CS598maf Algebraic and Geometric Complexity Theory

Lecture 6 (2023-02-09)

logistics: - pset 1 due
- pset 2 out tonight

last lecture: - polynomial families
- complexity classes
- reductions, completeness
- IMM, det are VBP-complete
  VQP-complete

Q: primitive ops unit cost?
$+, \times, \div$ over $\mathbb{F}$

Q: primitive ops on [homog] arborescences?
- $\mathbb{F}[x] \Rightarrow f, g$ deg d
  - addition: $O(d)$ time
  - multiplication:
    - $O(d^2)$ time
      ↳ used in homog. ckts

$f(\bar{y}) = \sum_i H_i(\bar{y})$

$f(x \cdot \bar{y}) = \sum_i H_i(\bar{y}) \cdot x^i \in \mathbb{F}[\bar{y}][x]$

↖ coord

$\Rightarrow$ naive algo for

$f(x \cdot \bar{y}) \cdot g(x \cdot \bar{y})$

is [homog] multiplication

in $\mathbb{F}[\bar{y}][x]$

- $O(d^{\log_2 3} < d^{1.58\ldots})$ [Karatsuba]

sketch: multiply $a_0 + a_1 y$, $b_0 + b_1 y$
- naively like 4 mult
- **fact:** can be 3 [3] mult's

decompose $f = a_0 + a_1 \cdot y$ w/ a_0, a_1
$g = b_0 + b_1 y$ b_0, b_1

deg $< x^{d/2}$

$y = x^{d/2}$

mult f, g via recursively

$T(d) \le 3 \cdot T(d/2) + O(d)$

$\le \cdots \le O(d^{\log_2 3})$

- $O(d \lg d)$ if $\mathbb{F}$ "supports" the fast fourier transform
  - has circuit complexity

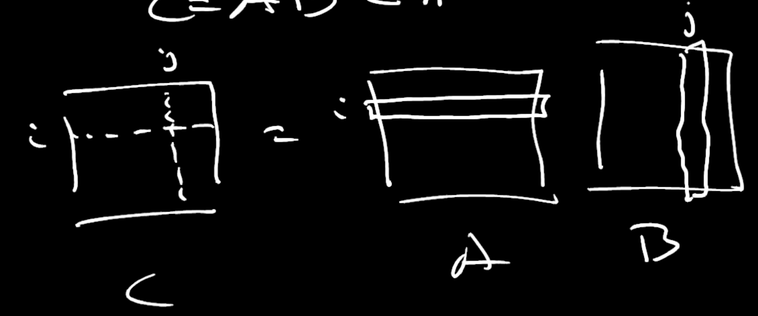
remark:
- not useful for homogeneous circuits
  - as FFT is not homogeneous
  - but can use if only computing homog parts (of homog)
- $O(d \cdot \lg d \cdot \lg\lg d)$ for any $\mathbb{F}$
- closely related to multiply in $\mathbb{Z}$

- $\mathbb{F}^{n \times n} \ni A, B$
  - addition: $O(n^2)$
  - multiplication:

prop: Matrix mult in $O(n^3)$ time

pf: $A, B \in \mathbb{F}^{n \times n}$

$C = AB \in \mathbb{F}^{n \times n}$



$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$

correctness: by def

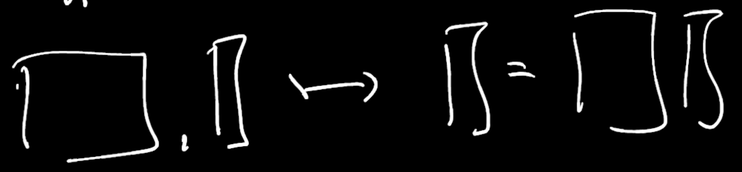
complexity: n^2 entries
per entry: $O(n)$ mult
$O(n)$ add

□

Q: do better?

