Rafael Oliveira
Princeton University

Factors of Low Individual Degree Polynomials

Outline

1. Introduction and Background
2. Main Ideas of this Work
3. Conclusion & Open Questions

1 Introduction & Background

Factoring in Real Life

Basic routine in many tasks:

Fast decoding of Reed Solomon Codes

Used to compute:
- Primary Decompositions of Ideals
- Gröbner Bases, etc.

Can be done efficiently in (randomized) poly time!

In theory, interested in:
- Derandomization
- Parallel complexity
- Structure of factors

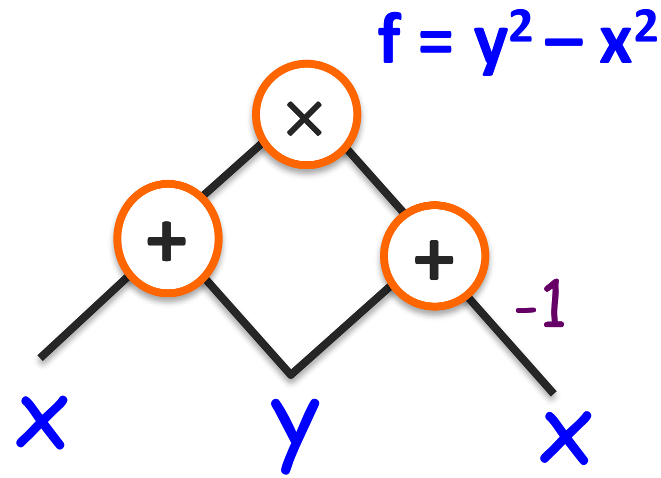
Arithmetic Circuits

Definition by picture

Main measures:

Size = # edges

Depth = length of longest path from root to leaf



Model captures our notion of algebraic computation

Many interesting polynomials have succinct rep. in this model, such as $Det_n(X), \sigma_k(x_1, \ldots, x_n)$.

It is a major open question whether $Perm_n$ has a succinct rep. in this model.

Polynomial Factorization

Problem: Given a circuit for $P(\mathbf{x})$, where

$$P(\mathbf{x}) = g_1(\mathbf{x})g_2(\mathbf{x})\ldots g_k(\mathbf{x})$$

output circuits for $g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_k(\mathbf{x})$

- **[LLL ’82, Kal ’89]:** if $P(\mathbf{x})$ is computed by a small circuit, then so are the factors $g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_k(\mathbf{x})$. Moreover Kaltofen gives a randomized algorithm to compute factors
- **Fundamental consequences to:**
  - Circuit Complexity & Pseudorandomness: [KI ’04, DSY ’09]
  - Coding Theory: [Sud ’97, GS’06]
  - Geometric Complexity Theory: [Mul’13]

What About Depth?

[Kaltofen '89]: factorization behaves nicely w.r.t. size.

What about depth?

More generally:

Structure: given polynomial $P(\mathbf{x})$ in circuit class $\mathcal{C}$, which classes $\mathcal{C}^*$ efficiently compute the factors of $P(\mathbf{x})$?

- If $P(\mathbf{x})$ has a small depth circuit, do its factors have small depth circuits?
- If $P(\mathbf{x})$ has a small formula, do its factors have small formula?

Gap of Understanding

If $P(\mathbf{x})$ is a polynomial with s monomials and degree d

Kaltofen & depth reduction

Factors of $P(\mathbf{x})$ computed by formulas of depth 4 and size $\exp(\tilde{O}(\sqrt{d}))$.

