Permanent and Determinant

non-identical twins

Avi Wigderson
IAS, Princeton

Meet the twins

F field, char(F)$\neq$2.

$X \in M_n(F)$ matrix of variables X_{ij}

$$Det_n(X) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_{i \in [n]} X_{i\sigma(i)}$$

$$Per_n(X) = \sum_{\sigma \in S_n} \prod_{i \in [n]} X_{i\sigma(i)}$$

Homogeneous, multi-linear, degree n polynomials on n^2 variables, with $0, \pm 1$ coefficients.

Meet the twins

	$Det_n(X)$	$Per_n(X)$
	$\sum_{\sigma\in S_n} sgn(\sigma) \prod_{i\in[n]} X_{i\sigma(i)}$	$\sum_{\sigma\in S_n} \prod_{i\in[n]} X_{i\sigma(i)}$
Physics:	Fermions	Bosons
Knots:	Alexander polynomial	Jones polynomial
	Linear Algebra	Enumeration /Counting
Uses:	Geometry / Volume	Statistical Mechanics
	Everywhere	Comput. Complexity
Counting:	Spanning trees	Matchings
	Planar matchings	Everything
Complexity:	Easy	Hard (?)
Boolean:	NC-complete	#P-complete
Arithmetic:	VP-complete	VNP-complete

Complexity classes

Permanent

Hard NP Efficient proof/verification

Easy P Efficient computation

Determinant

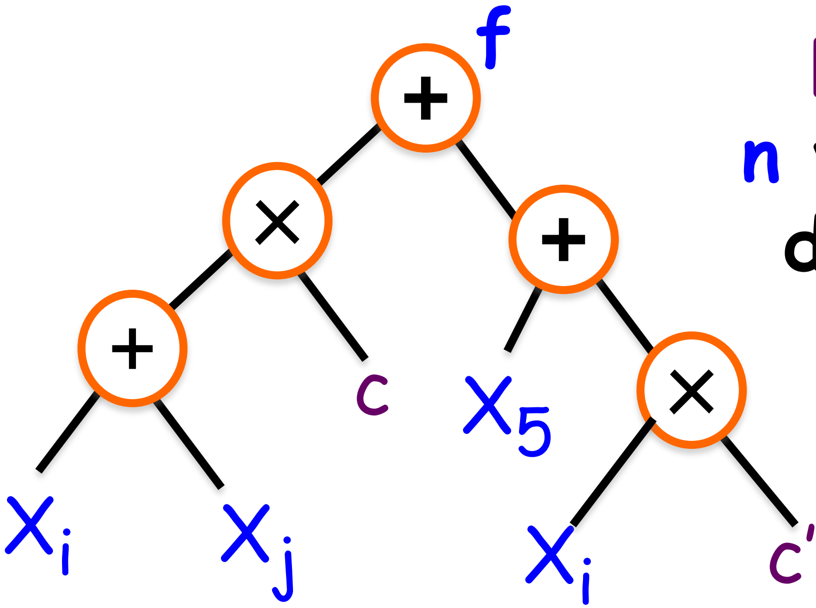
Completeness [Valiant]

Arithmetic			Boolean	
			EXP	Exponential time
			PSPACE	Polynomial space
VNP	Permenent [Toda]		#P [Feynman]	Counting
			BQP	efficient quantum computation
			PH	Bounded alternation
		Hard	NP	Efficient proof/verification
		Easy	P	Efficient computation
VP	Determinant		NC	Fast parallel computation
			L	Logarithmic space

