What is a random string of bits ?
what properties does it have ?

Distribution over strings

$U_n : \forall x \in \{0,1\}^n,$

$U_n(x) = \Pr[x \sim U_n] = \frac{1}{2^n}$

Shannon's Entropy

$H(D_n) = \mathbb{E}_{x \sim D_n}\left[\log \frac{1}{D_n(x)}\right]$

$= \sum D_n(x) \cdot \log \frac{1}{D_n(x)}$

Single string

10001011100111

00110111100101

Kolmogorov Complexity

$K(x) = \min\{|(M,w)| :$

TM M on input w prints out x.

$$= \sum_{x \in \{0,1\}^n} \mathcal{D}_n(x) \cdot \log \frac{1}{\mathcal{D}_n(x)}$$

prints out x.

$$H(U_n) = n$$

"most" random dist.

$$K(x) \geq |x|$$

"most" random string

$$H(\mathcal{D}_n) \cong \mathop{\mathbb{E}}_{x \sim \mathcal{D}_n} [K(x)]$$

(for recursive $\mathcal{D}_n$)

Uses of randomness in computation

- algorithms (randomized P)
- "complexity-theoretic" algorithms (randomized NP)
- (circuit) lower bound proofs
- cryptography & pseudorandomness
- derandomization
- learning algorithms
- average case complexity

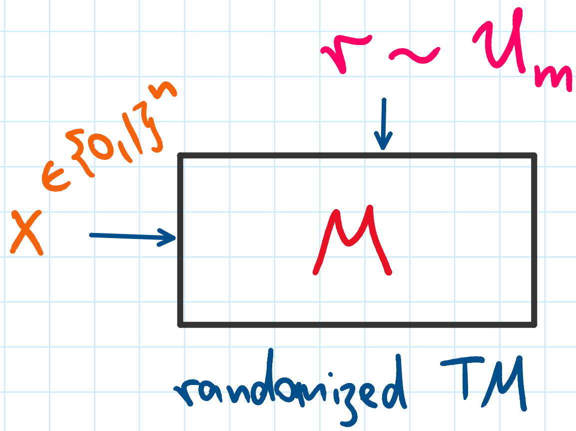
- average & case complexity

Randomized Algorithms

Turing (1948):

Partially random and apparently partially random machines

It is possible to modify the above described types of discrete machines by allowing several alternative operations to be applied at some points, the alternatives to be chosen by a random process. Such a machine will be described as 'partially random'. If we wish to say definitely that a machine is not of this kind we will describe it as 'determined'. Sometimes a machine may be strictly speaking determined but appear superficially as if it were partially random. This would occur if for instance the digits of the number π were used to determine the choices of a partially random machine, where previously a dice thrower or electronic equivalent had been used. These machines are known as apparently partially random.

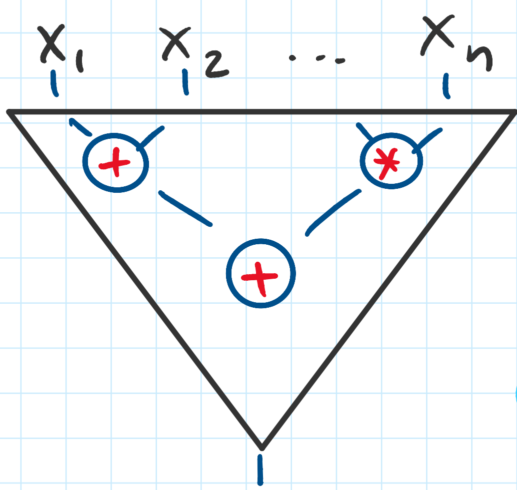


For all $x \in \{0,1\}^*$,

$\Pr_r\big[\, M(x,r)$ is correct on $x \,\big] \approx 1$

__Example:__ Randomized Polytime algorithm for Polynomial Identity Testing (PIT)

__Given:__



arithmetic circuit $C(x_1,\dots,x_n)$ (over $\mathbb{Z}$)

__Decide:__ $C \equiv 0$?

__SZ Lemma__ For any polynomial $p(x_1,\dots,x_n)$ over $\mathbb{Z}$ of degree d, and any finite set $S \subseteq \mathbb{Z}$.

and any finite set $S \subseteq \mathbb{Z}$,

$$p \not\equiv 0 \Rightarrow \Pr_{\substack{r_1 \sim S, \\ \vdots \\ r_n \sim S}} \left[p(r_1, \ldots, r_n) = 0 \right] \leq \frac{d}{|S|}.$$

(Proof by induction on n.)

