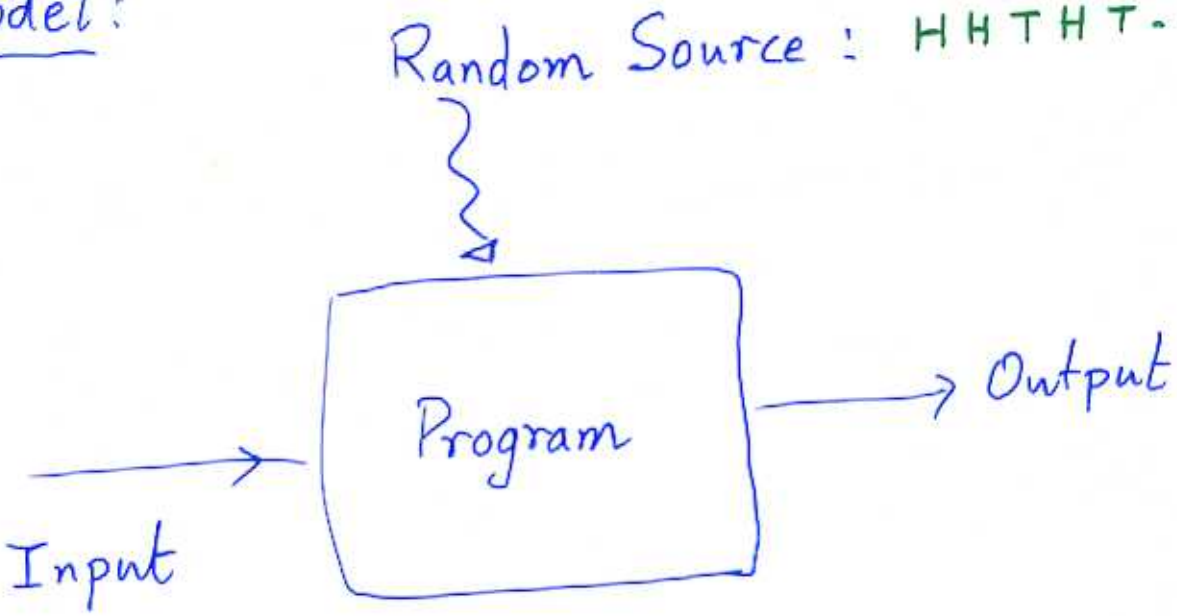
Computing with Coin Flips

V. Arvind,
Institute of Mathematical Sciences,
Chennai.

How can coin flips help compute?

Model:



Goals:

1. Programming solutions are simple.
2. ... are efficient.
3. Output is correct with high probability.

How efficient is a program?

- Our measure is time: the number of steps that the program takes to solve the problem.

Input I $\rightsquigarrow$ $T(I)$ steps.

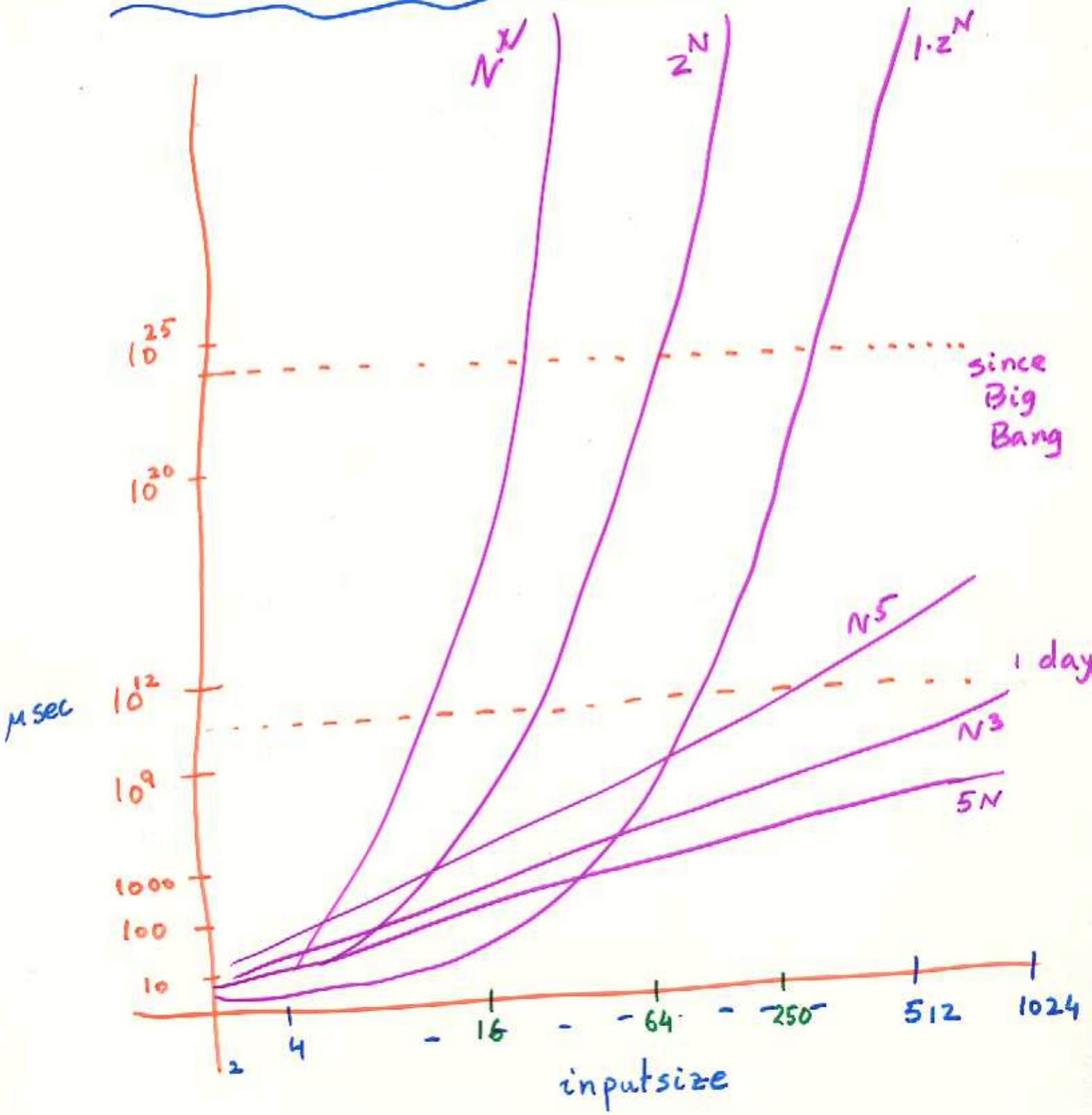
Inputs of size n $\rightsquigarrow$ $T(n)$ steps.

