifted-Lacunary Polynomial Interpolation

Sparse Methods

Definition (Lacunary Representation)

$$f(x) = b_1 x^{d_1} + b_2 x^{d_2} + \cdots + b_s x^{d_s},$$

where $d_1 < d_2 < \cdots < d_s = n$ and $b_1, \ldots, b_s \in \mathsf{R} \setminus \{0\}$

Example

If $f(x) = (x-3)^{107} - 485(x-3)^{54}$,
this helps iff we know the sparsest shift 3,
since $f(x+3) = x^{107} - 485x^{54}$ is 2-sparse.

Complexity of Shifted-Lacunary Polynomial Interpolation

Definition (Shifted-Lacunary Representation)

$$f(x) = c_1(x-\alpha)^{e_1} + c_2(x-\alpha)^{e_2} + \cdots + c_t(x-\alpha)^{e_t},$$

where $e_1 < \cdots < e_t = n$ and t is minimal for any α

- This is our problem.
- Can be reduced to finding the sparsest shift α.
- We restrict the domain to $\mathbb{Q}[x]$.
- No previous polynomial-time algorithm known.

Complexity of Shifted-Lacunary Polynomial Interpolation

We give an algorithm with
output-sensitive polynomial-time complexity,
specifically, *bit complexity* polynomial in:

- Number of nonzero terms t
- Logarithm of the degree n
- Size of the coefficients $c_1, \ldots, c_t$
- Size of the sparsest shift α

Black box calls are assumed to have constant cost.

Computing the Sparsest Shift

- Borodin & Tiwari (1991)
  Compute sparsest shift from evaluation points (open)
- Grigoriev & Karpinski (1993)
  Compute sparsest shift from a black-box function.
  State need for complexity *not* polynomial in n
- Lakshman & Saunders (1996)
  Compute sparsest shift from dense representation
- Giesbrecht, Kaltofen, Lee (2003)
  Current best results (deterministic & probabilistic)

Uniqueness and Rationality of Sparsest Shift

Theorem (Lakshman & Saunders (1996))

If the degree is at least twice the sparsity,
then the sparsest shift is unique and rational.

Example

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

$\Rightarrow$ 3 is the only shift with ≤ 54 terms

Condition not satisfied means polynomial is dense.

Black Box Model

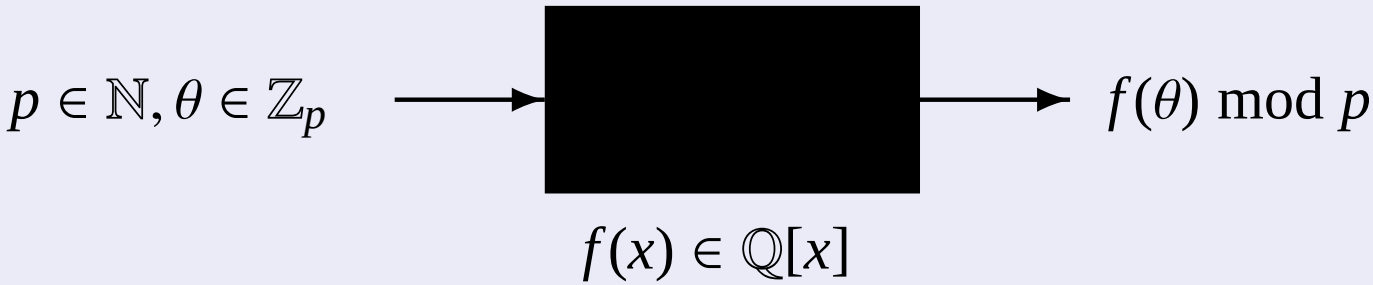
Arbitrary evaluations will usually be very large:

Example

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$
$$f(1) = -162259276829222100374855109050368$$

To control evaluation size, use modular arithmetic:

The “Modular Black-Box”



