The power and weakness of randomness
(when you are short on time)

Avi Wigderson
Institute for Advanced Study

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

Multiplication	Factoring
mult(23,67) = 1541	factor(1541) = (23,67)
grade school algorithm:	best known algorithm:
n^2 steps on n digit inputs	$\exp(\sqrt{n})$ steps on n digits
EASY	HARD?
P – Polynomial time algorithm	-- we don't know! -- the whole world thinks so!

Map Coloring and P vs. NP

<u>Input</u>: planar map M (with n countries)

2-COL: is M 2-colorable? **Easy**

3-COL: is M 3-colorable? **Hard?**

4-COL: is M 4-colorable? **Trivial**

Thm: If 3-COL is Easy then Factoring is Easy

- Thm [Cook-Levin '71, Karp '72]: 3-COL is NP-complete
- …. Numerous equally hard problems in all sciences

P vs. NP problem: **Formal:** Is 3-COL Easy?

Informal: Can creativity be automated?

Fundamental question #1

Is NP≠P ? Is any of these problems hard?

- Factoring integers
- Map coloring
- Satisfiability of Boolean formulae
- Traveling salesman problem
- Solving polynomial equations
- Computing optimal Chess/Go strategies

Best known algorithms: exponential time/size.
Is exponential time/size necessary for some?

Conjecture 1 : YES

The Power of Randomness

Host of problems for which:

- We have probabilistic polynomial time algorithms

- We (still) have *no* deterministic algorithms of subexponential time.

Coin Flips and Errors

Algorithms will make decisions using coin flips

011101100001000111010101010111…

(flips are **independent** and **unbiased**)

When using coin flips, we'll guarantee:

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

Why tolerate errors?

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

1975 [Solovay-Strassen, Rabin] : Probabilistic
2002 [Agrawal-Kayal-Saxena]: Deterministic !!

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

Algorithm: Pick at random $x_1, x_2, \dots, x_{1000n}$
For each x_i apply primality test.
Prime Number Theorem $\Rightarrow$ $\Pr[\exists i \; x_i \text{ prime}] > .99$

Algebra: Polynomial Identities

Is $\det\begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{pmatrix} - \Pi_{i<k} (x_i - x_k) \equiv 0$?

Theorem [Vandermonde]: YES

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

Algorithm [Schwartz-Zippel '80] :

Pick r_i indep at random in {1,2,…,100d}

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

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

Applications: Program testing, Polynomial factorization

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

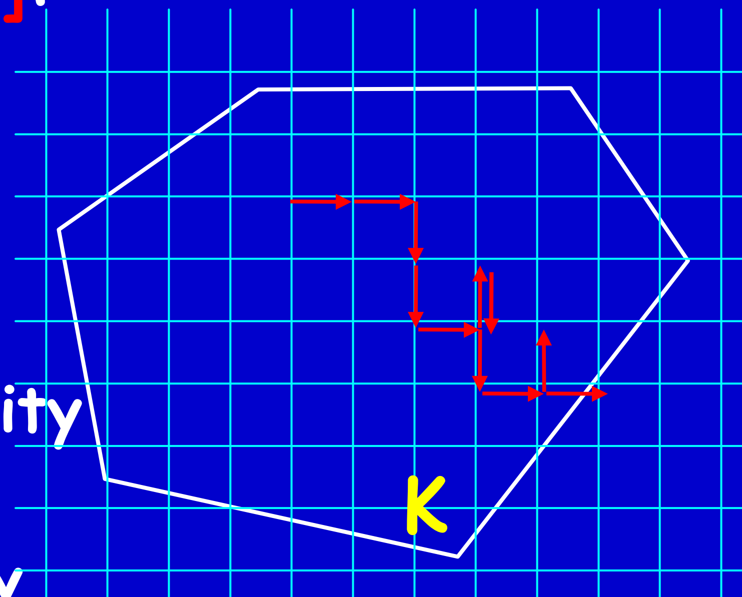
Algorithm [Goldreich-Levin '89] :
…adaptive sampling…   Pr[success] > .99

[AGS] : Extension to other Abelian groups.
Applications: Coding Theory, Complexity Theory
Learning Theory, Game 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 '91]:
Approx counting ≈ random sampling
Random walk inside K.
Rapidly mixing Markov chain.
Analysis:
Spectral gap ≈ isoperimetric inequality
Applications:
Statistical Mechanics, Group Theory



