P, NP and Mathematics
A computational complexity perspective

Avi Wigderson
Institute for Advanced Study
http://www.math.ias.edu/~avi

Algebra: Polynomial Identities

Is $\det(V(x_1, x_2,\dots, x_n)) - \Pi_{i<k} (x_i - x_k) \equiv 0$?

Theorem [Vandermonde]: YES

Given (implicitly, e.g. as a formula) a polynomial p of degree d. Is $p(x_1, x_2,\dots, x_n) \equiv 0$?

Algorithm [Schwartz-Zippel] :

Pick r_i indep at random in $\{1,2,\dots,100d\}$

$p \equiv 0 \Rightarrow \Pr[\, p(r_1, r_2,\dots, r_n) = 0 \,] = 1$

$p \neq 0 \Rightarrow \Pr[\, p(r_1, r_2,\dots, r_n) \neq 0 \,] > .99$

Comments: *Over small finite fields it is coNP-complete*

[Kaltofen] *Over large finite fields one can even factor p*

Analysis: Fourier coefficients

Given (implicitely) a function $f:(Z_2)^n \rightarrow \{-1,1\}$ (e.g. as a formula), and $\varepsilon>0$,

Find all characters χ such that $|\langle f,\chi\rangle| \geq \varepsilon$

Comment : At most $1/\varepsilon^2$ such χ

Algorithm [Goldreich-Levin] :

…adaptive sampling… Pr[success] > .99

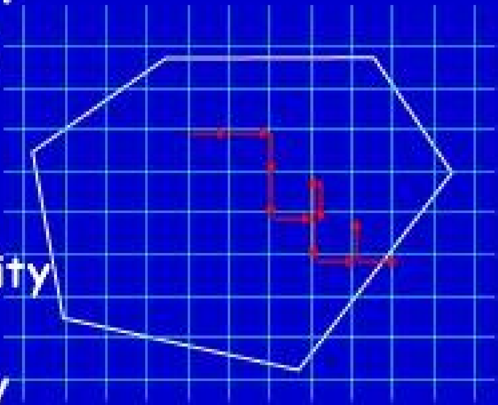
[AGS] : Extension to other Abelian groups.

Applications: Coding Theory, Complexity Theory

Geometry: Estimating Volumes

Given (implicitly) a convex body K in R^d (d large!)
(e.g. by a set of linear inequalities)
Estimate volume (K)
Comment: Computing volume(K) exactly is #P-complete

Algorithm [Dyer-Frieze-Kannan,...] :
Approx counting ≈ random sampling
Random walk inside K.
Rapidly mixing Markov chain.
Analysis:
Spectral gap ≈ isoperimetric inequality
Applications:
Statistical Mechanics, Group Theory



Fundhamental question #2

Does randomness help ?

Are there problems with probabilistic polytime algorithm but no deterministic one?

Conjecture 2: YES

Fundamental question #1

Does NP require exponential time/size ?

Conjecture 1: YES

Theorem: One of these conjectures is false!

Hardness vs. Randomness

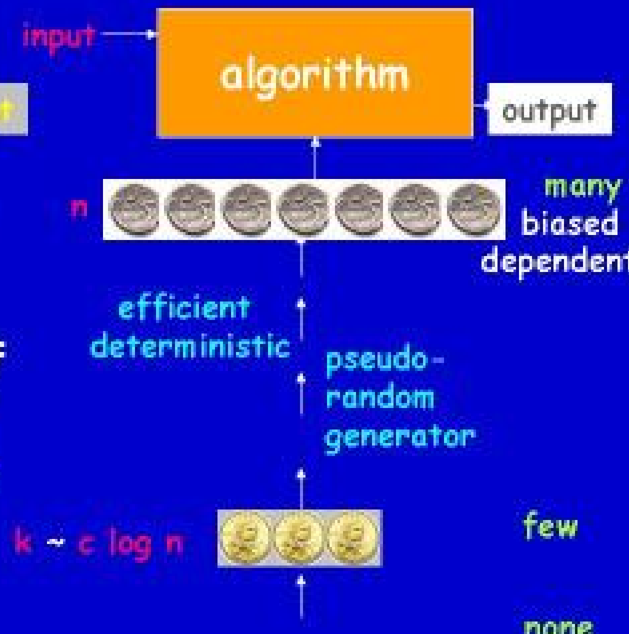
Theorem [Blum-Micali,Yao,Nisan-Wigderson, Impagliazzo-Wigderson,…] :

If there are natural hard problems
Then randomness can be efficiently eliminated.