Arithmetic Computation

Computing formal polynomials

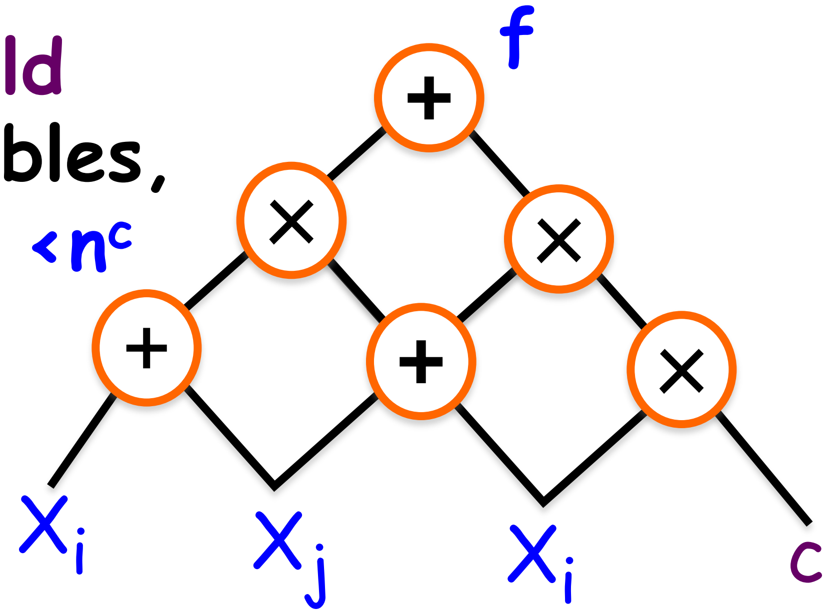
Arithmetic complexity - basics



F field

n variables,

deg $f < n^c$



Formula

$L(f)$ – formula size

Circuit

$S(f)$ – Circuit size

Thm[VSBR]: $S(f) \leq L(f) \leq S(f)^{\log n}$

Complexity of Det

Thm[Strassen]: $S(Det_n) \le n^3$ (no division!)

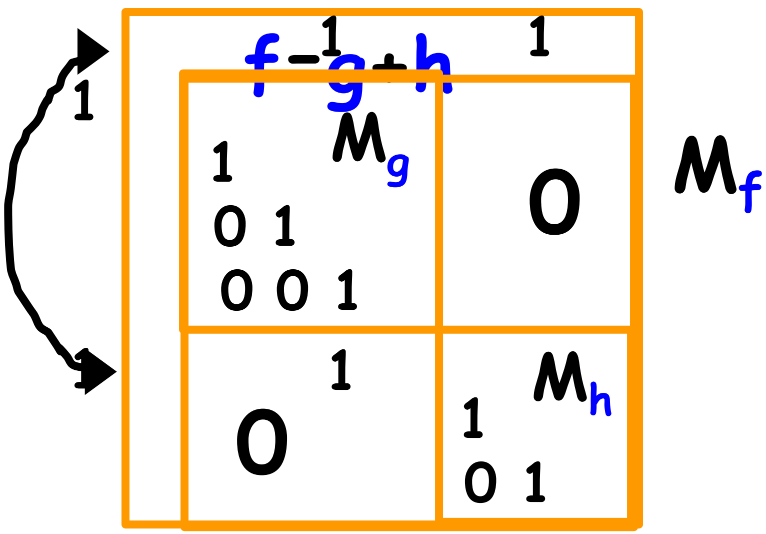
Thm[Csansky]: $L(Det_n) \le n^{\log n}$ (OPEN: poly?)

Thm[Valiant]: If $L(f)=s$, then there is a $2s \times 2s$ matrix M_f of vars and constants, $f=\det M_f$

Determinantal representations of polynomials

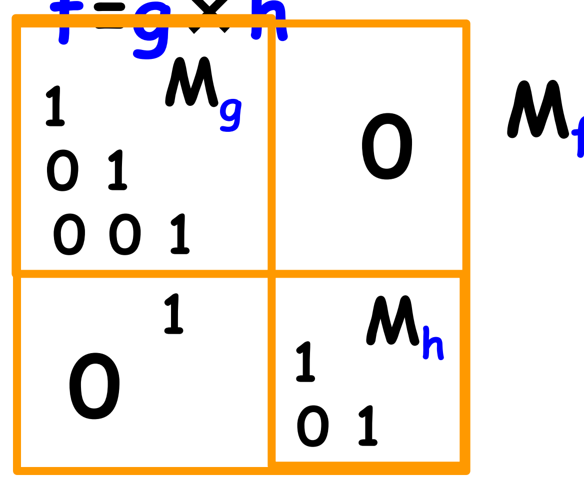
Proof: Induction

$f=g+h$

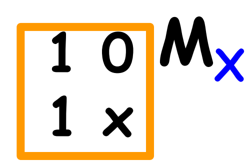


$|M_f|=|M_g|+|M_h|$

$f=g\times h$



$|M_f|=|M_g|\times|M_h|$



VNP completeness of Per

Def[Valiant]:
An integer polynomial $f \in Z[X_1, \ldots X_n]$ is in VNP if each coefficient is efficiently computable.

