New Algorithms for Lacunary (Supersparse) Polynomials

University of Waterloo

Mark Giesbrecht

Symbolic Computation Group
University of Waterloo
Waterloo, Ontario, Canada

Daniel Roche

University of Waterloo

1 Lacunary/Supersparse Polynomials

Let F be a field and $f(x) \in \mathsf{F}[x]$ of degree n, and write

$$f(x) = a_1 x^{e_1} + a_2 x^{e_2} + \cdots + a_t x^{e_t},$$

with $a_1, \ldots, a_t \in \mathsf{F} \setminus \{0\}$, $e_1, \ldots, e_t \in \mathbb{Z}^+$, and $e_1 < e_2 < \ldots < e_t = n$.

This corresponds to the *sparse representation* of $f(x)$ by a list of nonzero coefficient-exponent pairs $\langle (a_1, e_1), (a_2, e_2), \ldots, (a_t, e_t) \rangle$.

Size is $\sum_{i=1}^{t} (\text{size}(a_i) + \lg e_i)$

- Can be exponentially smaller than the dense size
- This representation is the default in Maple, Mathematica, etc.

1.1 Complexity Results

Operations on polynomials which are in P when the input is given in the dense representation may or may not be tractable when the input is given in the sparse representation.

For instance, we can interpolate [1; 7] and find low-degree factors [2; 9] of lacunary polynomals, but it is NP-Hard to compute GCDs [10]. For some basic operations, such as a divisibility test, neither a P-time algorithm nor a hardness result is known.

2 Sparsest Shift Interpolation

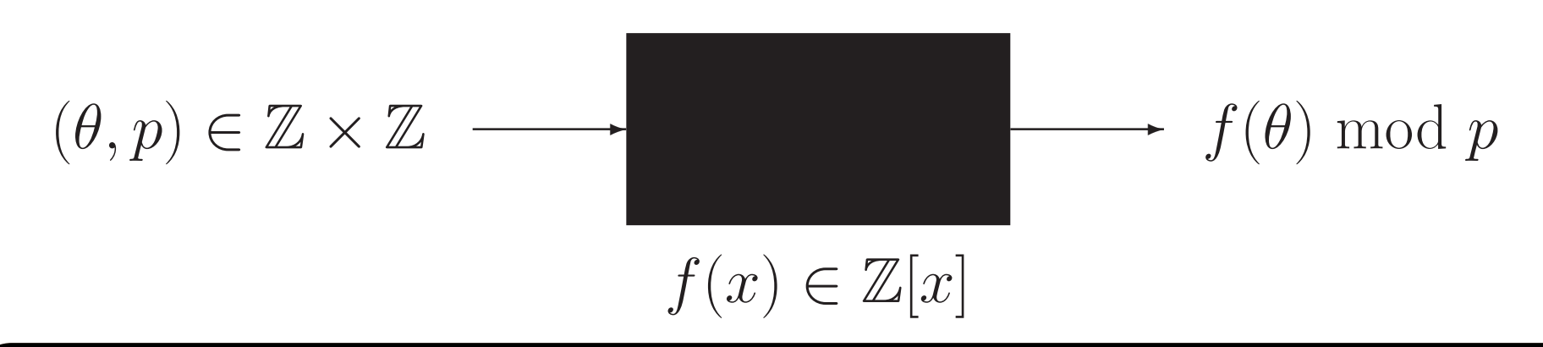
Definition (Sparse Shifts)**.** If $f(x)$ has at most t nonzero terms in the shifted power basis $1, (x-\alpha), (x-\alpha)^2, \ldots$, for some $\alpha \in \mathsf{F}$, then we say α is a *t-sparse shift* for $f(x)$.

Theorem (Lakshman & Saunders [8])**.** *If $t \le \frac{d+1}{2}$, then there is at most one t-sparse shift for a given polynomial $f(x) \in \mathsf{F}[x]$.*

P-time algorithms to find sparsest shift *when input is given as dense* [8; 5].

Goal: an algorithm to find the sparsest shift of $f(x)$ given a black box for evaluation, with complexity polynomial in the size of the sparsest shift.

We have a solution to a particular instance of this problem:
Let $f(x) \in \mathbb{Z}[x]$, and suppose we are given a black box which takes $\theta \in \mathbb{Z}$ and a prime p, and returns $f(\theta) \bmod p$.