Theorem [Impagliazzo-Wigderson]
NP requires exponential *size* ⇒
BPP=P (every probabilistic polynomial algorithm has a deterministic counterpart)

Theorem [IKW, Impagliazzo-Kabanets…] :
Partial converse! Derandomization ⇒ Hardness

Computational Pseudo-Randomness



 pseudorandom if
for every efficient
algorithm, for every input,

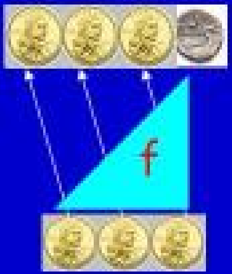
output ≈ output

Hardness $\Rightarrow$ Pseudorandomness

Need: $G: \{0,1\}^k \to :\{0,1\}^n$

NW generator

Show: $G: \{0,1\}^k \to :\{0,1\}^{k+1}$



k+1

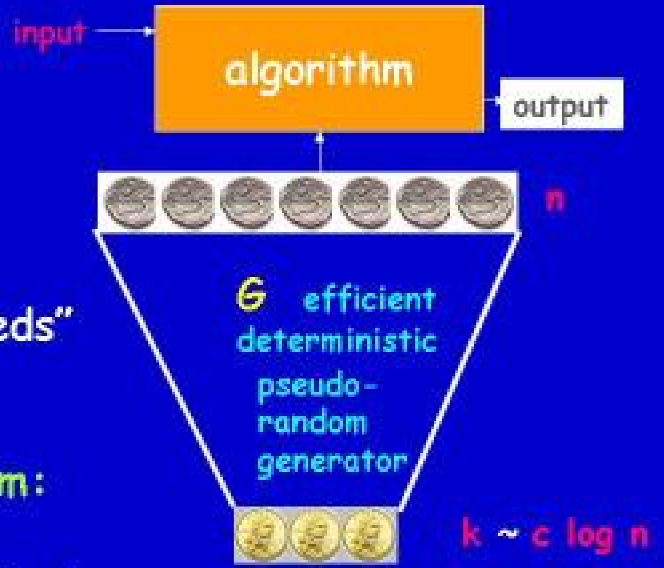
k ~ clog n

Need: Pr[$C(x) = f(x)$] < 1/2 + exp(-k) — Average-case hardness
for every computation C, size(C) < s

Hardness amplification

Have: Pr[$C'(x) = f'(x)$] < 1 — Worst-case hardness
for every computation C', size(C') < s'

Derandomization



Deterministic algorithm:

- Try all possible $2^k = n^c$ "seeds"
- Take majority vote

Pseudorandomness paradigm:

Can derandomize specific algorithms without assumptions!

e.g. Primality Testing & Maze exploration

The Power of Randomness

In other settings…

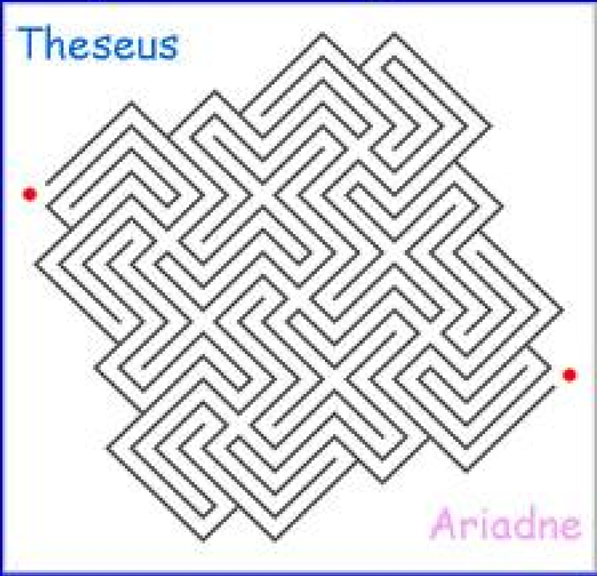
Getting out of mazes (when your memory is weak)

n-vertex maze/graph

Only a local view (logspace)

Theorem [Aleliunas-Karp-Lipton-Lovasz-Rackoff] :
A random walk will visit every vertex in n^2 steps (with probability >99%)

Theorem [Reingold] :
A deterministic walk, computable in logspace, will visit every vertex.
Uses ZigZag expanders [Reingold-Vadhan-Wigderson]



Crete, ~1000 BC

The Power and Weakness of Randomness (when you are short on time)

Avi Wigderson
Institute for Advanced Study

Probabilistic Proof System
[Goldwasser-Micali-Rackoff, Babai]

Is a mathematical statement claim true? E.g.

claim: "No integers x, y, z, n>2 satisfy $x^n + y^n = z^n$"

claim: "The Riemann Hypothesis has a 200 page proof"

Prover

probabilistic

An efficient Verifier V(claim, argument) satisfies:

*) If claim is true then V(claim, argument) = TRUE for some argument always (in which case claim=theorem, argument=proof)

**) If claim is false then V(claim, argument) = FALSE for every argument with probability > 99%

Remarkable properties of Probabilistic Proof Systems

- Probabilistically Checkable Proofs (PCPs)

- Zero-Knowledge (ZK) proofs

Probabilistically Checkable Proofs (PCPs)

claim: The Riemann Hypothesis
Prover: (argument)
Verifier: (editor/referee/amateur)

Verifier's concern: Is the argument correct?
PCPs: Verifier reads 100 (random) bits of the argument.