Intuitively, VNP captures all explicit polynomials!

Thm[Valiant]: If $f \in$ VNP, then there is a poly size matrix M_f with f = Per M_f

Proof - much more sophisticated

Algebraic analog of "P≠NP"

Affine map L: $M_n(F) \to M_k(F)$ is good if $Per_n = Det_k \circ L$

k(n): the smallest k for which there is a good map?

[Polya] k(2) =2 $\quad Per_2 \begin{bmatrix} a & b \\ c & d \end{bmatrix} = Det_2 \begin{bmatrix} a & b \\ -c & d \end{bmatrix}$

[Valiant] $k(n) < \exp(n)$

[Mignon-Ressayre] $k(n) > n^2$

[Valiant] $k(n) \neq \mathrm{poly}(n) \Leftrightarrow$ VP≠VNP

[Mulmuley-Sohoni] Geometric Complexity Theory (GCT):

Per & Det are defined by their symmetries. Find, for k small, representation theoretic obstacles for good maps.

Arithmetic lower bounds for Det_n & Per_n

Thm[Nisan] Both require non-commutative size 2^n arithmetic formulae. Open: l.b. for Circuits?

Thm[Raz] Both require multi-linear arithmetic formulae of size $n^{\log n}$. Open: Exponential l.b.?

Thm[Gupta-Kamath-Kayal-Saptharishi]:

$\mathrm{size}_4(\mathrm{Det}_n) > n^{\sqrt{n}}$ Tight!!

$\mathrm{size}_4(\mathrm{Per}_n) > n^{\sqrt{n}}$ Improvement → $VP \neq VNP$

Nice properties of Per

&

Complexity theoretic consequences

Nice properties of Per (and Det)
(1) Downwards self-reducible

Permanent of $n \times n$ matrices efficiently computed from (several) permanents of *smaller* matrices.

Row expansion

$$\mathrm{Per}_n(X) = \sum_{i \in [n]} X_{1i}\, \mathrm{Per}_{n-1}(X^{1i})$$

Nice properties of Per (and Det)

(2) Random self-reducible/correctible

[Beaver-Feigenbaum, Lipton]

The permanent of nxn matrices can be computed from the permanent of several *random* matrices.

Assume $C(Z)=Per_n(Z)$ on $\leq 1/(8n)$ of $Z \in M_n(F)$

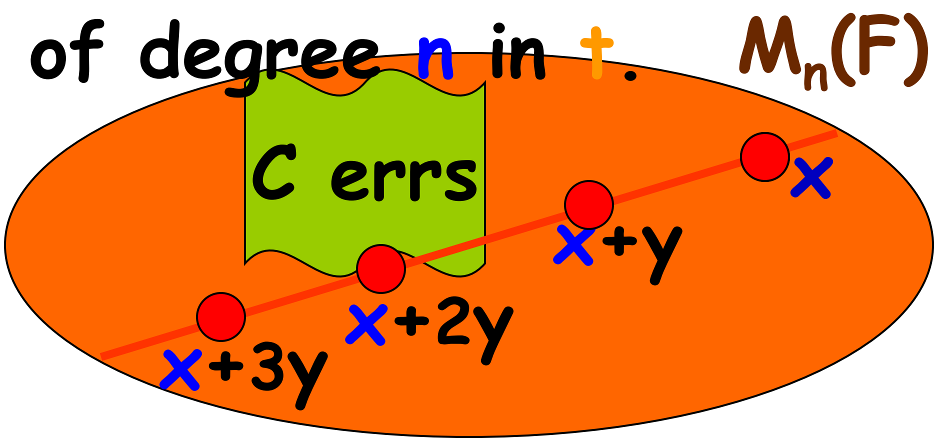
Interpolate $Per_n(X)$ on a random line: Y random,

let $g(t)=C(X+tY)$ – a poly of degree n in t.