Modular-Reduced Polynomial

Definition

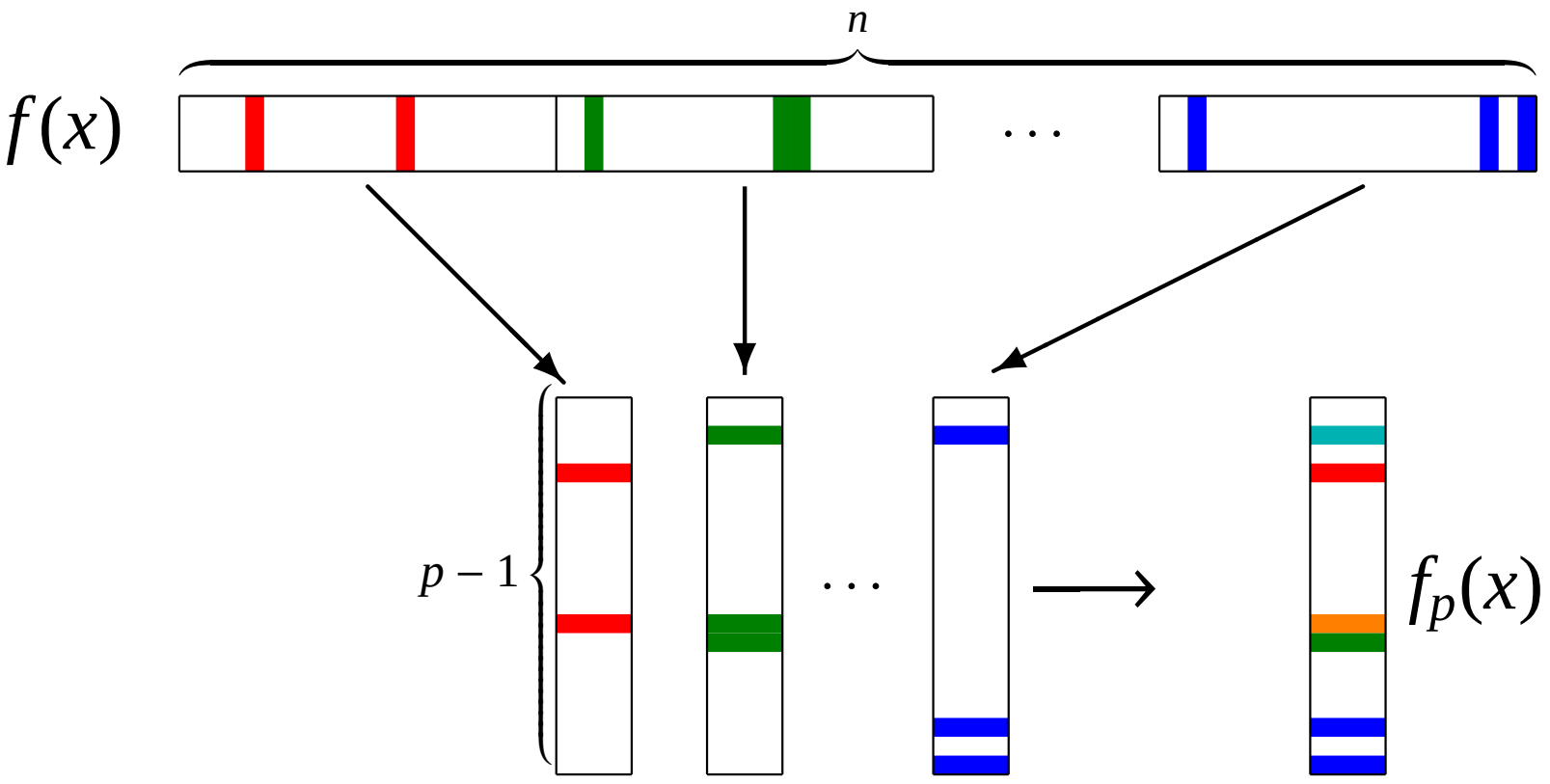
$$f(x) = c_1 (x-\alpha)^{e_1} + \cdots + c_t (x-\alpha)^{e_t}$$

$$f_p(x) = (c_1 \bmod p)(x-\alpha_p)^{e_1 \bmod (p-1)} + \cdots + (c_t \bmod p)(x-\alpha_p)^{e_t \bmod (p-1)},$$