A: sometimes naive algo is optimal

fact: M ($\mathbb{F}^{n\times n}$), v ($\mathbb{F}^n$) $\mapsto Mv$ requires $\Omega(n^2)$ ops



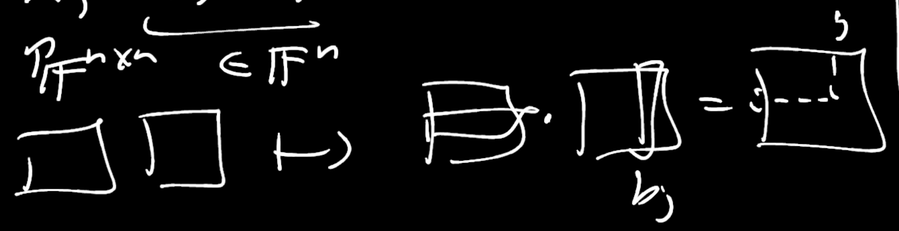
fact: $v \mapsto Mv$ requires $\Omega(n^2)$ ops for (some) M

A: solving disjoint problems on disjoint inputs "should be" independent

law: $A, B \mapsto A \cdot B$ ($\mathbb{F}^{n\times n}$, $\mathbb{F}^{n\times n}$)

is $A, b_1, \dots, b_n \mapsto Ab_1, \dots, Ab_n$ ($\mathbb{F}^{n\times n}$, $\in \mathbb{F}^n$)

sketch:



"=" matrix mult should take

$\Omega(n^2) \cdot n = \Omega(n^3)$ ops

A: - multiplication over $\mathbb{F}[x]$ has nonlinear algo, so matrix mult does as well

---

nearly linear time

---

fact: $n\times n$ matrix mult is in $O(n^{2.3729\dots})$ time

rmk: widely used algorithmic tool

- algebraic algo - det
  - solving linear systems of eqn $Ax=b$
  - representation thy
- bodies problems
  - bipartite matching
  - triangle detection
  - parsing context-free grammars

thm [Strassen 69] $n \times n$ matrix mult in $O(n^{\log_2 7} < n^{2.81\ldots})$ time

ref. idea: recursion

conserve (multiplications), possibly increase additions

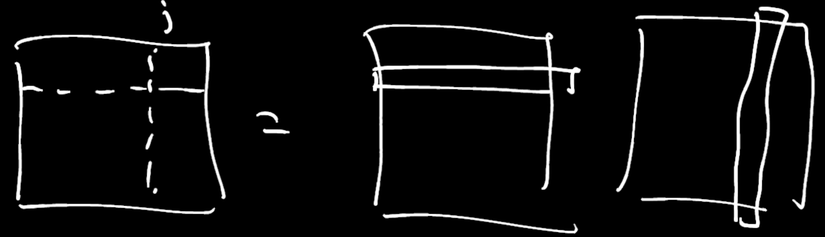
lem [block multiplication]:

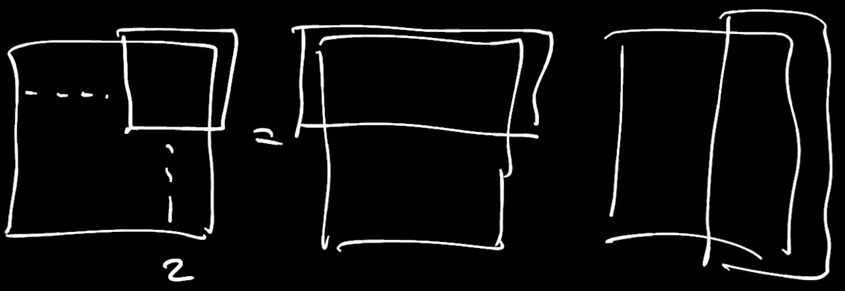
$$A = \left[\begin{array}{c|c} A^{1,1} & A^{1,2} \\ \hline A^{2,1} & A^{2,2} \end{array}\right] \qquad B = \left[\begin{array}{c|c} B^{1,1} & B^{1,2} \\ \hline B^{2,1} & B^{2,2} \end{array}\right]$$