The blackbox mod p model for interpolation

2.1 Preliminaries

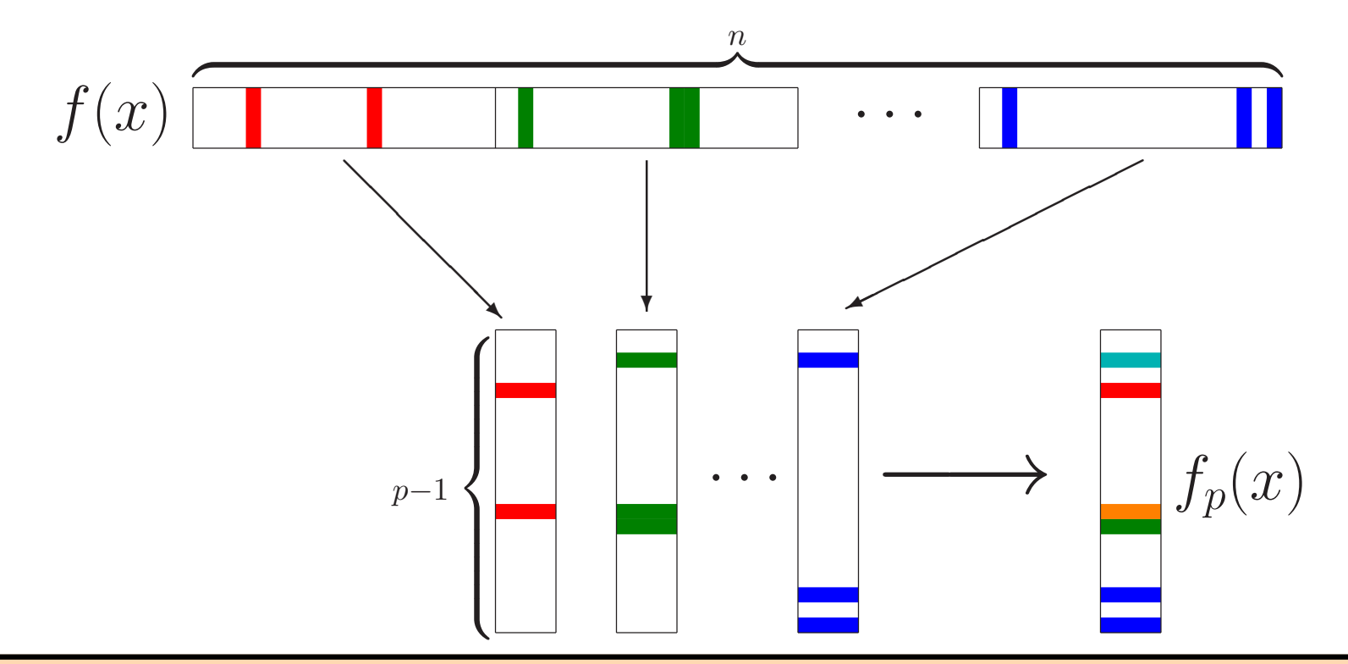
- Let p be a prime with $p \ge t^2$.
  From Fermat's Little Theorem, $a^{p-1} \equiv 1 \bmod p$ whenever $p \nmid a$.
  So $\exists\ f_p(x) \in \mathbb{Z}_p[x]$ with $\deg f_p \le p-2$ s.t. $f_p(\theta) \equiv f(\theta) \bmod p$ for all $\theta \in \mathbb{Z}$.
- If $f(x) = \sum_{i=1}^{t} a_i (x-\alpha)^{e_i}$, then

$$f_p(x) = \sum_{i=1}^{t} (a_i \bmod p)\,(x - (\alpha \bmod p))^{e_i \bmod (p-1)},$$

and therefore α is a t-sparse shift for $f_p(x)$.