Randomized algo for PIT on $C(x_1, \ldots, x_n)$

[if $\deg(C) \leq \mathrm{poly}(|C|)$]

1. Choose $r_1, \ldots, r_n \in \{1, 2, 3, \ldots, 4 \cdot \deg(C)\}$ independently uniformly at random
2. Evaluate $C(r_1, \ldots, r_n)$
3. If $C(r_1, \ldots, r_n) \neq 0$, output "Non-zero" else output "Zero".

- Time: $\mathrm{poly}(|C|, \deg(C))$
- Correctness:
  - $C \equiv 0 \Rightarrow$ output "Zero" always

• Correctness : • $C \equiv 0 \Rightarrow$ output zero always
• $C \not\equiv 0 \Rightarrow$ by SZ Lemma, will output "Non-zero" with prob. $\geq 1 - \frac{\deg(C)}{4 \cdot \deg(C)}$
$= \frac{3}{4}$.

Remark: Can get time $\text{poly}(|C|)$.

• $\deg(C) \leq 2^{|C|}$
• $S = \{1, 2, \ldots, 4 \cdot 2^{|C|}\}$
• pick $r_1, r_2, \ldots, r_n \in S$ (indep. at random)
• evaluate $C(r_1, r_2, \ldots, r_n) \bmod p$ for a random prime p, $2 \leq p \leq 2^{4 \cdot |C|}$ gate by gate
• if the result is 0, then output "Zero" else output "Non-zero".

Correctness: For $C \neq 0$,

$$\Pr_{r_1,\dots,r_n}\left[C(r_1,\dots,r_n)=0\right] \leq \frac{1}{4}.$$

- $|C(r_1,\dots,r_n)| \leq \left(4 \cdot 2^{|C|}\right)^{2^{|C|}}$

has $\leq (|C|+2)\cdot 2^{|C|} \leq 2^{2\cdot|C|}$

distinct prime factors

- $\{2,\dots,2^{4\cdot|C|}\}$ contains $\geq 2^{4\cdot|C|} / \ln(2^{4\cdot|C|})$

$\gtrapprox 2^{3\cdot|C|}$ distinct primes

(density of primes)

$$\Pr_{\substack{r_1,\dots,r_n \\ p}}\Big[\underbrace{C(r_1,\dots,r_n)=0 \bmod p}_{\text{error}}\Big] \leq \frac{1}{4} + \frac{2^{2\cdot|C|}}{2^{3\cdot|C|}}$$

$$\leq \frac{1}{4} + o(1).$$

Decision problem (language) $L \subseteq \{0,1\}^*$ is in

- BPP (two-sided error)

  if $\exists$ poly(n)-time TM M

  $\forall x \in \{0,1\}^n \quad \Pr_r\left[M(x,r) = L(x)\right] \geq \frac{3}{4}$

- RP (one-sided error)

  if $\exists$ poly(n)-time TM M

  $\forall x \in \{0,1\}^n$

  - $x \in L \implies \Pr_r\left[M(x,r) = 1\right] \geq \frac{1}{2}$
  - $x \notin L \implies \Pr_r\left[M(x,r) = 0\right] = 1$

- ZPP (zero error)

  if $\exists$ TM M, correct $\forall x \in \{0,1\}^n$

  where $M(x,r)$ runs in expected polytime

  $$\mathbb{E}_r\left[\text{time of } M(x,r)\right] \leq \text{poly}(|x|)$$

Equivalently,

Equivalently,

$$ZPP = RP \cap coRP$$

Error reduction: Running a given randomized algo K times (with fresh randomness each time), can reduce the error probability $\leq 2^{-\Omega(K)}$.