Good programs: $T(n)$ is a polynomial in n. Like n, n^2, or even n^{10} ...

All known algorithms for NP-complete problems take exponential time

$T(n) = n^{O(n)}$ or $2^{O(n)}$.

Why is exponential time bad?

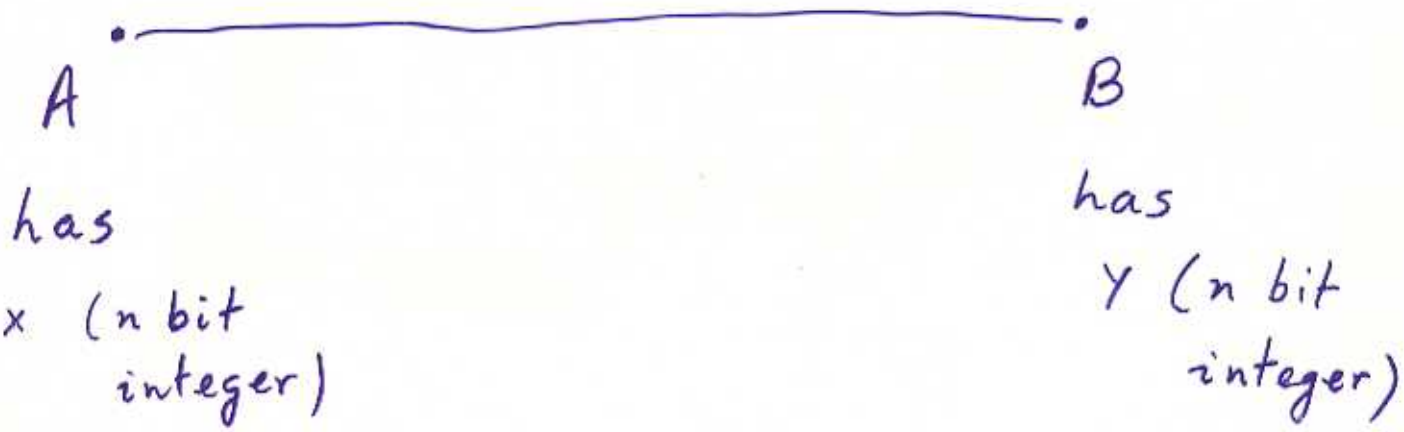


Size of unirerse : 1.4×10^{85} cubic inches

# of protons : $\sim 10^{79}$.

Randomized Fingerprints

- Example 1



Problem: Check if $x = y$ by communication of as few bits as possible.

Solution: Pick an $O(\log_2 n)$ bit prime number p at random (A does this, say). A sends both p and $x \bmod p$ to B.

B compares $x \bmod p$ with $y \bmod p$, and "declares" $x = y$ if they're equal.

Why does it work?

Chinese Remainder Theorem

Let $m_1, \ldots, m_k$ be pairwise relatively prime +ve integers and $m = \prod_{i=1}^{k} m_i$.

Then,

$\mathbb{Z}_m$ is isomorphic to

$$\mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \ldots \times \mathbb{Z}_{m_k}$$

under the function

$$f : x \longmapsto (x \bmod m_1, \ldots, x \bmod m_k)$$

In the example, let $m_1, \ldots, m_k$ be $O(\log n)$ bit primes.

There are enough of them so that $m = \prod_{i=1}^{k} m_i$ is more than n-bits long

- Example 2

Given three $n \times n$ matrices X, Y and Z over a field F (say, rationals)

we wish to verify

$$XY = Z.$$

The setting:

We'd like to check if the output is correct.

- Best known matrix product algo takes $O(n^{2.376})$ time.

- We'll design a checker that verifies $XY = Z$ in $O(n^2)$ time.

The Test:

- Pick a random vector $r \in \{0,1\}^n$

- In $O(n^2)$ time compute

$$y = Yr$$
$$x = Xy$$
$$z = Zr.$$

- Output "$XY = Z$" iff $x = z$.

Analysis: How "good" is the algorithm?
What is the error probability?

- If $XY = Z$ then for every $r \in \{0,1\}^n$ we have

$$XYr = Zr \quad \& \text{ hence } X = Z.$$

- If $XY \neq Z$ then we claim

$$\text{Prob}_{r \in \{0,1\}^n}\left[XYr = Zr\right] \leq \frac{1}{2}$$

Proof: $W = XY - Z \neq 0.$

$$\text{Prob}_{r \in \{0,1\}^n}\left[Wr = 0\right]$$

$$\leq \text{Prob}_{r \in \{0,1\}^n}\left[w^T \cdot r = 0\right] \leq \frac{1}{2}.$$

(for a non-zero row w of W)

Example 3:

Let $P_1(x)$, $P_2(x)$, $P_3(x)$ be polynomials over rationals.

$$\text{degree}(P_1) \leq n$$

$$\text{degree}(P_2) \leq n$$

We want to verify

$$P_1(x) \cdot P_2(x) = P_3(x)$$

- Explicit verification requires multiplying $P_1(x)$ & $P_2(x)$.