Eval on $t=1,2,\ldots,n+1$.

WHP $g(t)=Per(X+tY)$,

so $g(0)=Per(X)$



Hardness amplification

If the Permanent can be efficiently computed for most inputs, then it can for all inputs !

If the Permanent is hard in the worst-case, then it is also hard on average

Worst-case → Average case reduction

Works for any low degree polynomial.

Arithmetization: Boolean functions→polynomials

Lower bounds, derandomization, prob. proofs

Avalanche of consequences to probabilistic proof systems

Using both RSR and DSR of Permanent!

[Nisan]	Per $\in$ 2IP
[Lund-Fortnow-Karloff-Nisan]	Per $\in$ IP
[Shamir]	IP = PSPACE
[Babai-Fortnow-Lund]	2IP = NEXP
[Arora-Safra, Arora-Lund-Motwani-Sudan-Szegedy]	PCP = NP

Efficient Verification

(skeptical, efficient) verifier

vs.

(untrusted, all powerful) Prover

NP - theorems with short *written* proofs

sound & complete

IP - theorems with fast *interactive* proofs

sound & complete WHP

Per $\in$ IP [LFKN]

How to check a theorem that has no short proof?

$Z_i \in M_i(F)$ $\quad a_i \in F$

Verifier	(untrusted) Prover
Q_n: what is Per(Z_n)?	A_n: Per(Z_n)= a_n
Q_{n-1}: what is Per(Z_{n-1})?	A_{n-1}: Per(Z_{n-1})=a_{n-1}
Q_{n-2}: what is Per(Z_{n-2})?	A_{n-2}: Per(Z_{n-2})=a_{n-2}
......	......
Q_2: what is Per(Z_2)?	A_2: Per(Z_2)=a_2
Q_1: what is Per(Z_1)?	A_1: Per(Z_1)=a_1

Claim: If A_i is correct, than A_{i+1} is correct whp!

Verifier can check Per(Z_1)=a_1 without help.

A twist on Random-self-reducibility

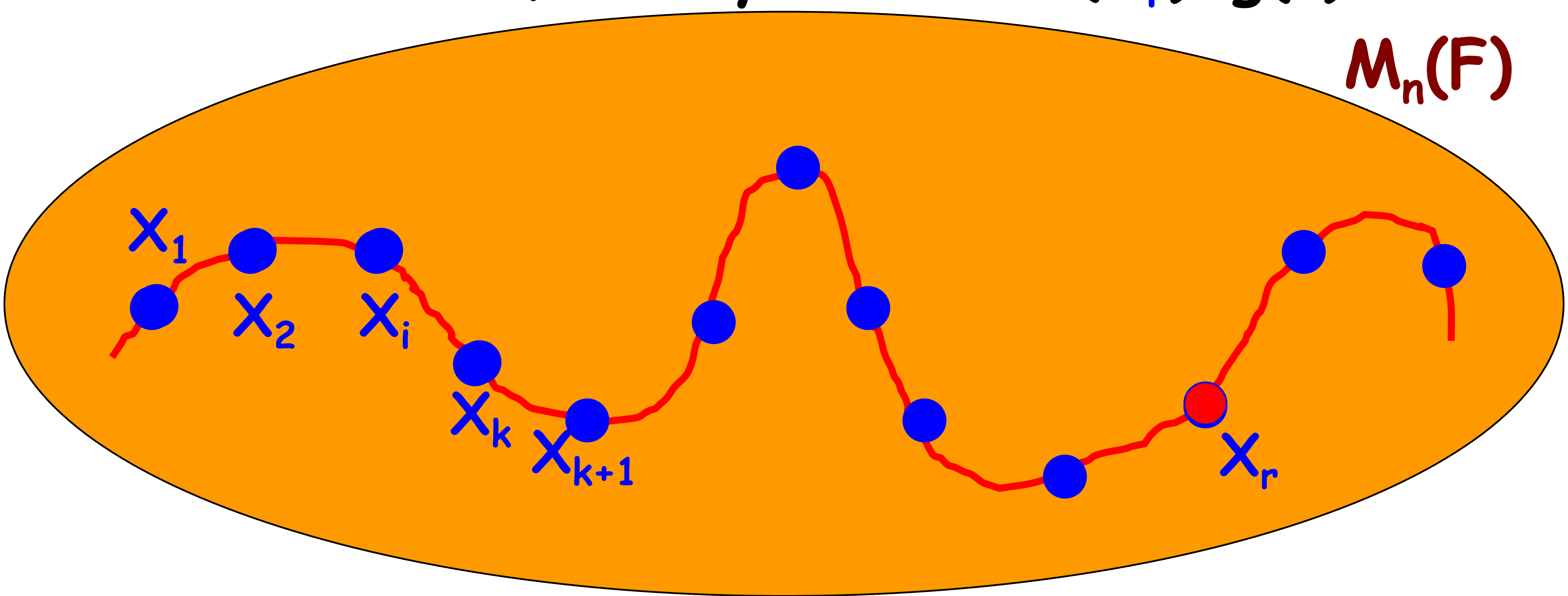
saw: *compute* one from many random inputs