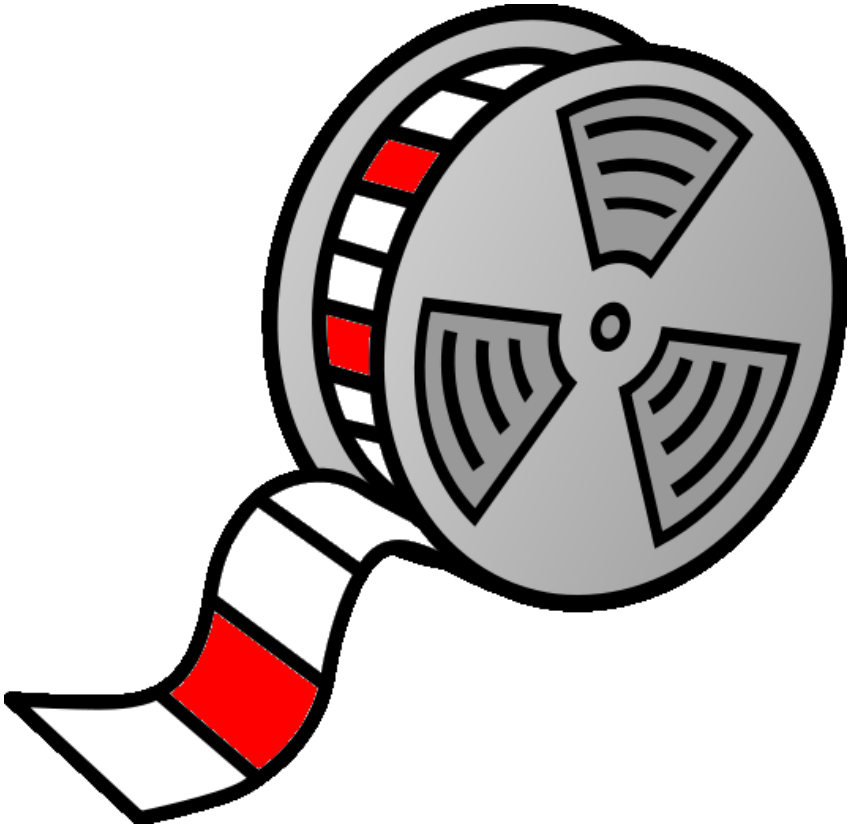
where $\alpha_p \equiv \alpha \bmod p$.

- $f(\theta) \bmod p = f_p(\theta \bmod p)$ whenever $\theta \not\equiv \alpha \bmod p$ (Fermat's Little Theorem)
- α_p is at least a t-sparse shift of $f_p(x)$

Pretty Picture #1



Pretty Picture #2



- Red squares indicate nonzero terms in the polynomial.
- The reel is the unit circle in $\mathbb{Z}_p$.

Outline of Algorithm

Input: Bound B on the bit length of the lacunary-shifted representation

1. Choose a prime p with $p \in O(B^{O(1)})$
2. Evaluate $f(1), \ldots, f(p-1) \bmod p$ to attempt to interpolate $f_p(x)$.
3. Use a dense sparsest shift method to compute α_p
4. Repeat Steps 1–3 enough times to recover α

Example

Unknown Polynomial in $\mathbb{Q}[x]$

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

1. Choose a prime p with $p \in O(B^{O(1)})$

Step 1

$$p = 11$$

Example

Unknown Polynomial in $\mathbb{Q}[x]$

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

1. Choose a prime p with $p \in O(B^{O(1)})$
2. Evaluate $f(1), \ldots, f(p-1) \bmod p$ to attempt to interpolate $f_p(x)$