- Can we use randomness to do it faster?

- Pick a random number $r \in [1..n^2]$
- Check if $P_1(r) \cdot P_2(r) = P_3(r)$

Probabilistic Proof ?

Binomial theorem :

$$(x+1)^n = \sum_{i=0}^{n} \binom{n}{i} x^i \qquad (*)$$

Pick a random $r \in [1 \ldots n^2]$ and check (*).

If (*) were false then

$$L\text{-}H\text{-}S \neq R\text{-}H\text{-}S$$

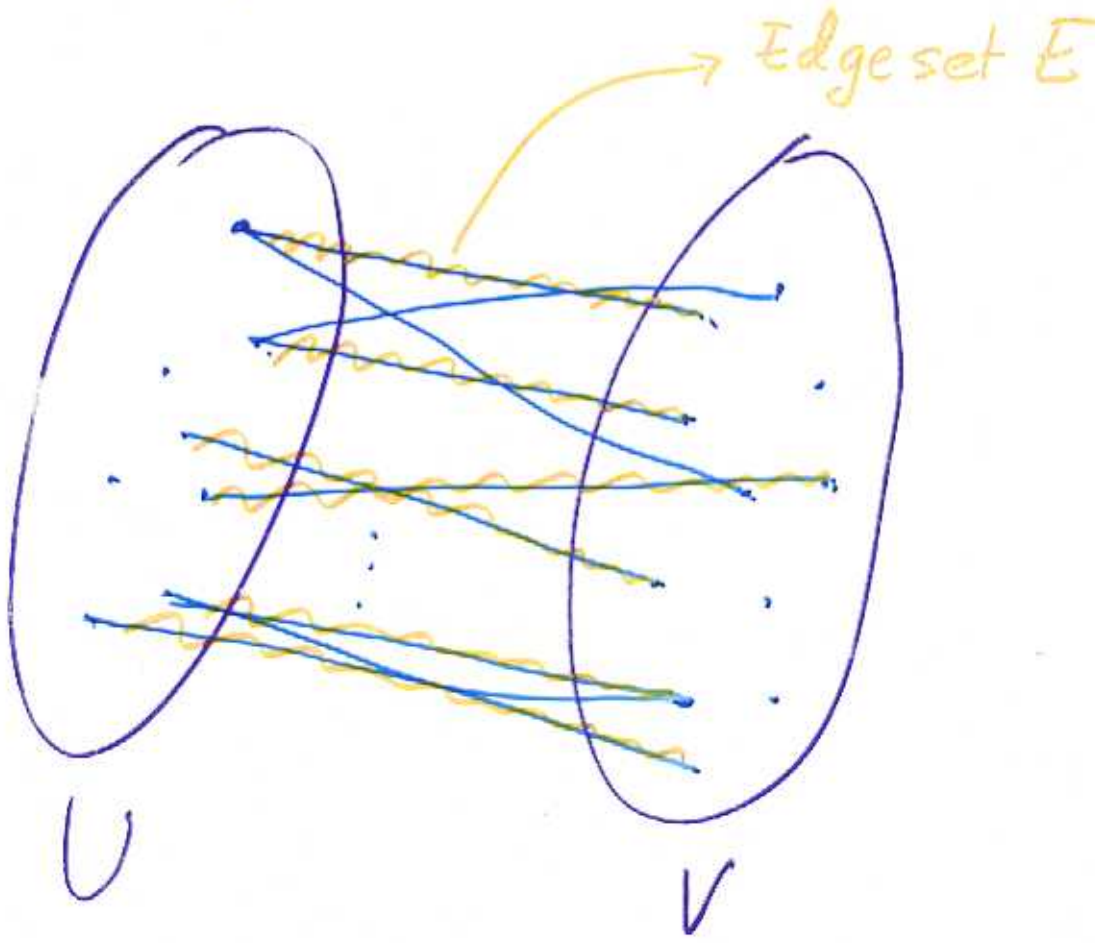
with probability $\geq 1 - \frac{1}{n}$.

A different notion of mathematical proof ??

- Example 4 : (PERFECT MATCHINGS IN GRAPHS)

Consider bipartite graph $G = (U, V, E)$

$|U| = |V| = n$.



- A <u>matching</u> is a collection of edges no two of which share an end-point.

- A <u>perfect matching</u> in G is a matching of size n (every vertex is matched).

Problem: Given a bipartite graph $G = (U, V, E)$ with $|U| = |V| = n$ find a perfect matching (if one exists).

$$U = \{u_1, u_2, \ldots, u_n\}$$

$$V = \{v_1, v_2, \ldots, v_n\}$$

Any perfect matching looks like $\{(u_i, v_{\pi(i)}) \mid i = 1, \ldots, n\}$ for some permutation π on $\{1, 2, \ldots, n\}$.

ADJACENCY MATRIX OF G:

$$A = \begin{array}{c} \\ u_1 \\ u_2 \\ \vdots \\ u_i \\ \vdots \\ u_n \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & & v_j & & v_n \end{array} \\ \begin{bmatrix} 1 & 1 & & 0 & & \\ 1 & 0 & & & & \\ 0 & 1 & & 1/0 & & \\ & 0 & & & & \end{bmatrix} \end{array}$$

$A_{ij} = 1$ if $(u_i, v_j) \in E$

$A_{ij} = 0$ if $(u_i, v_j) \notin E$

$$\det(A) = \sum_{\pi} \operatorname{sgn}(\pi)\, A_{1,\pi(1)} A_{2,\pi(2)} \cdots A_{n,\pi(n)}$$

This term is 1 if $\{(u_i, v_{\pi(i)}) \mid 1 \le i \le n\}$ is a perfect ~~match~~ matching & 0 otherwise.

