Arithmetic Circuit Complexity of Division and Truncation

Pranjal Dutta (Chennai Mathematical Institute & IIT Kanpur)
Gorav Jindal (Technische Universität Berlin, Berlin)
Anurag Pandey (Saarland University, Saarbrücken)
Amit Sinhababu (Aalen University, Germany)

Computational Complexity Conference (CCC) 2021

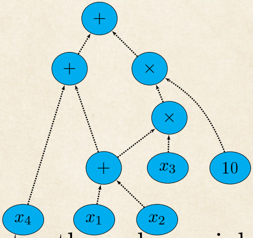
June 24, 2021

Motivation

- Suppose we can "compute" polynomials g, h efficiently.
- If h divides g, can we also "compute" $f \stackrel{\text{def}}{=} \frac{g}{h}$ efficiently?
- What do "compute" and "efficiently" mean here?

Polynomials and Arithmetic Circuits

- Every arithmetic circuit computes a polynomial and vice versa.



- Above circuit computes the polynomial $f \in \mathbb{C}[x_1, x_2, x_3, x_4]$ where $f = 10x_3(x_1 + x_2) + x_1 + x_2 + x_4$.
  - Size and depth have same definitions as in the Boolean case.

Arithmetic Complexity

Definition

The arithmetic complexity $L(f)$ of a polynomial $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ is defined as the minimum size of any arithmetic circuit computing f.

- Thus $L(f) \leq 10$, where $f = 10x_3(x_1 + x_2) + x_1 + x_2 + x_4$.

Permanent vs. Determinant

- It is not hard to show that $L(\det_n) = \mathrm{poly}(n)$ where

$$\det{}_n = \sum_{\sigma \in S_n} \mathrm{sign}(\sigma) \prod_{i \in [n]} x_{i,\sigma(i)}$$

  is the famous determinant polynomial.
- Define the permanent polynomial:

$$\mathrm{per}_n \stackrel{\text{def}}{=\!=} \sum_{\sigma \in S_n} \prod_{i \in [n]} x_{i,\sigma(i)}$$

Conjecture (Valiant)

$L(\mathrm{per}_n)$ *is super-polynomial in* n.

Divisions in Arithmetic Circuits

- We only used $\{+, \times\}$ gates in the arithmetic circuits above.
  - What if we also used divisions?

Lemma (Folklore)

If f can be computed by a size s circuit using $\{+, \times, \div\}$ gates then there exist g, h with $L(g), L(h) \leq 6s$ such that $f = \frac{g}{h}$.

Division Elimination

Problem (1)(Kaltofen 87)

If a polynomial can be computed by an arithmetic circuit (with division) of size s, can it be computed by a division-free arithmetic circuit of size poly(s)?

Problem (2)

If $L(g), L(h) \leq s$ and h divides g then is it true that $L(\frac{g}{h}) \leq \text{poly}(s)$?

- Problem (1) $\Longleftrightarrow$ Problem (2).

Known Results

Theorem (Strassen 73)

If f can be computed by an arithmetic circuit (with division) of size s, then $L(f) \leq \mathrm{poly}(s, \deg(f))$.

Example

If $g = x^{2^s} - 1$ and $h = x - 1$ then Strassen's result implies the upper bound $L(f) \leq 2^{O(s)}$.

Main Result

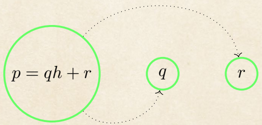
Theorem (Main Theorem)

If $L(g) \leq s_1, L(h) \leq s_2$ and h divides g then $L(\frac{g}{h}) \leq O((s_1 + s_2)d_h^2)$, where $d_h = \deg(h)$.

- Essentially, it is "easy" to divide by low degree polynomials.
- It is an exponential improvement over Strassen's result when $\deg(h)$ is $\text{poly}(s_1)$ and $\deg(f)$ is $\exp(s_2)$.

Proof Technique

- First consider the simpler case when g, h are uni-variate.
- C is a circuit of size $L(g)$ computing g.
- We split every gate in C into two gates as:
  - First gate computes quotient modulo h and other remainder.