2.2 Algorithm: Sparsest Shift Interpolation

1. Choose a prime p from a sufficiently large set such that $t^2 < p < t^{O(1)}$.
2. Use the black box to compute $v_i = f(i) \bmod p$ for $i = 1, 2, \ldots, p-1$.
3. Use (dense) Lagrange interpolation to find $f_p(x)$.
4. If $\deg(f_p(x)) \ge 2t-1$, then use the algorithm from [5] to find the sparsest shift α_p in $\mathbb{Z}_p$.
5. Repeat $O(\log \alpha)$ times until α can be recovered from the α_p's via Chinese Remaindering



The basic idea for sparse shift interpolation. The colored bars represent nonzero coefficients. The polynomial $f_p(x)$ is really a sum of sections of $f(x)$.

2.3 Comments on the algorithm

- If $\deg f_p \ge 2t-1$, then from [8], $\alpha \bmod p$ is the sparsest shift, since it is a t-sparse shift (from before).
- α must be the root of $n-t$ derivatives of $f(x)$.
  Roots of any derivative of $f(x)$ in $\mathbb{Z}$ are bounded by the maximal and minimal roots of $f(x)$ itself, which in turn must divide the trailing coefficient of $f(x)$.
  So the size of α is less than the size of $f(x)$.
- Still remains to construct the set of primes $\mathcal{S}$ such that $\deg f_p \ge 2t-1$ with high probability. This is true asymptotically, but we don't have practical bounds yet.
- Algorithm runs in polynomial time in the sparse size of $f(x+\alpha)$.

3 Polynomial Decomposition

Functional Decomposition Problem (univariate, simple): Given $f(x) \in \mathsf{F}[x]$, find $g(x), h(x) \in \mathsf{F}[x]$ with $\deg g, \deg h \ge 2$ and $f(x) = g(h(x))$.

Well-studied problem when input is given in the dense representation. The usual approach is to find $h(x)$ first, then use h to find $g(x)$.

3.1 Problem Statement and Simplifications

Problem:

Given $f(x)$, find $g(x)$ and $h(x)$ such that $f(x) = g(h(x))$.

- $f(x)$ is given in the α-shifted power basis
- $g(x)$ is returned in the sparsest shifted power basis, β
- $h(x)$ is returned in the α-shifted power basis
- Polynomial time in the size of the input *and* output

Can assume that f, g, h are all monic and $\alpha = \beta = 0$, since

$$\frac{f(x+\alpha)}{\mathrm{lc}(f)} = \left(\frac{g\left(\mathrm{lc}(h)(x+\beta)\right)}{\mathrm{lc}(f)} \right) \circ \left(\frac{h(x+\alpha)}{\mathrm{lc}(h)} - \beta \right)$$

And let $n = \deg f$, $r = \deg g$, and $s = \deg h$ so that $n = rs$.

3.2 Finding $h(x)$ of low degree

$f(x)$ and $h(x)$ agree in their high-order s coefficients (see figure). So if we define $\tilde{f}(x) = x^n f(\frac{1}{x})$ and $\tilde{h}(x) = x^s h(\frac{1}{x})$, the *reversals* of f and h, then

$$\tilde{f}(x) \equiv \tilde{h}(x)^r \mod x^s. \tag{1}$$

- This uniquely determines $h(x)$ up to the constant term.
- Can be solved with $O(s^{O(1)})$ field operations, as in [12]

So if s is small, we can find $h(x)$ in polynomial time in the sparse size of $f(x)$.

3.3 Certifying low-degree h

Question: How to efficiently check whether a given $h(x)$ is a right composition factor of $f(x)$?

Let $\Psi_h(x,y) = h(x) - h(y)$ and $\Psi_f(x,y) = f(x) - f(y)$

- $h(x)$ is a right composition factor of $f(x)$ iff $\Psi_h(x,y) \mid \Psi_f(x,y)$ [4]
- Note $\Psi_h(x,y)$ does not depend on $h(0)$

[6] gives a method to efficiently check whether a low-degree bivariate factor divides a high-degree sparse bivariate polynomial. We can use this method to efficiently (probabilistically) check whether $\Psi_h(x,y) \mid \Psi_f(x,y)$, thereby checking whether the $h(x)$ we have found is correct.

3.4 Finding $h(x)$ of high degree

Conjecture (Schinzel [11])**.** *If any power of a polynomial is sparse, then the polynomial itself must also be sparse.*

Subject to this conjecture, we can compute $h(x)$ (up to its constant coefficient) in polynomial time in the size of f and the size of h, by using a careful Newton-like iteration.