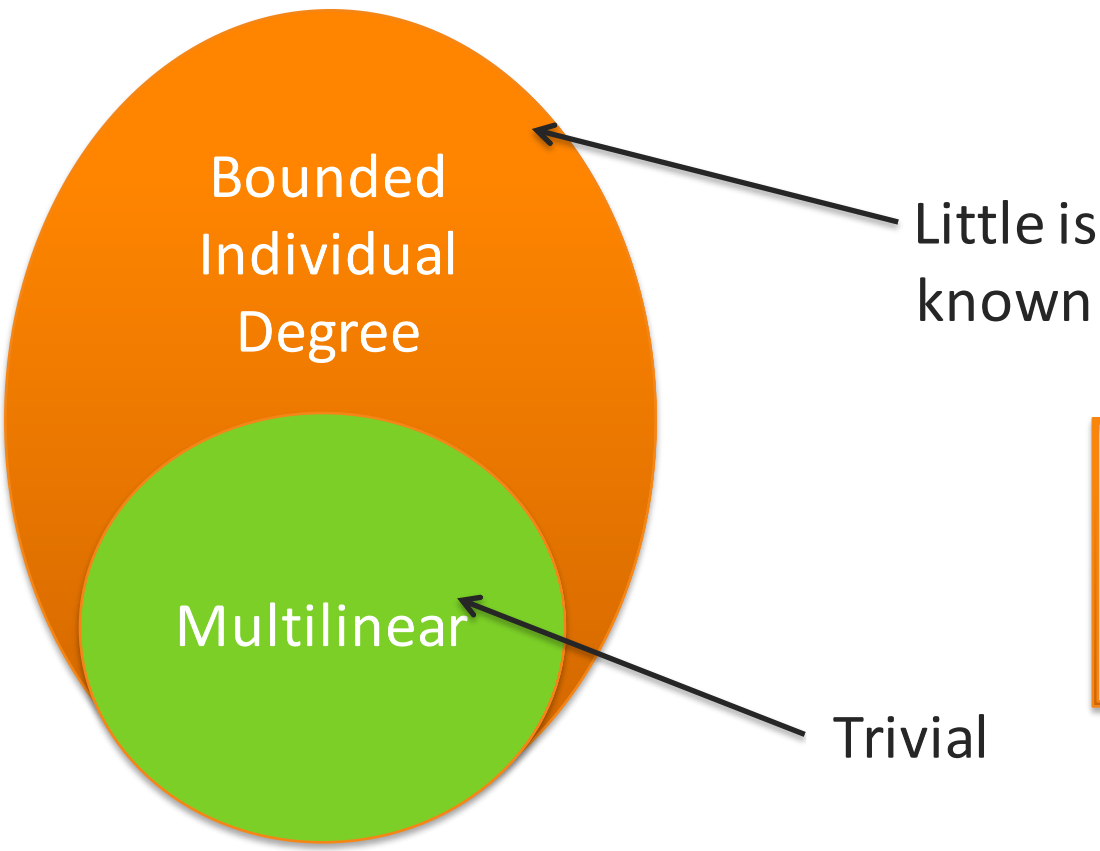
General depth reductions **[AV'08, Koi'12, GKKS'13, Tav'13]** give subexponential gap.

Can this be improved?

Why Bound Individual Degrees?

Polynomials with bounded ind. deg. form a very rich class, which generalizes multilinear polynomials.

Well studied, works of **[Raz ’06, RSY ’08, Raz ’09, SV ’10, SV ’11, KS ’15[2], KCS’15, KCS’16].**



Step towards understanding general case

This Work

Theorem: If $P(\mathbf{x})$ is a polynomial which:

- has individual degrees bounded by r,
- is computed by a circuit (formula) of size s & depth d

Then any factor $f(\mathbf{x})$ of $P(\mathbf{x})$ is computed by a circuit (formula) of size

$$\text{poly}(n^r, s)$$

& depth

$$d + 5$$

Furthermore, result provides a randomized algorithm for computing all factors of $P(\mathbf{x})$ in time $\text{poly}(n^r, s)$

Prior Work

[DSY '09]: if $P(\mathbf{x}, y)$ is computed by a circuit of size s, depth d

- $\deg_y(P)$ is bounded by r

Then its factors of the form $y - g(\mathbf{x})$ have circuits of depth $d+3$ and size poly(n^r, s)

Extend Hardness vs Randomness approach of **[KI '04]** to bounded depth circuits.

[DSY '09] noticed that only factors of the form $y - g(\mathbf{x})$ are important to extend **[KI '04]** to bounded depth.