Easy to evaluate by Gaussian elimination.

Modify A as follows:

$$A_{i,j} = \begin{cases} x_{ij} & \text{if } (u_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Claim: G has a perfect matching iff the multivariate polynomial $\det(A) \not\equiv 0$.

Theorem: Let $Q(x_1, x_2, \ldots, x_n)$ be an n-variate polynomial (over rationals, say) of total degree m.

Let S be a finite subset of the field F and let $r_1, r_2, \ldots, r_n$ be picked independently and uniformly at random from S.

Then

$$\text{Prob}\left[Q(r_1, \ldots, r_n) = 0 \mid Q \not\equiv 0 \right] \leq \frac{m}{|S|}$$

Proof: (By induction on n)

Base case: ($n = 1$) we know that a poly $Q(x_1)$ of degree m has at most m zeros

Induction hypothesis: Suppose theorem holds for $(n-1)$-variate polynomials, where $n > 1$.

Tossing a coin over the phone :

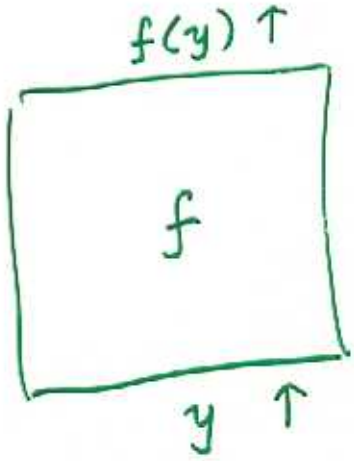
Saurav — Inzamam

Sends random $r \in \{0,1\}^n$ →

Toss to get $b \in \{0,1\}$.

Pick $y \in_R \{0,1\}^n$.

f is a 1-way permutation.



← send $\langle f(y), b \oplus y \cdot r \rangle$.

Toss done!

Send guess b' →

← reveal b & y

Secret Sharing:

k = secret.

Problem: Share it among n parties so that at least t should get together to find k.

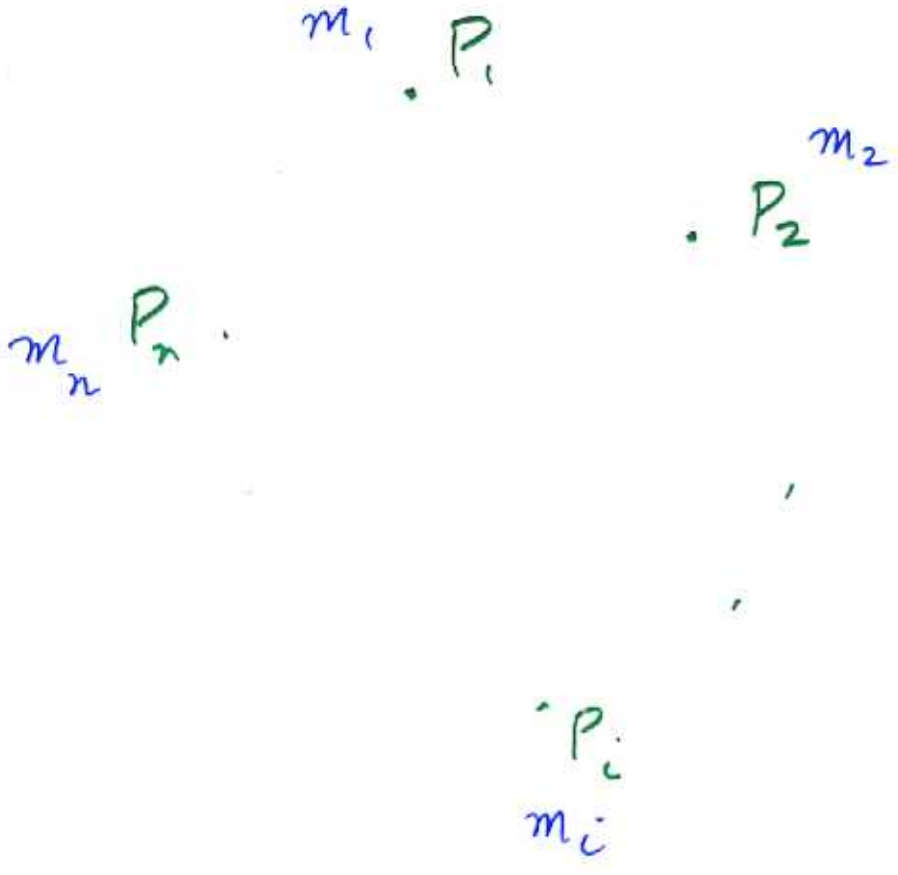
$k \in \mathbb{Z}_p$. Pick $a_1, \ldots, a_{t-1} \in_R \mathbb{Z}_p$

$$a(x) = k + a_1 \cdot x + \cdots + a_{t-1} \cdot x^{t-1}.$$

Let $u_1, u_2, \ldots, u_n \in \mathbb{Z}_p$ be distinct.

Give $a(u_i)$ to P_i, $1 \le i \le n$.