Let $\tilde{h}_1(x)$ and $\tilde{h}_2(x)$ be polynomials of degree k and l such that

$$\tilde{h}(x) \equiv \tilde{h}_1(x) + \tilde{h}_2(x) x^k \mod x^{k+l},$$

where $k, l \in \mathbb{Z}$ with $1 \le l \le k$ and $k + l \le s$.

Then, from (1) and the binomial theorem,

$$\tilde{f}(x) \equiv \tilde{h}_1(x)^r + r\tilde{h}_1(x)^{r-1}\tilde{h}_2(x) x^k \mod x^{k+l}. \tag{2}$$

Through some careful manipulation, we obtain

$$\tilde{h}_1(x)^{r+1} \equiv \tilde{h}_1(x)\tilde{f}(x) - r\tilde{f}(x)\tilde{h}_2(x) x^k \mod x^{k+l}.$$

So $\tilde{h}_1(x)^{r+1} \bmod x^{k+l}$ is sparse, and therefore from Shinzel's conjecture, we can compute it by repeated squaring.

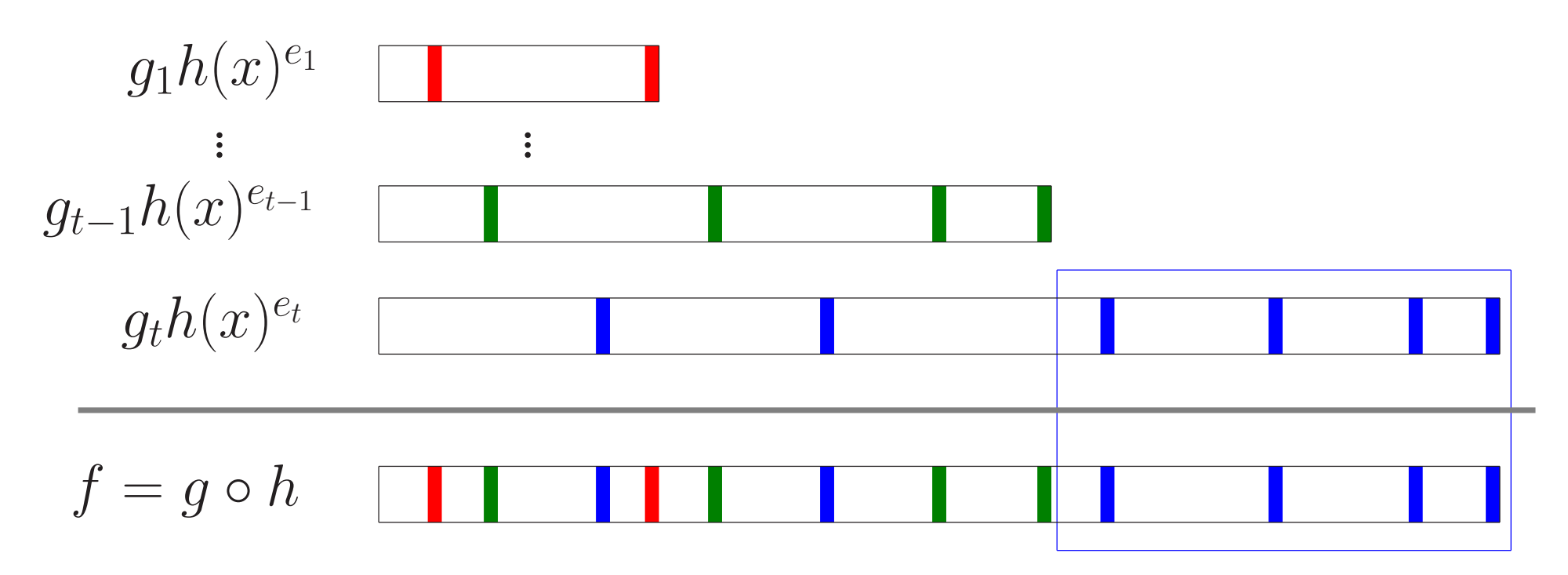
Manipulating (1) again, we see that

$$\left(\frac{1}{rx^k}\right)\left(\tilde{h}_1(x)\tilde{f}(x) - \tilde{h}_1(x)^{r+1}\right) \equiv \tilde{f}(x)\tilde{h}_2(x) \mod x^l.$$

We can compute the quotient of the left-hand side divided by $\tilde{f}(x) \bmod x^l$ in polynomial time since the quotient, $\tilde{h}_2(x)$, is sparse, and $\tilde{f}(x)$ has constant coefficient equal to 1.

