Constant-Depth Arithmetic Circuits for Linear Algebra Problems

Robert Andrews
Waterloo

Avi Wigderson
IAS

arXiv: 2404.10839

Greatest Common Divisors

Given polynomials $f(x), g(x) \in \mathbb{F}[x]$, their greatest common divisor is the highest-degree polynomial that divides f and g

Question: Given coefficients of f and g, how many arithmetic ops are needed to compute $\gcd(f, g)$?

- Fundamental operation in computer algebra
- Applications to
  - Polynomial factorization
  - Coding theory
  - ⋮

Computing the GCD I

Theorem [Euclid, Stevin 1585, Newton 1707]

Given $f, g \in \mathbb{F}[x]$ of degree $\leq d$, can compute $\gcd(f, g)$ using $O(d^3)$ arithmetic ops

Theorem [Knuth 1970, Schönhage '71, Moench '73]

Compute $\gcd(f, g)$ using $O(d \log^2 d)$ arithmetic ops

Both algorithms are sequential. Parallel algo?

Computing the GCD II

Theorem [Borodin, von zur Gathen, Hopcroft '82]

Compute $\gcd(f, g)$ in $d^{O(1)}$ arithmetic ops and $O(\log^2 d)$ parallel time

Main idea:

(1) Express $\gcd(f, g)$ as solution of system of linear equations

(2) Invoke parallel algos for determinant and matrix inverse

Computing the GCD II

Fact: $\gcd(f, g) = 1$ $\quad f = \sum_{i=0}^{n} f_i x^i \quad g = \sum_{i=0}^{m} g_i x^i$

$\Updownarrow$

$\exists$ polynomials s, t

- $\deg(s) < \deg(g)$
- $\deg(t) < \deg(f)$
- $s \cdot f + t \cdot g = 1$

linear equations in coeffs of s, t

$\Leftrightarrow$ $\det \begin{pmatrix} f_n & & & & g_m & & \\ f_{n-1} & \ddots & & & \vdots & \ddots & \\ \vdots & & f_n & & g_0 & & g_m \\ f_0 & & \vdots & & & \ddots & \vdots \\ & \ddots & f_0 & & & & g_0 \end{pmatrix} \neq 0$

m n

"resultant of f and g"

Computing the GCD II

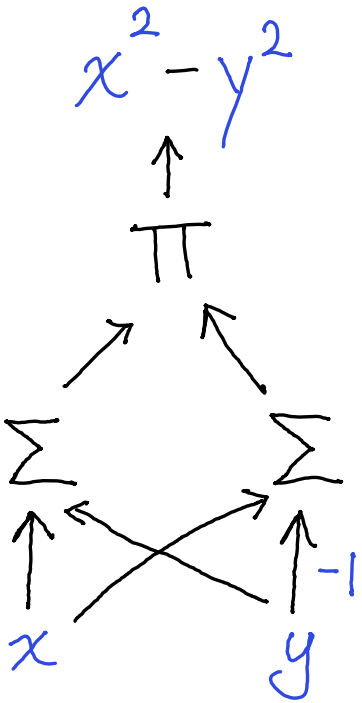
To decide if $\gcd(f, g) = 1$, check if

$$0 \overset{?}{\neq} \operatorname{res}(f, g) := \det \begin{pmatrix} f_n & & & & g_m & & \\ f_{n-1} & \ddots & & & \vdots & \ddots & \\ \vdots & & f_n & & g_0 & & g_m \\ f_0 & & \vdots & & & & \vdots \\ & \ddots & f_0 & & & & g_0 \end{pmatrix}$$

Theorem [Csanky '76, Berkowitz '82]

Compute $n \times n$ determinant with algebraic circuit of size $\text{poly}(n)$ and depth $O(\log^2 n)$

Algebraic Circuits



Circuits built from addition, multiplication, and division gates

Inputs: numbers $x, y, \ldots$

Output: number(s) $f(x, y, \ldots)$

Size: # of gates

Depth: distance from input to output

Faster Parallel Algo?

Question: Compute $n \times n$ determinant with algebraic circuit of size $\text{poly}(n)$ and depth $o(\log^2 n)$?

Conjecture: Probably not (Equivalent to $\text{trace}(X^n)$, $\text{IMM}_{n,n}$, etc.)