now: *verify* many from one random input

Claims: $Per(X_1)=a_1,\dots,Per(X_k)=a_k,\quad X_1,\dots,X_k \in M_n(F)$

Pick random X_{k+1}, ask for $g(t)=Per(X_t)$, the unique deg k curve through $X_1,\dots,X_{k+1}$. Check for $[1,k]$

Pick random $r \in F$, verify claim $Per(X_r)=g(r)$



Boolean Computation

Evaluating functions

The class #P (and $P^{\#P}$)

All "natural" counting problems.

✔ # of sat assignments of a Boolean formula

✔# of cliques in a graph

✔# Hamilton cycles in a graph

Decision Problem NP-complete

✔# perfect matchings in a graph (Per) [Valiant]

✔# of linear extensions of a poset

-# of spanning trees of a graph (≤Det [Kirchoff])

in P

#P - # of accepting paths of an NP-machine.

#P-complete problems

Knot Theory   Graph Theory   Statistical Physics

✔ Evaluating Tutte, Jones, Chromatic,…polynomials

- # perfect matchings in *planar* gphs (≤Det [Kasteleyn])

Quantum Computation

BPP: Efficient probabilistic computation

$\cap$

BQP: Efficient quantum computation

Thm[Feynman, Bernstein-Vazirani] BQP $\subset$ $P^{\#P}$

Thm[Shor] Factoring $\in$ BQP (assumed not in BPP)

-Can quantum computers be built? What can they do?

Particles: Fermions (matter) Bosons (light, force)

Wave function: Determinant Permanent

[Valiant, Terhal-DiVincenzo, Knill]

Fermionic computers = holographic algs $\leq$ Determinant

[Aaronson-Arkhipov]

Bosonic computers can "sample" the Permanent

Approximating Permanents

of non-negative matrices

Approximating Per_n

[Valiant] Permanent of 0/1 matrices is #P-hard

[Jerrum-Sinclair-Vigoda] Efficient *probabilistic* algorithm for $(1+\varepsilon)$-approximation for the permanent of any non-negative real matrix.

Monte-Carlo Markov Chain

(Glauber Dynamics, Metropolis algs,…)

Such algs exist now for many #P-hard problems.

Important interaction area for CS, Math, Physics

Approx Per_n deterministically

A: $n \times n$ non-negative real matrix.

[Linial-Samorodnitsky-Wigderson]

Deterministic, efficient e^n -factor approximation.

Two ingredients:

(1) [Falikman,Egorichev] If B Doubly Stochastic

then $e^{-n} \approx n!/n^n \le Per(B) \le 1$

(the lower bound solved van der Vaerden's conj)

(2) Strongly polynomial algorithm for the following reduction to DS matrices:

Matrix scaling: Find diagonal X,Y s.t. XAY is DS

[Gurvits-Samorodnitsky'14] 2^n -factor approx.

OPEN: Find a deterministic subexp approx.

Thanks!