Main Ideas of this Work

- Lifting
- Root Approximation
- Reversal
- Outline

Lifting

Suppose input is:

$$P(\mathbf{x}, y) = (y - g_1(\mathbf{x}))(y - g_2(\mathbf{x}))$$

Where

$$\mu_1 = g_1(\mathbf{0}), \mu_2 = g_2(\mathbf{0}) \text{ and } \mu_1 \neq \mu_2$$

How do we factor in this case?

Can try to build the homogeneous parts of $g_i(\mathbf{x})$ one at a time.

Lifting

Note that:

$$P(\mathbf{0}, y) = (y - \mu_1)(y - \mu_2)$$

Which we know how to factor.

Hence, found the constant terms of the roots.

How to find the linear terms of the roots?

Lifting

Setting $y = \mu_1$ in the input polynomial:

$$P(\mathbf{x}, \mu_1) = (\mu_1 - g_1(\mathbf{x}))(\mu_1 - g_2(\mathbf{x}))$$

Since $\mu_1 \neq \mu_2$, the constant term of

$$\mu_1 - g_2(\mathbf{x})$$

is nonzero, whereas the constant term of

$$\mu_1 - g_1(\mathbf{x})$$

is zero! Hence, linear term of $P(\mathbf{x}, \mu_1)$ equals the linear term of $g_1(\mathbf{x})$, up to a constant factor.

Lifting

Continuing this way, we can recover the roots and factor the input polynomial.

Hensel Lifting/Newton Iteration.
Pervasive in factoring algorithms, such as **[Zas ’69, Kal ’89, DSY ’09]**, and many others.

[DSY ’09]: if $P(\mathbf{x}, y)$ is computed by a circuit of size s, depth d

- $\deg_y(P)$ is bounded by r

Then its factors of the form $y - g(\mathbf{x})$ have circuits of depth $d+3$ and size $\text{poly}(n^r, s)$

Lifting

Two main issues

- What if $P(\mathbf{x}, y)$ does not factor into linear factors in y?

  Approximate roots in algebraic closure of $\mathbb{F}(\mathbf{x})$ by low degree polynomials in $\mathbb{F}[\mathbf{x}]$.

- What if $P(\mathbf{x}, y)$ is not monic in y?

  Use reversal to reduce the number of variables

2 Main Ideas of this Work

- Lifting
- Root Approximation
- Reversal
- Outline

Approximation Polynomials

Suppose input is:

$$P(\mathbf{x}, y) = y^r + \sum_{i=0}^{r-1} P_i(\mathbf{x}) y^i$$

Which does not factor into linear factors. Let

$$P(\mathbf{x}, y) = f(\mathbf{x}, y) Q(\mathbf{x}, y)$$

where

$$f(\mathbf{x}, y) = y^k + \sum_{i=0}^{k-1} f_i(\mathbf{x}) y^i$$

Is irreducible and does not divide the other factor.

Approximation Polynomials

Any polynomial factors completely in the algebraic closure of $\mathbb{F}(\mathbf{x})$!

$$P(\mathbf{x}, y) = \prod_{i=1}^{r} (y - \varphi_i(\mathbf{x}))$$

⬇

$$f(\mathbf{x}, y) = \prod_{i=1}^{k} (y - \varphi_i(\mathbf{x}))$$

Where each $\varphi_i(\mathbf{x})$ is a "function" on the variables $\mathbf{x}$

Approximation Polynomials

Since $P(\mathbf{x}, y)$ and $f(\mathbf{x}, y)$ share roots $\varphi_i(\mathbf{x})$, can try to approximate these roots by polynomials $g_{i,t}(\mathbf{x})$ of degree t such that

$$f(\mathbf{x}, g_{i,t}(\mathbf{x}))$$

only has terms of degree higher than t.

Definition: we say that

$$f(\mathbf{x}) =_t g(\mathbf{x})$$

if the polynomial $f(\mathbf{x}) - g(\mathbf{x})$ only has terms of degree higher than t.

Approximation Polynomials

Definition: we say that

$$f(\mathbf{x}) =_t g(\mathbf{x})$$

if the polynomial $f(\mathbf{x}) - g(\mathbf{x})$ only has terms of degree higher than t.

This definition gives us a topology:

- Two polynomials are close if they agree on low degree parts
- Can use this topology to derive analogs of Taylor series for elements of $\overline{\mathbb{F}(\mathbb{x})}$.