Theorem [Limaye, Srinivasan, Tavenas '21, Forbes '24]

No circuits of size $\text{poly}(n)$ and depth $O(1)$

Our Results

Theorem [A. & Wigderson]

Given polys $f, g \in \mathbb{C}[x]$ of degree d, can compute

- resultant of f and g
- $\gcd(f, g)$
- polys s, t such that $s \cdot f + t \cdot g = \gcd(f, g)$
- squarefree decomposition of f and g

with algebraic circuits* of size $d^{O(1)}$ and depth $O(1)$

* some branching on "$x = 0$?" required

Tool 1: Interpolation

Write $f(x_1, \ldots, x_n, y)$ as

$$f(x_1, \ldots, x_n, y) = \sum_{i=0}^{d} f_i(x_1, \ldots, x_n) \cdot y^i$$

Lemma:

compute $f(x_1, \ldots, x_n, y)$ in size s & depth Δ

$\Rightarrow$

$\forall i \in \{0, 1, \ldots, d\}$, compute $f_i(x_1, \ldots, x_n)$ in size $O(sd)$ & depth $\Delta + 1$

Sketch: Polynomial interpolation.

$\exists c_{i,j} \in \mathbb{Q}$ s.t. $f_i(\vec{x}) = \sum_{j=0}^{d} c_{i,j} \cdot f(\vec{x}, j)$ $\square$

Elementary Symmetric Polynomials

$$e_d(x_1, \ldots, x_n) := \sum_{\substack{S \subseteq [n] \\ |S| = d}} \prod_{i \in S} x_i$$

Definition $f(\bar{x})$ is <u>symmetric</u> if $\forall$ permutations π,

$$f(x_1, \ldots, x_n) = f(x_{\pi(1)}, \ldots, x_{\pi(n)})$$

Theorem $\forall$ symmetric $f(\bar{x})$, $\exists$ poly $g(y_1, \ldots, y_n)$

$$f(x_1, \ldots, x_n) = g(e_1(\bar{x}), \ldots, e_n(\bar{x}))$$

Elementary Symmetric Polynomials

$$e_d(x_1, \ldots, x_n) := \sum_{\substack{S \subseteq [n] \\ |S| = d}} \prod_{i \in S} x_i$$

Fact If $f(x) = \prod_{i=1}^{n} (x - \alpha_i)$, $\alpha_i \in \mathbb{C}$

then $f(x) = \sum_{i=0}^{n} (-1)^i e_i(\bar{\alpha}) x^{n-i}$

Interpolation: Example

Theorem [Ben-Or 198X]
For every $d \in [n]$, there is a circuit of size $O(n^2)$ and depth 3 that computes

$$e_d(x_1, \ldots, x_n) := \sum_{S \in \binom{[n]}{d}} \prod_{i \in S} x_i$$

Proof: Apply interpolation to

$$\prod_{i=1}^{n} (1 + y \cdot x_i) = \sum_{i=0}^{n} e_i(\vec{x}) \cdot y^i \qquad \square$$

Power Sum Polynomials

$$p_d(x_1, \ldots, x_n) := x_1^d + \cdots + x_n^d$$

Theorem $\forall$ symmetric $f(\bar{x})$, $\exists$ poly $g(y_1, \ldots, y_n)$

$$f(x_1, \ldots, x_n) = g\left(p_1(\bar{x}), \ldots, p_n(\bar{x})\right)$$

Question. Explicit conversion

$$\left(e_1(\bar{x}), \ldots, e_n(\bar{x})\right) \longleftrightarrow \left(p_1(\bar{x}), \ldots, p_n(\bar{x})\right) \; ?$$

Tool 2: Newton's Identities

Fact: there are explicit low-depth identities between $(e_i(\vec{x}))_i$ and $(p_i(\vec{x}))_i$

Lemma Given input $p_1, \dots, p_n$, can compute $e_1, \dots, e_n$ in $\text{poly}(n)$ size & $O(1)$ depth

Proof Use explicit identity + interpolation $\square$

Lemma Given input $e_1, \dots, e_n$, can compute $p_1, \dots, p_m$ in $\text{poly}(m)$ size & $O(1)$ depth

$\uparrow$

$m \geq n$

Division

Lemma Given input $f, g \in \mathbb{C}[x]$ where $g \mid f$, can compute f/g in $\text{poly}(n)$ size, $O(1)$ depth

Proof Write $f = \prod_i (x - \alpha_i) \prod_i (x - \beta_i)$

$$g = \prod_i (x - \alpha_i)$$

1) Compute $p_d(\bar{\alpha}, \bar{\beta})$ & $p_d(\bar{\alpha})$ via Newton's identities
2) Subtract: $p_d(\bar{\beta}) = p_d(\bar{\alpha}, \bar{\beta}) - p_d(\bar{\alpha})$
3) Compute $e_d(\bar{\beta})$ via Newton's identities $\square$

[Bini, Pan '85] Division w/ remainder

Roots of Perfect Powers

Lemma Given input $f \in \mathbb{C}[x]$ where $f = g^r$, can compute g in $\text{poly}(n)$ size, $O(1)$ depth

Proof Write $f = \prod_i (x - \alpha_i)^r$

$g = \prod_i (x - \alpha_i)$

Newton's identities applied to f

1) Compute $p_d(\underbrace{\alpha_1, \ldots, \alpha_1}_{r \text{ times}}, \alpha_2, \ldots) = r \cdot p_d(\bar{\alpha})$

2) Divide: $p_d(\bar{\alpha}) = \frac{1}{r} \cdot r \cdot p_d(\bar{\alpha})$

3) Compute $e_d(\bar{\alpha})$ via Newton's identities $\square$

Summing over Roots

Lemma: Given coefficients of $f(x) = \prod_{i=1}^{n} (x - \alpha_i)$ and $g(x)$, can compute

$$\sum_{i=1}^{n} g(\alpha_i)$$

in $\text{poly}(n, m)$ size and $O(1)$ depth.