Fundamental 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

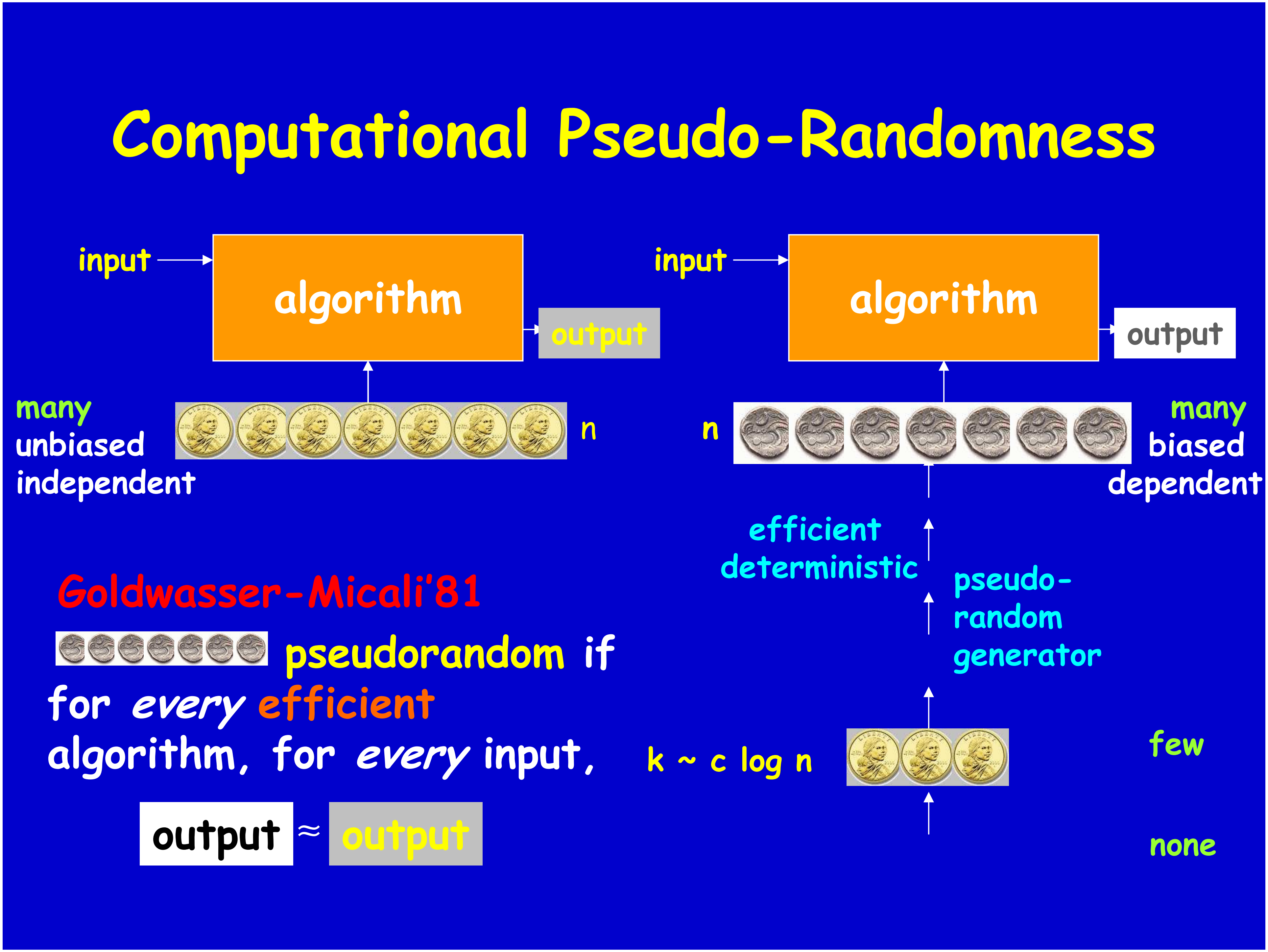
Theorem: One of these conjectures is false!

Hardness vs. Randomness

Theorems [Blum-Micali, Yao, Nisan-Wigderson, Impagliazzo-Wigderson…] :
If there are natural hard problems, then randomness can be efficiently eliminated.

Theorem [Impagliazzo-Wigderson '98]
NP requires exponential *size* circuits ⇒ every probabilistic polynomial-time algorithm has a deterministic counterpart

Theorem [Impagliazzo-Kabanets'04, IKW'03]
Partial converse!