Finding the class average "secretly"



P_1: Pick $k_2, k_3, \ldots, k_n \in_R \mathbb{Z}_N$

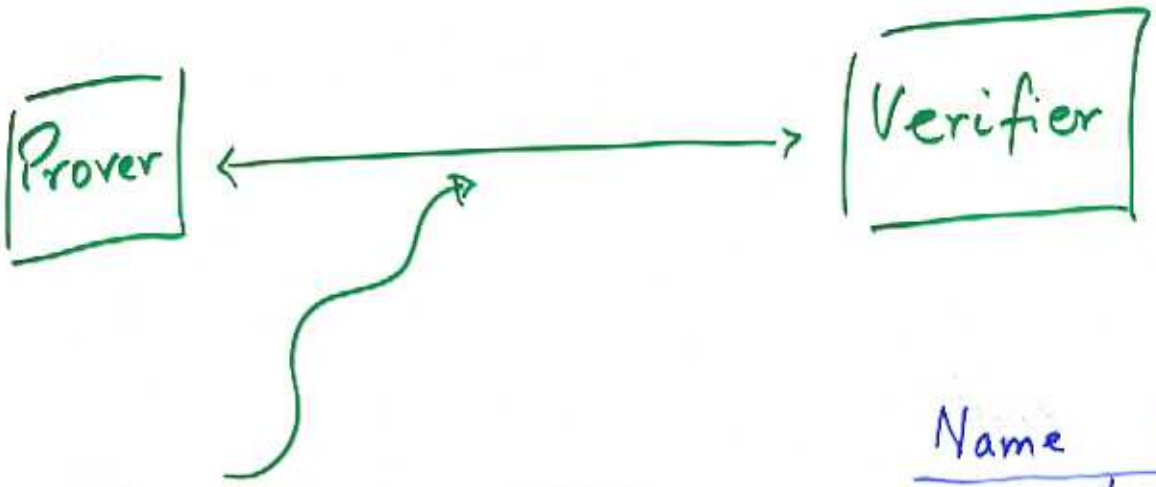
Let $k_1 = m_1 - \left(\sum_{j=2}^{n} k_j\right) \bmod N$.

(i) Send k_j to P_j, $2 \le j \le n$.

(ii) Add numbers received to k_1 and announce the total T_1.

(iii) $\sum_{i=1}^{n} T_i \pmod N$ is $\sum_{i=1}^{n} m_i$.

Zero knowledge proofs :

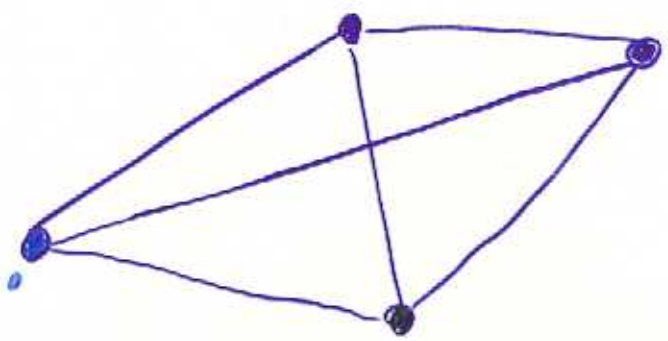


Eavesdropper sees random strings.

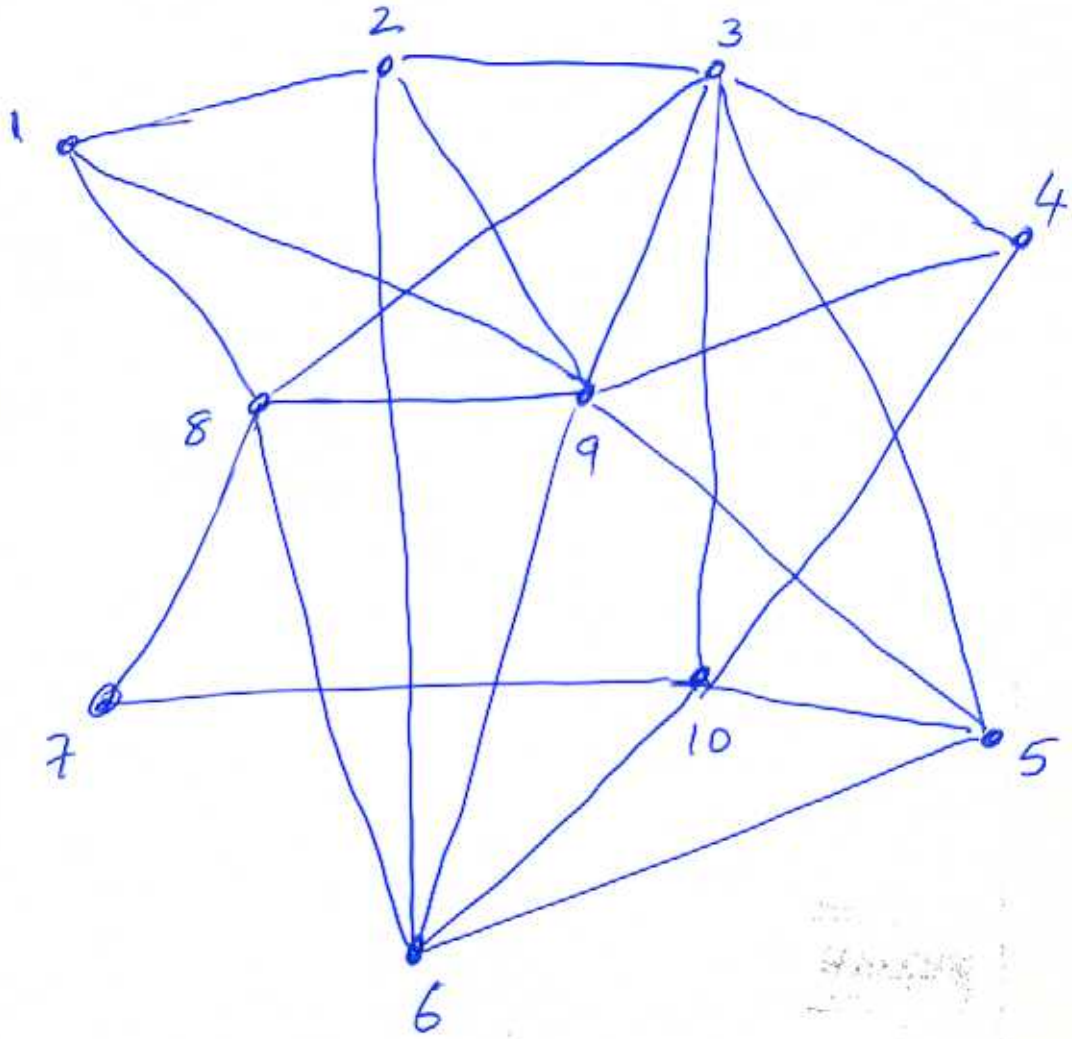
Name	Graph
⋮	
arvind	G.