Addition Gate

- Suppose $p = p_1 + p_2$ (in C) with $p_1 = q_1 h + r_1, p_2 = q_2 h + r_2$.
- Then:

$$
\begin{aligned}
p \bmod h &= r_1 + r_2 \\
p \operatorname{div} h &= q_1 + q_2
\end{aligned}
$$

Multiplication Gate

- Suppose $p = p_1 \times p_2$ (in C) with $p_1 = q_1 h + r_1, p_2 = q_2 h + r_2$.
- Then:

$$p \bmod h = r_1 r_2 \bmod h$$
$$p \text{ div } h = q_1 q_2 h + q_1 r_2 + q_2 r_1 + (r_1 r_2 \text{ div } h)$$

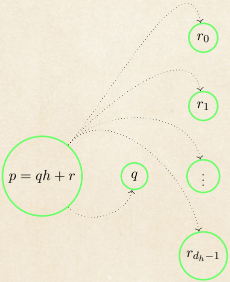
- Need to only compute $(r_1 r_2 \bmod h)$ and $(r_1 r_2 \text{ div } h)$.
  - Easy to compute since they are polynomials of degree at most $\deg(h) - 1$ (naively).

Multivariate Case

- Assume h to be monic in x_n.
  - Achievable by an invertible linear transformation of variables.
  - Thus $\bmod h$ and $\operatorname{div} h$ are defined (w.r.t x_n).
- C is a circuit of size $L(g)$ computing g.

Multivariate Case

- We split every gate T in C to $d_h + 1$ ($d_h \stackrel{\text{def}}{=\!=} \deg(h)$) many gates.
  - T computes the polynomial $p = qh + r$.
  - $r = r_0 + r_1 x_n + \cdots + r_{d_h-1} x_n^{d_h-1}$ with $r_i \in \mathbb{C}[x_1, x_2, \ldots, x_{n-1}]$.
  - First d_h gates compute $r_0, r_1, \ldots, r_{d_h-1}$.
  - Last gate computes q.



Addition Gate

- Suppose $p = p_1 + p_2$ (in C) with $p_1 = q_1 h + a, p_2 = q_2 h + b$

$$a = a_0 + a_1 x_n + \cdots + a_{d_h - 1} x_n^{d_h - 1}$$
$$b = b_0 + b_1 x_n + \cdots + b_{d_h - 1} x_n^{d_h - 1}$$

- Then:

$$r_i \overset{\text{def}}{=\!=} a_i + b_i$$
$$p \bmod h = r_0 + r_1 x_n + \cdots + r_{d_h - 1} x_n^{d_h - 1}$$
$$p \operatorname{div} h = q_1 + q_2$$

Multiplication Gate

- Suppose $p = p_1 \times p_2$ (in C) with $p_1 = q_1 h + a, p_2 = q_2 h + b$.

$$a = a_0 + a_1 x_n + \cdots + a_{d_h - 1} x_n^{d_h - 1}$$
$$b = b_0 + b_1 x_n + \cdots + b_{d_h - 1} x_n^{d_h - 1}$$

- Then:

$$p \bmod h = ab \bmod h$$
$$p \text{ div } h = q_1 q_2 h + q_1 r_2 + q_2 r_1 + (ab \text{ div } h)$$

- By using the polynomial long division, we can efficiently compute:
  - ab div h
  - Coefficients of $ab \bmod h$

Power Series

- $A = \sum_{i \geq 0} A_i x^i$ is a power series in the power series ring $\mathbb{C}[[x]]$.
- Degree d truncation $\mathrm{trunc}(A, d)$ of A is: $\mathrm{trunc}(A, d) \stackrel{\text{def}}{=\!=} \sum_{0 \leq i \leq d} A_i x^i$.
- A uni-variate polynomial family $(f_d)_{d \in \mathbb{N}}$ ($\deg(f_d) = d$) is "easy" if $L(f_d) = \mathrm{poly}(\log d)$.
  - Otherwise we call it "hard".
- Study the complexity of polynomial families obtained by truncation of power series.

Rational Functions

Theorem

If g, h are constant degree polynomials and $f = \frac{g}{h} \in \mathbb{C}[[x]]$ is a power series then the polynomial family $(\mathrm{trunc}(f, d))_{d \in \mathbb{N}}$ is easy.