Can “approximate” elements of $\overline{\mathbb{F}(\mathbb{x})}$ by polynomials!

Approximation Polynomials

If we can find $g_{i,t}(\mathbf{x})$ for each root $\varphi_i(\mathbf{x})$ of $f(\mathbf{x}, y)$ such that

$$f(\mathbf{x}, g_{i,t}(\mathbf{x})) =_t 0$$

Then we can prove the following:

Lemma: the polynomials $g_{i,t}(\mathbf{x})$ are such that

$$f(\mathbf{x}, y) =_t \prod_{i=1}^{k} (y - g_{i,t}(\mathbf{x}))$$

Can convert approximations to the roots into approximations to the factors!

Approximation Polynomials

How do we obtain these polynomials $g_{i,t}(\mathbf{x})$?

Since each $\varphi_i(\mathbf{x})$ is also a root of $P(\mathbf{x}, y)$, can obtain $g_{i,t}(\mathbf{x})$ from $P(\mathbf{x}, y)$ via lifting!

Looking at our parameters:

$$f(\mathbf{x}, y) =_t \prod_{i=1}^{k} (y - g_{i,t}(\mathbf{x}))$$

Depth $d + 4$ size poly(n^r, s)

With standard techniques, can recover $f(\mathbf{x}, y)$

fl Observation: for the general case, need to keep the product top fan in!

$i=1$

Main Ideas of this Work

Lifting

Root Approximation

Reversal

Outline

Set Up

Suppose input now is:

$$P(\mathbf{x}, y) = \sum_{i=0}^{r} P_i(\mathbf{x}) y^i, \;\; P_0(\mathbf{x}) P_r(\mathbf{x}) \neq 0$$

Let

$$P(\mathbf{x}, y) = f(\mathbf{x}, y) Q(\mathbf{x}, y)$$

where

$$f(\mathbf{x}, y) = \sum_{i=0}^{k} f_i(\mathbf{x}) y^i$$

is irreducible and does not divide the other factor.

The Game Plan

Reduce to the monic case:

$$P(\mathbf{x}, y) = P_r(\mathbf{x}) \cdot \left(y^r + \sum_{i=0}^{r-1} \frac{P_i(\mathbf{x})}{P_r(\mathbf{x})} y^i \right)$$

$$f(\mathbf{x}, y) = f_k(\mathbf{x}) \cdot \left(y^k + \sum_{i=0}^{k-1} \frac{f_i(\mathbf{x})}{f_k(\mathbf{x})} y^i \right)$$

1. Recover $f_k(\mathbf{x})$ from $P_r(\mathbf{x})$ by some kind of induction
2. Recover the part of $f(\mathbf{x}, y)$ that depends on y

Naïve Recursion

Let $P(\mathbf{x}, y)$ have individual degrees r, n variables and computed by circuit of size s and depth d

Let $T(s, n)$ be such that:

$$f(\mathbf{x}, y) \mid P(\mathbf{x}, y)$$

$$\Downarrow$$

There exists $\Phi(\mathbf{x}, y)$ with

$$\Phi(\mathbf{x}, y) =_t f(\mathbf{x}, y)$$

- depth $d + 4$
- size $\leq T(s, n)$
- top fan in product gate

Naïve Recursion

Our recurrence becomes:

$$T(s,n) \le \underbrace{T(3rs, n-1)}_{\text{Recover } f_k(\mathbf{x}) \text{ fro}} + \underbrace{\mathsf{poly}(n^r, s)}_{\text{Size of part depending on } y}$$

After t steps, our recursion would become

$$T(s,n) \le T((3r)^t s, n-t) + \Omega(n^{tr} s)$$

Exponential when $t \sim n$!

Dealing with Exp. Growth

How do we avoid exponential growth?

It is hard to get $P_r(\mathbf{x})$ from $P(\mathbf{x}, y)$, but it is easy to get $P_0(\mathbf{x})$ from $P(\mathbf{x}, y)$

$$P_0(\mathbf{x}) = P(\mathbf{x}, 0)$$

$P_0(\mathbf{x})$ has smaller circuit size than $P(\mathbf{x}, y)$!