Proof:

(1) Compute $\alpha_1^j + \cdots + \alpha_n^j$ for all $0 \le j \le \deg(g)$ via Newton's identities

(2) $$\sum_{i=1}^{n} g(\alpha_i) = \sum_{j=0}^{m} g_j \cdot \left(\sum_{i=1}^{n} \alpha_i^j \right)$$ □

Symmetric Funcs of Roots

Lemma: Given coefficients of $f(x) = \prod_{i=1}^{n} (x - \alpha_i)$ and $g(x)$, can compute

$$e_d\left(g(\alpha_1), \ldots, g(\alpha_n)\right)$$

in $\text{poly}(n, m)$ size and $O(1)$ depth.

Proof:

(1) $\forall\, 1 \le k \le n$, interpolate coeffs of $g(x)^k$

(2) $\forall\, 1 \le k \le n$, compute $\sum_{i=1}^{n} g(\alpha_i)^k$

(3) Apply Newton's identities $\square$

The Resultant

Fact: $\gcd(f, g) = 1 \iff \operatorname{res}(f, g) \neq 0$

$$\operatorname{res}(f, g) := \det \begin{pmatrix} f_n & & & g_m & & \\ f_{n-1} & \ddots & & \vdots & \ddots & \\ \vdots & & f_n & g_0 & & g_m \\ f_0 & & \vdots & & \ddots & \vdots \\ & \ddots & & & & g_0 \\ & & f_0 & & & \end{pmatrix}$$

The Resultant

Fact: $\gcd(f, g) = 1 \iff \operatorname{res}(f, g) \neq 0$

Fact: if $f(x) = \prod_{i=1}^{n} (x - \alpha_i)$, $\alpha_1, \ldots, \alpha_n \in \mathbb{C}$

then $\operatorname{res}(f, g) = \prod_{i=1}^{n} g(\alpha_i)$

Theorem: Given coefficients of $f(x)$ and $g(x)$ can compute $\operatorname{res}(f, g)$ in $\operatorname{poly}(n, m)$ size, $O(1)$ depth

SQUAREFREE GCD

let $f = \prod_{i=1}^{n} (x - \alpha_i) \prod_{i=1}^{d} (x - \gamma_i)$

$g = \prod_{i=1}^{m} (x - \beta_i) \prod_{i=1}^{d} (x - \gamma_i)$

squarefree: all $\alpha_i, \beta_i, \gamma_i \in \mathbb{C}$ are distinct

want $\prod_{i=1}^{d} (x - \gamma_i)$

SQUAREFREE GCD

Define $h(x, z) := (z - x) \cdot g(x)$

$$\lambda_1 := h(\alpha_1, z) = (z - \alpha_1) \cdot g(\alpha_1) \neq 0$$
$$\vdots$$
$$\lambda_n := h(\alpha_n, z) = (z - \alpha_n) \cdot g(\alpha_n) \neq 0$$
$$\lambda_{n+1} := h(\gamma_1, z) = (z - \gamma_1) \cdot g(\gamma_1) = 0$$
$$\vdots$$
$$\lambda_{n+d} := h(\gamma_d, z) = (z - \gamma_d) \cdot g(\gamma_d) = 0$$

SQUAREFREE GCD

Observe

$$e_k(\lambda_1, \dots, \lambda_{n+d}) = \sum_{\substack{S \subseteq [n] \\ T \subseteq [d] \\ |S|+|T|=k}} \prod_{i \in S} \lambda_i \prod_{i \in T} \lambda_{n+i}$$

$$= \sum_{\substack{S \subseteq [n] \\ |S|=k}} \prod_{i \in S} \lambda_i$$

$$= e_k(\lambda_1, \dots, \lambda_n)$$

$h(x,z) := (z-x) \cdot g(x)$

$\lambda_1 := (z-\alpha_1) \cdot g(\alpha_1) \neq 0$

$\vdots$

$\lambda_n := (z-\alpha_n) \cdot g(\alpha_n) \neq 0$

$\lambda_{n+1} := (z-\gamma_1) \cdot g(\gamma_1) = 0$

$\vdots$

$\lambda_{n+d} := (z-\gamma_d) \cdot g(\gamma_d) = 0$

SQUAREFREE GCD

Observe $e_k(\lambda_1, \dots, \lambda_{n+d}) = e_k(\lambda_1, \dots, \lambda_n)$

$$e_n(\lambda_1, \dots, \lambda_n) = \prod_{i=1}^{n} (z - \alpha_i)\, g(\alpha_i)$$