- This theorem also holds for some cases where g, h have non-constant degree.

Upper Bound Idea

Lemma (Partial fraction decomposition)

If $\frac{g}{h} \in \mathbb{C}[[x]]$ *is a rational function with* $\deg(g) < \deg(h)$ *and* $h(x) = \prod_{i \in [k]} (x - a_i)^{d_i}$ *then:*

$$\frac{g}{h} = \sum_{i \in [k]} \sum_{j \in [d_i]} \frac{b_{ij}}{(x - a_i)^j}. \qquad \text{(for some } b_{ij} \in \mathbb{C}\text{)}$$

- $(\mathrm{trunc}(1/(x-a), d))_{d \in \mathbb{N}}$ is easy
- By computing higher order derivatives, $(\mathrm{trunc}(1/(x-a)^j, d))_{d \in \mathbb{N}}$ is also easy.

Constant Free Complexity

Definition

For any polynomial f, $\tau(f)$ is the size of the minimal constant-free circuit computing f. Only constants allowed are $\{-1, 0, 1\}$.

- This definition also extends to computation of integers.
- A sequence $(a_n)_{n \in \mathbb{N}}$ of integers is "easy" to compute if $\tau(a_n) \leq \mathrm{poly}(\log n)$.

Known Results

Lemma (Folklore)

If $(n!)_{n\in\mathbb{N}}$ is easy then integer factorization can be performed in polynomial time.

- (Andrews 2020)$\Longrightarrow$ $\left(\binom{2n}{n}\right)_{n\in\mathbb{N}}$ is easy then so is $(n!)_{n\in\mathbb{N}}$.
- Easiness of truncation of $\sqrt{1+4x}$ implies easiness of $\left(\binom{2n}{n}\right)_{n\in\mathbb{N}}$.
  - Thus polynomial time algorithms for integer factorization.

Generalizing Hardness of $\sqrt{1+4x}$

Theorem

For any constant k, if $\tau(\mathrm{trunc}((1+k^2x)^{\frac{i}{k}}, d)) = \mathrm{poly}(\log d)$ (for all $i \in [k-1]$) then integer factorization can be performed in polynomial time (in the non-uniform setting).

- The case $k = 2$ follows from (Andrews 2020).

Hardness Idea

- Easiness of $\operatorname{trunc}((1+k^2x)^{\frac{i}{k}}, d)$ (for all $i \in [k-1]$) implies the easiness of:

$$N(n,k) \stackrel{\text{def}}{=\!=} \frac{k^{(k-2)d}(nk)!}{(n!)^k}$$

- By a variant of binary search, easiness of $N(n,k)$ implies efficient integer factorization.
- We do not know if easiness of $\operatorname{trunc}((1+k^2x)^{\frac{i}{k}}, d)$ implies easiness of $(n!)_{n\in\mathbb{N}}$.

Stern Sequence (Easy)

Definition

Sequence $(a_n)_{n\in\mathbb{N}}$ given by
$a_0 = 0, a_1 = 1, a_{2n} = a_n, a_{2n+1} = a_n + a_{n+1}$, is the Stern sequence.

Lemma

The generating function $A(x) \overset{def}{=\!=} \sum a_n x^n$ *of the Stern sequence is transcendental.*

Theorem

$L(\mathrm{trunc}(A(x), d)) = O(\log^2 d)$.

Hard Transcendental Power Series

Lemma

The power series $F(x) \overset{def}{=\!=} \sum_{n \geq 0} n! x^n$ *is transcendental.*

- If truncation of $F(x)$ is easy to compute then $(n!)_{n \in \mathbb{N}}$ is easy to compute.
- So truncation of $F(x)$ is likely to be hard.

Conclusion

- Can divide by low degree polynomials efficiently.
- Our division complexity upper bound also holds for the border complexity.
- Truncation of general algebraic functions is likely to be hard
  - Truncation of rational functions is easy
- We also show some examples of Transcendental power series:
  - Whose truncation is easy
  - Whose truncation is conditionally hard

Thanks

Thanks for your attention ☺