Step 2

$$f(1), \ldots, f(p-1) \bmod p = 10, 9, 0, 0, 2, 5, 2, 5, 10, 3$$

$$f_p(x) = x^7 + x^6 + 2x^5 + 9x^3 + 2x^2 + 8x + 9$$

Example

Unknown Polynomial in $\mathbb{Q}[x]$

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

1. Choose a prime p with $p \in O(B^{O(1)})$
2. Evaluate $f(1), \ldots, f(p-1) \bmod p$ to attempt to interpolate $f_p(x)$
3. Use a dense sparsest shift method to compute α_p

Step 3

$$f_p(x) \equiv (x-3)^7 + 10(x-3)^4 \mod p$$

$$\alpha_p = 3$$

Example

Unknown Polynomial in $\mathbb{Q}[x]$

$$f(x) = (x-3)^{107} - 485(x-3)^{54}$$

1. Choose a prime p with $p \in O(B^{O(1)})$
2. Evaluate $f(1), \ldots, f(p-1) \bmod p$ to attempt to interpolate $f_p(x)$
3. Use a dense sparsest shift method to compute α_p
4. Repeat Steps 1–3 enough times to recover α

Step 4

$$\alpha_{11} = 3, \quad \alpha_{13} = 3, \quad \alpha_{17} = 3, \quad \ldots$$

$$\alpha = 3$$

Types of Failures

Two categories of failures:

- $f_p(x)$ is not computed correctly
- The sparsest shift of $f_p(x)$ is not α_p
  Equivalent to $\deg f_p(x) < 2t - 1$.

Next we develop sufficient conditions on p to avoid failure.

Exponents Vanish

$f_p(x)$ computed incorrectly

$$f(x) = 10(x-1)^{12} + 8(x-1)^3$$

$$p = 7$$

$$f_7(x) = 3(x-1)^0 + (x-1)^3$$

$$f(1) \bmod 7 = 0 \not\equiv 3 = f_7(1)$$

Condition: $(p-1) \nmid \max\{1, e_1\} \cdot e_2 \cdot e_3 \cdots e_t$

Test: Constant coeff. of computed $f_p(x)$ equals constant coeff. of $f(x)$ modulo p

Exponents Too Small

Sparsest shift of $f_p(x)$ is not α_p

$$f(x) = -4(x-2)^{145} + 14(x-2)^{26} + 3$$

$$p = 13$$

$$\begin{aligned} f_{13}(x) &= 9(x-2)^1 + (x-2)^2 + 3 \\ &\equiv (x-4)^2 + 12 \end{aligned}$$

Condition: $(p-1) \nmid e_t(e_t-1)(e_t-2)\cdots(e_t-(2t-2))$

Test: $\deg f_p(x) \geq 2B - 1 \geq 2t - 1$

Exponents Collide

Sparsest shift of $f_p(x)$ is not α_p

$$f(x) = 4(x-1)^{59} + 2(x-1)^{21} + 7(x-1)^{19} + 20$$

$$p = 11$$

$$\begin{aligned} f_{11}(x) &= 4(x-1)^9 + 2(x-1)^1 + 7(x-1)^9 + 9 \\ &= 2(x-1) + 9 \\ &\equiv 2(x-2) \end{aligned}$$

Condition: $(p-1) \nmid (e_t - e_1)(e_t - e_2)\cdots(e_t - e_{t-1})$

Test: $\deg f_p(x) \geq 2B - 1 \geq 2t - 1$

Coefficients Vanish

Sparsest shift of $f_p(x)$ is not α_p

$$f(x) = 69(x-5)^{42} - 12(x-5)^{23} + 4$$

$$p = 23$$

$$\begin{aligned} f_{23}(x) &= 0(x-5)^{20} + 11(x-5)^1 + 4 \\ &= 11(x-5) + 4 \\ &\equiv 11(x-13) \end{aligned}$$