Normalize lead coeff to 1 $\rightsquigarrow$ $\prod_{i=1}^{n} (z - \alpha_i)$

$$= \frac{f}{\gcd(f, g)}$$

$$h(x, z) := (z - x) \cdot g(x)$$

$$\begin{aligned}
\lambda_1 &:= (z - \alpha_1) \cdot g(\alpha_1) \neq 0 \\
&\vdots \\
\lambda_n &:= (z - \alpha_n) \cdot g(\alpha_n) \neq 0 \\
\lambda_{n+1} &:= (z - \gamma_1) \cdot g(\gamma_1) = 0 \\
&\vdots \\
\lambda_{n+d} &:= (z - \gamma_d) \cdot g(\gamma_d) = 0
\end{aligned}$$

SQUAREFREE GCD

Theorem Given input $f, g \in \mathbb{C}[x]$ where f & g are squarefree, can compute $\gcd(f,g)$ in $\text{poly}(n,m)$ size and $O(1)$ depth.

Proof

(1) Compute $e_k(\lambda_1, \dots, \lambda_{n+d})$

(2) Compute $n := \max\{k : e_k(\bar{\lambda}) \neq 0\}$

(3) Normalize $e_n(\bar{\lambda})$ to have leading coeff 1

(4) Output $\dfrac{f(z)}{e_n(\bar{\lambda})}$ $\square$

$$h(x,z) := (z-x)\cdot g(x)$$

$$\begin{aligned}
\lambda_1 &:= (z-\alpha_1)\cdot g(\alpha_1) \neq 0 \\
&\vdots \\
\lambda_n &:= (z-\alpha_n)\cdot g(\alpha_n) \neq 0 \\
\lambda_{n+1} &:= (z-\gamma_1)\cdot g(\gamma_1) = 0 \\
&\vdots \\
\lambda_{n+d} &:= (z-\gamma_d)\cdot g(\gamma_d) = 0
\end{aligned}$$

FILTERING

Even when f and g are not squarefree, the preceding algorithm is useful

Theorem Given input $f, g \in \mathbb{C}[x]$ where f factors as $f(x) = \prod_i (x - \alpha_i)^{a_i}$, can compute

$$f_{g=0}(x) = \prod_{i : g(\alpha_i)=0} (x - \alpha_i)^{a_i}$$

and

$$f_{g\neq 0}(x) = \prod_{i : g(\alpha_i)\neq 0} (x - \alpha_i)^{a_i}$$

in polynomial size and $O(1)$ depth

Bézout Coefficients

Theorem Given input $f, g \in \mathbb{C}[x]$ where $\gcd(f, g) = 1$, can compute $s, t \in \mathbb{C}[x]$ such that

- $s(x) f(x) + t(x) g(x) = 1$
- $\deg(s) < \deg(g)$
- $\deg(t) < \deg(f)$

in $\text{poly}(n, m)$ size and $O(1)$ depth

Idea: If $g(x) = \prod_{i=1}^{m} (x - \beta_i)$, then $s(x)$ is determined by $s(\beta_i) = 1 / f(\beta_i)$, $\forall i \in \{1, \ldots, m\}$

Bézout Coefficients

Proof Lagrange interpolation. (Assume β_i are distinct)

$$s(x) = \sum_{i=1}^{m} \frac{1}{f(\beta_i)} \prod_{j \neq i} \frac{x - \beta_j}{\beta_i - \beta_j}$$

$$= \sum_{i=1}^{m} \frac{1}{f(\beta_i)\, g'(\beta_i)} \prod_{j \neq i} (x - \beta_j)$$

$$\operatorname{res}(f,g) \cdot \operatorname{res}(g,g') \cdot s(x) = \sum_{i=1}^{m} \prod_{j \neq i} f(\beta_j)\, g'(\beta_j)\, (x - \beta_j)$$

Compute ↑, divide by $\operatorname{res}(f,g) \cdot \operatorname{res}(g,g')$ □

Open Questions

1) Circuits of size $n^{O(1)}$ and depth $o(\log^2 n)$ for Extended Euclidean Algorithm? Subresultants?

(A. & Wigderson 202X): Depth $O(1)$ requires size $n^{\omega(1)}$

2) Which symmetric polynomials can be computed in low depth?

If $f(\vec{x}) = g(e_1(\vec{x}), \ldots, e_n(\vec{x}))$ and f has a small $\Sigma\Pi\Sigma$ circuit, does g have a small formula?

THANK YOU!