So, to find $h(x)$, we start with $\tilde{h}_1(x) = 1$ (since $h(x)$ is monic), and repeat the iteration approximately $\log_2 s$ times to recover $h(x)$ in polynomial time in the size of f and h.

Suppose $g(x)$ in sparse form is $g_1 x^{e_1} + g_2 x^{e_2} + \cdots + g_t x^{e_t}$.



An illustration of the composition of two polynomials. Since we can assume $g_t = 1$ and $e_t = r$, the top terms of $f(x)$ agree with those of $h(x)^r$.

3.5 Finding $g(x)$ when r is small

We now show how to find $g(x)$ when $h(x) - h(0)$ is known, using dense interpolation.

1. Choose $r+1$ distinct points $\theta_0, \ldots, \theta_r \in \mathsf{F}$
2. Compute $u_i = h(\theta_i) - h(0)$ and $v_i = f(\theta_i)$ for $i = 0, \ldots, r$
3. Use Lagrange interpolation to compute $g(x + h(0))$.
4. Use the sparsest shift algorithm of [5] to find $h(0)$, and finally compute $g(x)$ and $h(x)$

We need all the u_i's to be distinct; the Schwartz-Zippell Lemma guarantees this with high probability if the θ_i's are chosen from a large enough set.

Also note that over some fields (for instance $\mathbb{Z}$), evaluating a large sparse polynomial at a point is actually intractable (the size of the output can be exponentially large). In this case, a modular evaluation approach combined with Chinese Remaindering will likely be necessary.

4 Future Work

- Using our sparsest shift interpolation algorithm to find $g(x)$ of high degree given $h(x)$
- Extending the sparsest shift interpolation algorithm to work over fields other than $\mathbb{Z}[x]$
- Eliminating the dependency of the algorithm for finding high-degree $h(x)$ on any conjectures
- Removing the output-sensitivity of the runtime (i.e. proving that $h(x)$ and $g(x)$ are always sparse when $f(x)$ is sparse) — relates to [3] (and many others).
- Finding an algorithm to certify candidate right composition factors $h(x)$ of high degree. Note that an algorithm to perform a parse polynomial divisibility check would solve this.

References

[1] Ben-Or and Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation. *STOC '88*.

[2] Cucker, Koiran, Smale. A polynomial time algorithm for Diophantine equations in one variable. *JSC*, 1999.

[3] P. Erdös. On the number of terms of the square of a polynomial. *Nieuw Arch. Wiskunde (2)*, 23:63–65, 1949.

[4] M. D. Fried and R. E. MacRae. On the invariance of chains of fields. *Illinois J. Math.*, 13:165–171, 1969.

[5] Giesbrecht, Kaltofen, Lee. Algorithms for computing sparsest shifts of polynomials in power, Chebyshev and Pochhammer bases. *JSC*, 2003. (ISSAC'2002).

[6] Kaltofen and Koiran. On the complexity of factoring bivariate supersparse (lacunary) polynomials. *ISSAC'05*

[7] Kaltofen and Lee. Early termination in sparse interpolation algorithms. *JSC*, 2003. (ISSAC'2002).

[8] Y. N. Lakshman and B. D. Saunders. Sparse shifts for univariate polynomials. *Appl. Algebra Engrg. Comm. Comput.*, 7(5):351–364, 1996.

[9] H. W. Lenstra, Jr. Finding small degree factors of lacunary polynomials. In *Number theory in progress, Vol. 1 (Zakopane-Kościelisko, 1997)*, pages 267–276. de Gruyter, Berlin, 1999.

[10] D. A. Plaisted. New NP-hard and NP-complete polynomial and integer divisibility problems. *Theoret. Comput. Sci.*, 31(1-2):125–138, 1984.

[11] A. Schinzel. On the number of terms of a power of a polynomial. *Acta Arith.*, 49(1):55–70, 1987.

[12] J. von zur Gathen. Functional decomposition of polynomials: the tame case. *JSC*, 1990.