What if we could make $P_0(\mathbf{x})$ the leading coefficient of $P(\mathbf{x}, y)$?

Reversal

Definition by example: If

$$P(x,y) = P_5(x)y^5 + P_4(x)y^4 + P_0(x)$$

Then its reversal is defined as

$$\tilde{P}(x,y) = P_0(x)y^5 + P_4(x)y + P_5(x)$$

The reversal can be efficiently computed from circuit computing original polynomial.

Recursion with Reversal

If we take the reversal to compute the factors, our recurrence for $T(s,n)$ becomes

$$T(s,n) \leq \underbrace{T(s,n-1)}_{\text{Recover } f_0(\mathbf{x}) \text{ from } P_0(\mathbf{x})} + \underbrace{\text{poly}(n^r, 9r^2 s)}_{\text{of part depending on } y}$$

After t steps, our recursion remains

$$T(s,n) \leq T(s,n-t) + \text{poly}(n^r, 9r^2 s)$$

No exponential growth!

2 Main Ideas of this Work

Lifting

Root Approximation

Reversal

Outline

Outline

$$P(\mathbf{x}, y) = f(\mathbf{x}, y)Q(\mathbf{x}, y) \Rightarrow \tilde{P}(\mathbf{x}, y) = \tilde{f}(\mathbf{x}, y)\tilde{Q}(\mathbf{x}, y)$$

Size becomes $9r^2 s$
Depth remains d

$$\tilde{P}(\mathbf{x}, y) =_t P_0(\mathbf{x}) \cdot G(\mathbf{x}, y)$$

Monic in y

$$\tilde{f}(\mathbf{x}, y) =_t f_0(\mathbf{x}) \cdot g(\mathbf{x}, y)$$

Monic in y

Outline

Each approximate root of $g(\mathbf{x}, y)$ is also approx. root of $G(\mathbf{x}, y)$

$$g(\mathbf{x}, y) =_t \prod_{i=1}^{k} (y - g_{i,t}(\mathbf{x}))$$

Size $\text{poly}(s, n^r)$
Depth $d + 3$
Top gate: addition gate

By induction, $f_0(\mathbf{x}) =_t h(\mathbf{x})$

Size $\text{poly}(s, n^r)$
Depth $d + 4$
Top gate: product gate

Outline

$$\tilde{f}(\mathbf{x}, y) =_t h(\mathbf{x}) \cdot g(\mathbf{x}, y)$$

Size $\text{poly}(s, n^r)$
Depth $d + 4$
Top gate: product gate

$\tilde{f}(\mathbf{x}, y)$ computed by circuit of

Size $\text{poly}(s, n^r)$
Depth $d + 5$
Top gate: addition gate

3 Conclusions and Open Problems

This Work - Recap

We showed: If $P(\mathbf{x})$ is a polynomial with individual degrees bounded by r, and has a small low-depth circuit (formula), then any factor $f(\mathbf{x})$ of $P(\mathbf{x})$ is computed by a small low-depth circuit (formula).

Furthermore, result provides a randomized algorithm for computing all factors of $P(\mathbf{x})$ in time $\text{poly}(n^r, s)$

General Framework

In **[SY '10]**, it is asked whether factors of low depth circuits have poly size circuits of low depth, without the bounded degree restriction.

Question open even for factors of the form $y - g(\mathbf{x})$

Theorem: If $P(\mathbf{x}, y)$ is a polynomial computed by a low depth circuit, and all its approximate roots are computed by small low depth circuits, then any factor of $P(\mathbf{x}, y)$ is computed by small low depth circuits.

Corollary: To settle above conjecture, it is enough to solve question above for approximate roots, instead of factors of the form $y - g(\mathbf{x})$.

Open Questions

- Remove exponential dependence on the degree for factors of the form $y - g(\mathbf{x})$
- Reduce the depth bounds in the work of **[DSY '09]**
  - Can we show that factors of sparse have small depth 4 circuits?
- Derandomize polynomial factorization, even for bounded individual degree polynomials.
  - Question is open even for sparse polynomials
  - Will require stronger PITs than current techniques

Thank you!