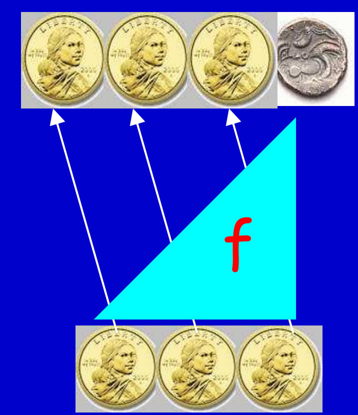
Computational Pseudo-Randomness
input
algorithm
output
many
unbiased
independent
n
input
algorithm
output
n
many
biased
dependent
efficient
deterministic
pseudo-
random
generator
k ~ c log n
few
none
Goldwasser-Micali'81
pseudorandom if
for every efficient
algorithm, for every input,
output ≈ output

Hardness ⇒ Pseudorandomness

Need G: k bits → n bits

NW generator

Show G: k bits → k+1 bits



k+1

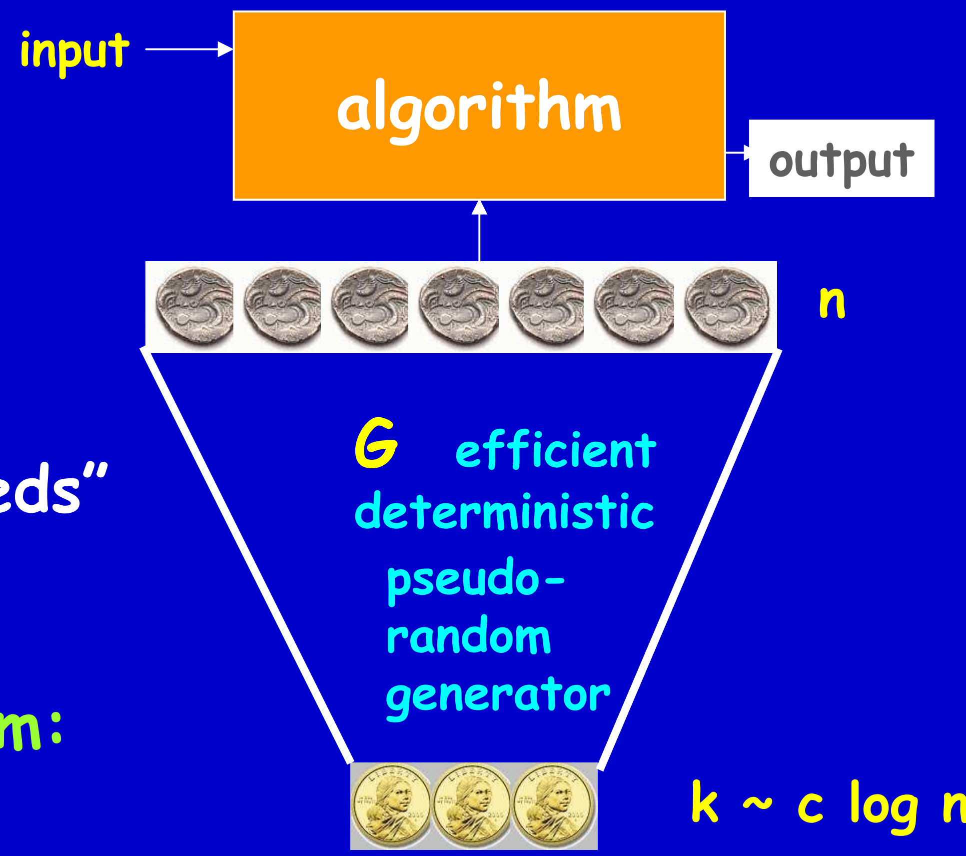
k ~ clog n

Need: f hard on *random* input **Average-case hardness**

Hardness amplification

Have: f hard on *some* input **Worst-case hardness**

Derandomization



Deterministic algorithm:

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

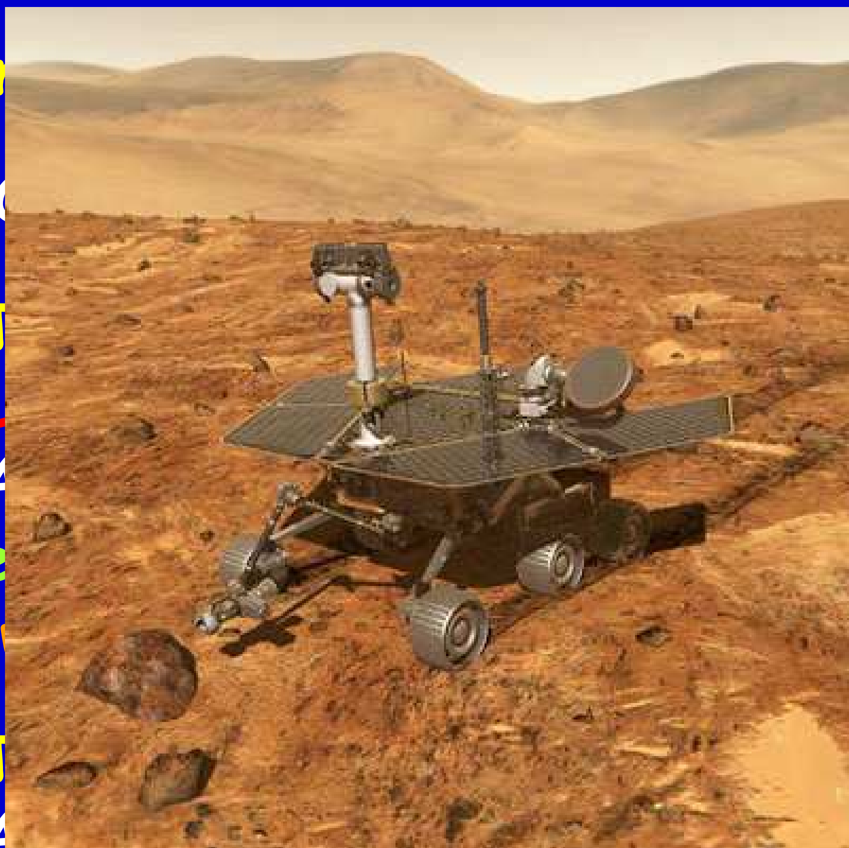
Pseudorandomness paradigm:

Can derandomize specific algorithms without assumptions!

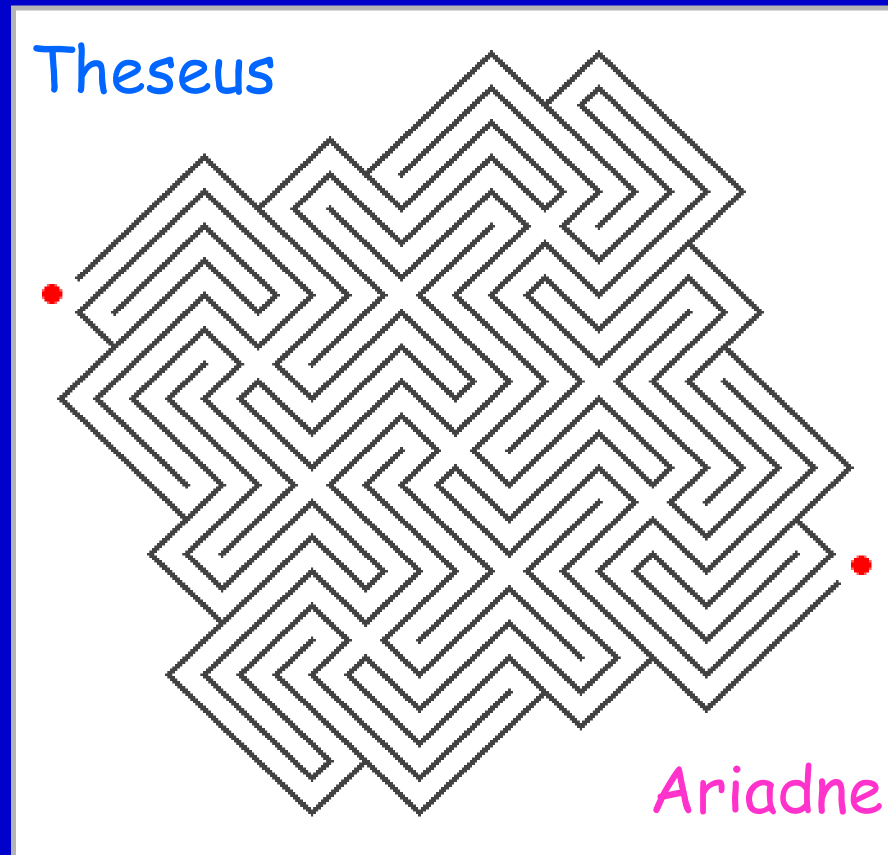
e.g. Primality Testing & Maze exploration

Randomness and space complexity

Getting out of mazes (when your memory is weak)



Mars, 2003AD



Crete, ~1000 BC

computable in logspace,
will visit every vertex.
Uses ZigZag expanders [Reingold-Vadhan-Wigderson '02]

The power of pandomness

in Proof Systems

Probabilistic Proof System

[Goldwasser-Micali-Rackoff, Babai '85]

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"

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: Has no time…

PCPs: Ver reads 100 (random) bits of argument.

Th[Arora-Lund-Motwani-Safra-Sudan-Szegedy'90]

Every proof can be eff. transformed to a PCP

Refereeing (even by amateurs) in seconds!

Major application – approximation algorithms

Zero-Knowledge (ZK) proofs
[Goldwasser-Micali-Rackoff '85]

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 '86]:
Every proof can be efficiently transformed to a 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?