$\rightsquigarrow$ $A^{i,j}, B^{i,j}$ $m \times m$, $n = 2m$

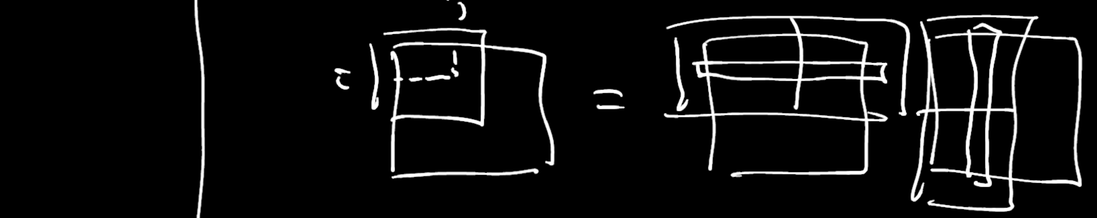
$\Rightarrow C = AB$ has $C = \left[\begin{array}{c|c} C^{1,1} & C^{1,2} \\ \hline C^{2,1} & C^{2,2} \end{array}\right]$ $\rightsquigarrow$ $C^{i,j}$ $m \times m$

$$C^{i,j} = \sum_{k=1}^{2} A^{i,k} B^{k,j}$$





Strassen: for $i, j \in m$



$$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

$$= \sum_{k=1}^{m} A_{ik} B_{kj} + \sum_{k=1}^{m} A_{i,m+k} \cdot B_{m+k,j}$$

$(A^{1,1})_{ik}$ $(B^{1,1})_{kj}$ $(A^{1,2})_{ik}$ $(B^{2,1})_{kj}$

$(A^{1,1}B^{1,1})_{ij}$ $(A^{1,2}B^{2,1})_{ij}$

$$(C^{1,1})_{ij} = (A^{1,1}B^{1,1} + A^{1,2}B^{2,1})_{ij}$$

$$C^{1,1} = A^{1,1}B^{1,1} + A^{1,2}B^{2,1}$$

rmk: - bm ≡ multiplying matrices w/ matrix entries is a (larger) matrix multiplication

⇒ recursion

- matrix multiplication does not exploit commutativity of matrix entries

thm: 2×2 matrix multiplication can be performed using
- 7 multiplications
- 18 additions / subtractions

Strassen: $\begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$

also: compute
$p_1 = (a_{11}+a_{22})(b_{11}+b_{22})$
$p_2 = (a_{21}+a_{22})\cdot b_{11}$
$p_3 = a_{11}(b_{12}-b_{22})$
$p_4 = a_{22}(-b_{11}+b_{21})$
$p_5 = (a_{11}+a_{12})\,b_{22}$
$p_6 = (-a_{11}+a_{21})(b_{11}+b_{12})$
$p_7 = (a_{12}-a_{22})(b_{21}+b_{22})$

output
$c_{11} = p_1+p_4-p_5+p_7$
$c_{12} = p_3+p_5$
$c_{21} = p_2+p_4$
$c_{22} = p_1+p_3-p_2+p_6$

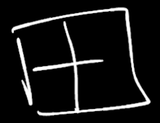
complexity: 7 multiplications
18 add/subtract

correctness: (next) "clear"

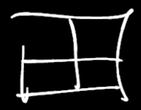
rmk: naive also - 4 entries of C
- per entry
- 2 mult
- 1 add
⇒ 8 mult, 4 add
(>7, <18)

- discuss of "disjoint problems are independent"

algo: for A, B size $N = 2^k$

$A =$  $B =$ 

blocks size $2^{k-1} = N/2$

compute $C =$  via – 7 mults
– 18 additions

recursively

naively

correctness: clear

complexity: $T(N) \leq 7 \cdot T(N/2) + O(N^2)$
$\leq \cdots$
$\leq O(N^{\log_2 7})$

for general N, embed A, B into matrices size $2^{\lceil \log N \rceil}$ via zero padding

$\boxed{A} \mapsto \begin{array}{|c|c|} \hline A & \\ \hline 0 & 0 \\ \hline \end{array}$

F2

rmk: disjointness "independent problems are independent"

today: - complexity of primitive ops
- matrix mult
  - naive
  - block
  - Strassen

next lecture: bilinear complexity

logistics: pset 1 due
pset 2