Challenge :

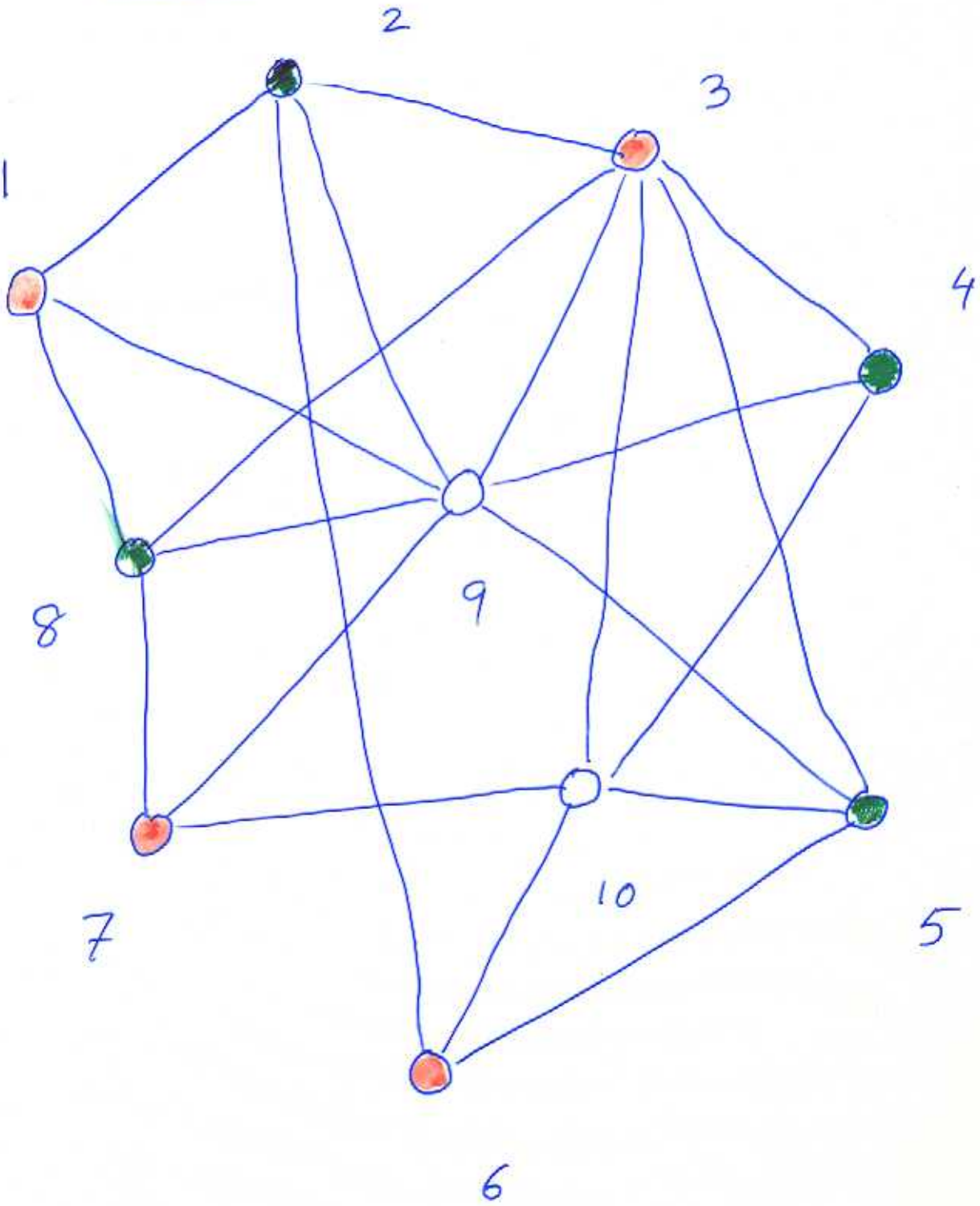
Prove that you know a 3-coloring of G !