Thm [Arora-Safra, Arora-Lund-Motwani-Sudan-Szegedy]:
Every proof can be efficiently transformed to a PCP
Refereeing (even by amateurs) in a jiffy!
Major application - approximation algorithms

Zero-Knowledge (ZK) proofs
[Goldwasser-Micali-Rackoff]

claim: The Riemann Hypothesis
Prover: (argument)
Verifier: (editor/referee/amateur)

Prover's concern: Will Verifier publish first?
ZK proofs: argument reveals only correctness!

Theorem [Goldreich-Micali-Wigderson]:
Every proof can be efficiently transformed to ZK proof assuming Factoring is HARD
Major application - cryptography

Conclusions & Problems

When resources are limited, basic notions get new meanings (randomness, learning, knowledge, proof, …).

- Randomness is in the eye of the beholder.
- Hardness can generate (good enough) randomness.
- Probabilistic algs seem powerful but probably are not.
- Sometimes this can be proven! (Mazes,Primality)
- Randomness is essential in some settings.

Is Factoring HARD? Is electronic commerce secure?

Is Theorem Proving Hard? Is P≠NP? Can creativity be automated

How to prove P≠NP

F field, char(F)≠2.

$X \in M_k(F)$ $\quad$ $\mathrm{Det}_k(X) = \sum_{\sigma \in S_k} \mathrm{sgn}(\sigma) \prod_{i \in [k]} X_{i\sigma(i)}$

$Y \in M_n(F)$ $\quad$ $\mathrm{Per}_n(Y) = \sum_{\sigma \in S_n} \prod_{i \in [n]} Y_{i\sigma(i)}$

Affine map L: $M_n(F) \to M_k(F)$ is good if $\mathrm{Per}_n = \mathrm{Det}_k \circ L$

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

Thm [Valiant] $\forall F$ $k(n) < \exp(n)$

Thm [Mignon-Ressayre] $\forall F$ $k(n) > n^2$

Thm [Valiant] $k(n) \neq \mathrm{poly}(n) \Leftrightarrow$ "P≠NP"

Plan of the talk

- Computational complexity
  - -- efficient algorithms, hard and easy problems, P vs. NP
- The power of randomness
  - -- in saving time
- The weakness of randomness
  - -- what is randomness ?
  - -- the hardness vs. randomness paradigm
- The power of randomness
  - -- in saving space
  - -- to strengthen proofs

Easy and Hard Problems
asymptotic complexity of functions

Turing: Formal definition of an algorithm

Multiplication
mult(23,67) = 1541

grade school algorithm:
n^2 steps on n digit inputs

EASY
P - Polynomial time
algorithm

Factoring
factor(1541) = (23,67)

best known algorithm:
$\exp(\sqrt{n})$ steps on n digits

HARD?
-- we don't know!
-- the whole world thinks so!

Theorem Proving and P vs. NP

Theorem proving :

Input: a mathematical statement S (e.g. Riemann's hypothesis)
and an integer n

Task: Find a proof of S (e.g. in ZF) of length $\leq n$ (if exists)

Theorem: If Theorem proving is Easy
then Factoring is Easy

Theorem [Cook-Levin] : Theorem proving is NP-complete

…. Numerous equally hard problems in all sciences

P vs. NP problem: Formal: Is Theorem proving Easy?
Informal: Can creativity be automated?

Fundamental question #1

Is NP$\neq$P ?   More generally, consider

- Factoring integers
- Theorem proving
- Computing the Permanent of a matrix
- Deciding knottedness of a knot
- Solving a system of polynomial equations over GF(2)
- …….

Best algorithms: exponential time.

Does any require exponential time ?

Conjecture 1 : YES

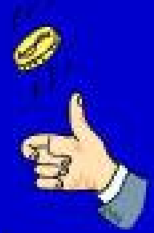
The Power of Randomness

Host of problems for which:

- We have probabilistic polynomial time algorithms

- We have no deterministic algorithms of subexponential time.

Coin Flips and Errors



Algorithms will make decisions using coin flips

0111011000010001110101010111…

(flips are independent and unbiased)

When using coin flips, we'll guarantee:

"task will be achieved, with probability >99%"

- We tolerate uncertainty in life
- Here we can reduce error arbitrarily <exp(-n)
- To compensate - we can do much more…

Number Theory: Primes

Problem 1 [Gauss]: Given $x \in [2^n, 2^{n+1}]$, is x prime?

Algorithm [Solovey-Strassen] : Probabilistic

NEW [Agrawal-Kayal-Saxena]: Deterministic !!

Problem 2: Given n, find a prime in $[2^n, 2^{n+1}]$

Algorithm: Pick at random $x_1, x_2, \ldots, x_{1000n}$

For each x_i apply primality test.

Prime Number Theorem $\Rightarrow$ Pr [$\exists i\ x_i$ prime] > .99