Condition: $p \nmid c_t$

Test: $\deg f_p(x) \geq 2B - 1 \geq 2t - 1$

Sufficient Conditions

Definition

$$C = \max\{e_1, 1\} \cdot \prod_{i=2}^{t-1} e_i \cdot \prod_{i=1}^{t-1} (e_t - e_i) \cdot \prod_{i=0}^{2t-2} (e_t - i) \quad \leq 2^{4B^2}$$

Sufficient Conditions for Success

- $p \nmid c_t$
- $(p-1) \nmid C$

Approach

Choose primes $p = qk + 1$, for distinct primes q.
$q \mid (p-1)$, so $q \nmid C \Rightarrow (p-1) \nmid C$.

Choosing Primes Deterministically

1. Let $q_1, \ldots, q_k$ be the first k primes, for $k \in \tilde{O}(B^2)$.
2. Let p_i be the smallest prime s.t. $p_i \equiv 1 \bmod q_i$.
3. Set $\mathcal{P} = \{p_1, p_2, \ldots, p_k\}$.

Facts

- $|P| \in \Omega(B^2)$
- $p_i \in O(q_i^{5.5}) = \tilde{O}(B^{11})$ (Linnik, Heath-Brown '92)
- With ERH, $p_i \in \tilde{O}(q_i^2) = \tilde{O}(B^4)$ (Bach & Sorenson '96)

Probabilistic Algorithm

1. Choose a random prime $q \in O(B^2 \log B)$.
2. Choose random k's less than q until a prime $p = qk + 1$ is found.

We use Rousselet (1988) to show the density of primes of this type in the range $[q, q^2]$ is high, *once q is larger than some unknown constant*.

For this version, $p_i \in O(q_i^2) \in O\tilde{}(B^4)$.

Heuristic

Our analysis has been rather crude.
In fact, “most” primes will be good.

The following works well in practice:

1. Choose a random prime p in the range $[4B, 8B]$.

That’s it!

Complexity of Deterministic Algorithm

- Step 3 (computing sparsest shift of $f_p(x)$) dominates.

Bit Complexity

$$O(B^{78}(\log B)^{24} \log\log B)$$

Factor	Source
5.5	Bounds on Linnik's constant
2	Need $(p-1) \nmid C$ and $\log C \in O(B^2)$
7	Deterministic sparsest shift algorithm

This gives 77; one more factor from repeating $O(B)$ times.

Complexity of Probabilistic Algorithm

Bit Complexity

$$\tilde{O}(B^{17})$$

Factor	Source
2	Bounds on Linnik's constant using Rousselet
2	Need $(p-1) \nmid C$ and $\log C \in O(B^2)$
4	Probabilistic sparsest shift algorithm

Again, one more factor from $O(B)$ iterations.

Complexity of Heuristic

Bit Complexity

$$\tilde{O}(B^5)$$

This method gives us $p \in O(B)$ rather than $O(B^4)$ for probabilistic or $O(B^{11})$ for deterministic.

If we know $\alpha \in O(B)$, then we can just iterate once, for total cost $\tilde{O}(B^4)$.

Summary of Sparsest Shift Computation Techniques

Deterministic Algorithm

Actual complexity only $\tilde{O}(B^{29})$ unless ERH is false.

Probabilistic Algorithm

Always correct and (provably) probably much faster.

Heuristic

Faster in practice and provably never wrong, but might not terminate

Interpolation

Once α is known, we can construct a modular black box for evaluating $f(x + \alpha)$.

Then use lacunary interpolation along the lines of Kaltofen, Lakshman, & Wiley (1990) and Avendaño, Krick, & Pacetti (2006).

Conclusions

- Shifted-lacunary interpolation can be performed in polynomial time, for rational polynomials given by a modular black box.
- How to apply these techniques to other problems on lacunary polynomials?
- What about domains other than $\mathbb{Q}[x]$?
- What about multivariate rational polynomials?