The graph:

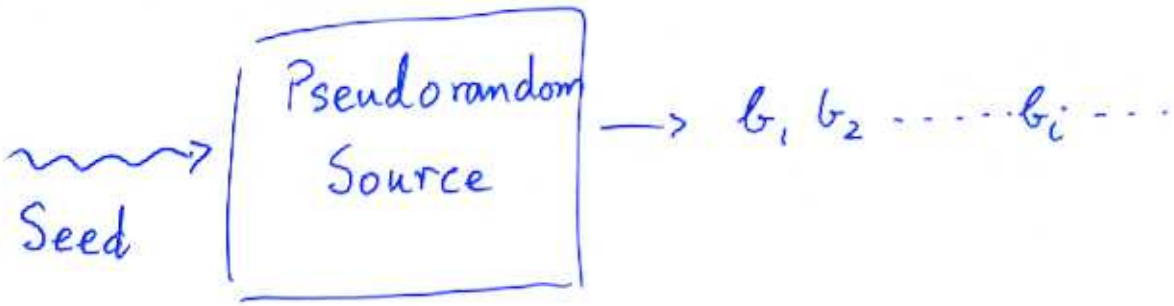


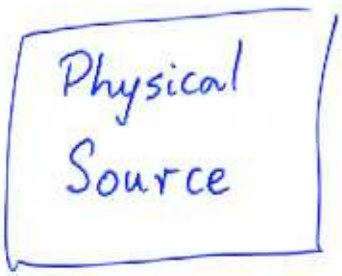
A zero-knowledge proof of 3-colorability



Randomness as resource :

. Is there randomness in nature ?





Extractors : $\{0,1\}^m \times \{0,1\}^s \to \{0,1\}^{m+s-e}$

Raw — pure — close to pure.

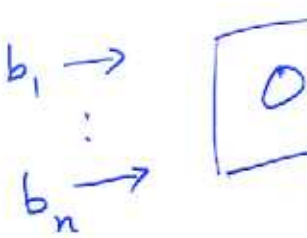
Physics & Computation

- Does computation dissipate heat?

- Irreversible vs reversible computation.

"thermodynamics" ?

Landauer's principle:



Entropy increases by $\log_2 n \cdot 2.8 \times 10^{-21}$ Joules.

Information entropy $\overset{?}{=}$ Thermodynamic entropy.

Quantum Computing:

Randomness in nature

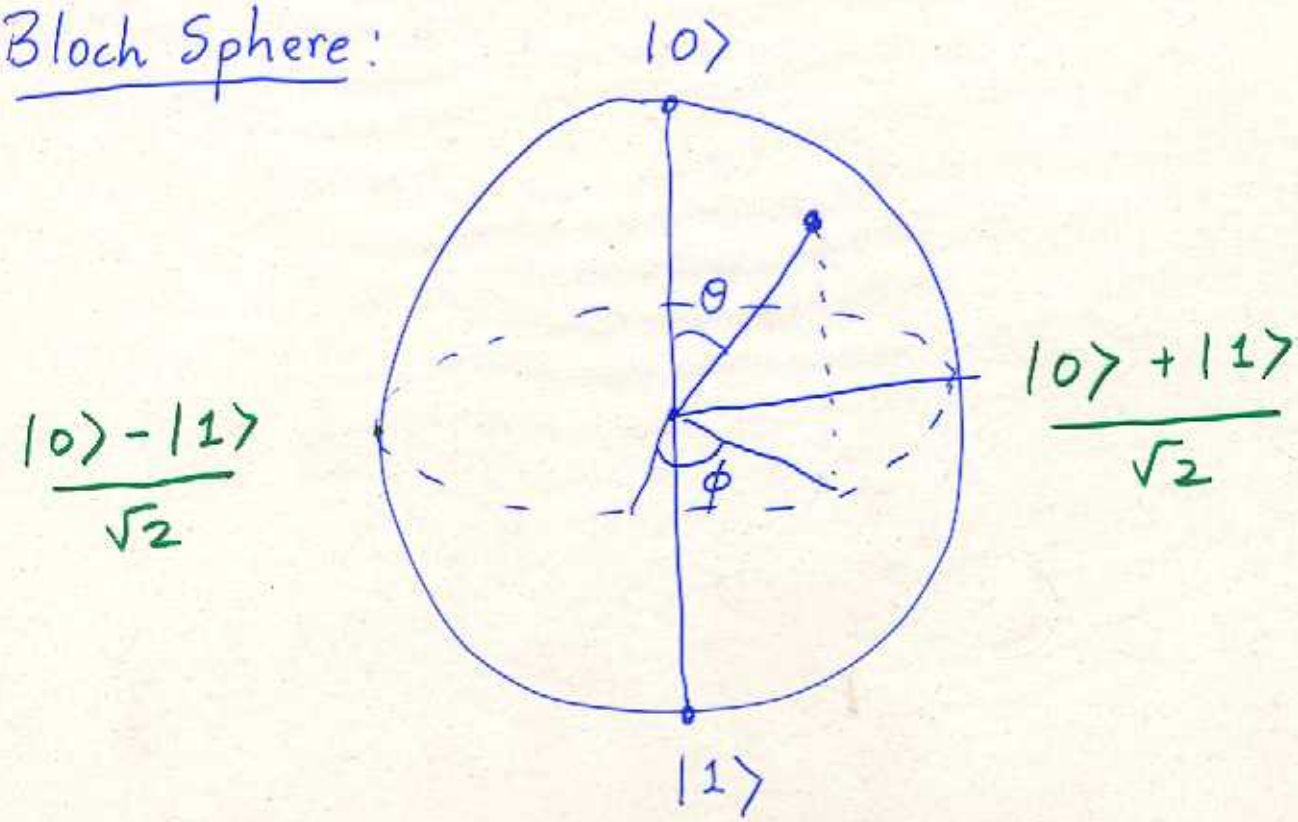
Qubit:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

α, β : complex numbers

$$|\alpha|^2 = |\beta|^2 = 1.$$

Bloch Sphere:

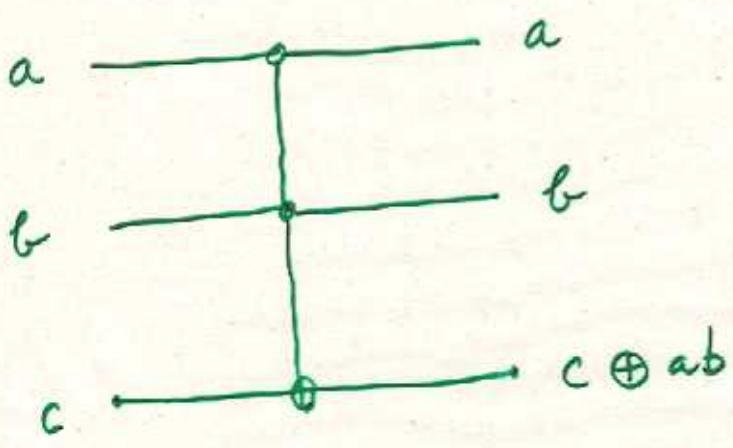


Reversible computation:

1. Time-bounded computation is easily converted to circuits.

$T(n)$ time-bound $\longrightarrow$ $O(T(n))^2$ size boolean circuits.

2. Reversible boolean gates:

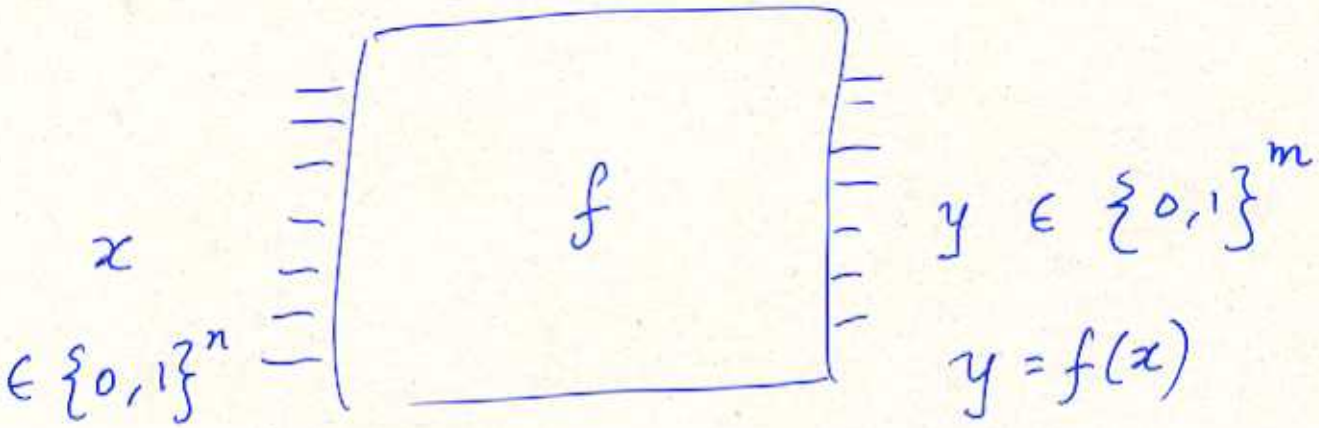


Toffoli gate

Permutation on $\{0,1\}^8$.

$$\left(\begin{array}{c:c} I & 0 \\ \hdashline 0 & I \end{array}\right)$$

Reversible computation contd.



Convert it to

$$U_f : \langle x, z \rangle \mapsto \langle x, f(x) \oplus z \rangle.$$

- Can be done by replacing gates in f by suitable Toffoli gates.

Thus,

Classical computation $\equiv$ Multiplying permutation matrices.

Back to Quantum Computing:

"Basis" states $= \{0,1\}^n$.

Quantum states are unit vectors in a 2^n-dimensional complex vector space.

$$|\psi\rangle = \sum_{x \in \{0,1\}^n} \alpha_x \cdot |x\rangle$$

$$\sum_{x \in \{0,1\}^n} |\alpha_x|^2 = 1 .$$

Quantum Gates:

Unitary matrices : map unit vectors to unit vectors.

$$U \cdot U^\dagger = I .$$

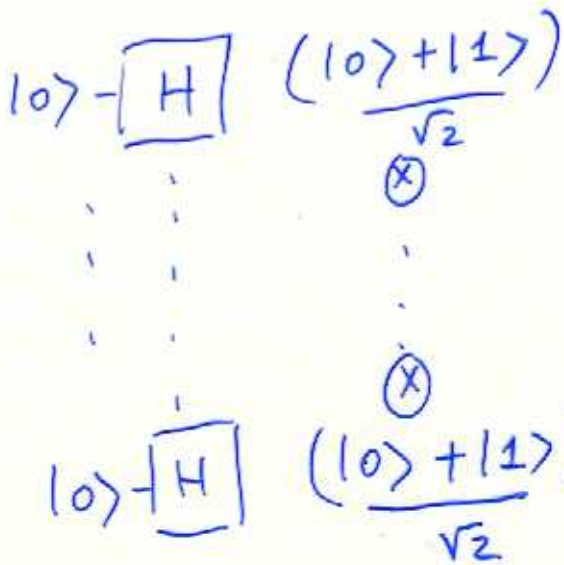
Quantum coin-toss gate:

Hadamard gate:

$$H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

$$H : |0\rangle \longmapsto \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$|1\rangle \longmapsto \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$



$$H_n|u\rangle = \frac{1}{2^{n/2}} \sum_x (-1)^{x \cdot u} |x\rangle.$$

Theory and Cutting-Edge?

"The trouble with Computer Science is that it is so applicable that you can be blamed for not doing immediately applicable research.

There is such an urgent demand for applications, that if you do work with long term implications, you can be criticized.

That is a great danger, because C.S. is so broad, so much computing happening, ... we will all be lost without a basic theory."

Robin Milner, ACM Turing Award Lecture, 1992.