[Chernoff Bounds]

Note: $PIT \in coRP$

$$P \subseteq ZPP \subseteq RP \subseteq BPP$$

Conjecture: $P = BPP$

- $ZPP = P$?
- $ZPP \neq EXP$?

(cf. P vs. $NP \cap coNP$)

Randomized NP and beyond

$$P \subseteq ZPP \subseteq RP \subseteq BPP \subseteq NP^{NP} = \Sigma_2^P$$
$$RP \subseteq NP$$

Def of NP: $L \in NP$ if

$x \in \{0,1\}^n$

Prover $\xrightarrow{w \in \{0,1\}^{poly(n)}}$ Verifier $\in P$

$x \in L \Rightarrow \exists w \;\; V(x,w)$ accepts

$x \in L \Rightarrow \exists w \quad V(x,w)$ accepts
$x \notin L \Rightarrow \forall w \quad V(x,w)$ rejects

~~Def of NP~~ MA: $\quad L \in \cancel{NP}$ MA if

$x \in \{0,1\}^n$

Prover $\xrightarrow{w \in \{0,1\}^{poly(n)}}$ Verifier $\in \cancel{P}$ BPP

$x \in L \Rightarrow \exists w \ \Pr_r[V(x,w,r) \text{ accepts}] \geq \frac{3}{4}$
$x \notin L \Rightarrow \forall w \ \Pr_r[V(x,w,r) \text{ rejects}] \geq \frac{3}{4}$

AM $= AM(2)$:

$x \in \{0,1\}^n$

Prover $\xleftarrow{r \in \{0,1\}^*}$ Verifier

Prover $\xrightarrow{w \in \{0,1\}^*}$ Verifier

$x \in L \Rightarrow \Pr_r[\exists w \quad V(x,w,r) \text{ accepts}] \geq \frac{3}{4}$
$x \notin L \Rightarrow \Pr_r[\forall w \quad V(x,w,r) \text{ rejects}] \geq \frac{3}{4}$

AM(k): allow $k \geq 2$ rounds of interaction between Arthur & Merlin

$$AM(\text{const}) = AM(2) = AM$$

Examples:

[MA]

Define $\text{perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}$ for $A \in \mathbb{Z}^{n \times n}$.

(#P-complete)

Thm: perm has arithmetic polysize circuits $\Rightarrow$ perm $\in$ MA.

Proof:

Important property of perm: $\text{perm}(A) = \sum_{i=1}^{n} a_{1,i} \cdot \text{perm}(A_{1,i})$

drop row 1 & column i

Defining axioms:

$f_1, f_2, \ldots, f_n$ compute perm on $1\times1, 2\times2, \ldots, n\times n$ matrices $\iff$

$f_1(x) \equiv x$, and $\forall k \geq 2$

$$(*)\begin{cases} f_1(x) \equiv x, \text{ and } \forall k \geq 2 \\ f_k(X) = \sum_{i=1}^{k} x_{1,i} \cdot f_{k-1}(X_{1,i}) \end{cases}$$

Suppose perm has polysize arithmetic circuits.
To compute perm(A), $A \in \mathbb{Z}^{n \times n}$,

Merlin sends to Arthur polysize ar. circuits
$C_1, C_2, \ldots, C_n$

Arthur checks that
$$\left\{\begin{array}{l} C_1(x) \equiv x, \ \& \\ C_k(X) = \sum x_{1,i} \cdot C_{k-1}(X_{1,i}) \\ \forall \ 2 \leq k \leq n \end{array}\right.$$
PIT tests (hence in BPP)

If all checks pass, output $C_n(A)$.

Corollary 1: If (1) perm $\in$ arithmetic PolySize, &
(2) PIT $\in$ P,

then

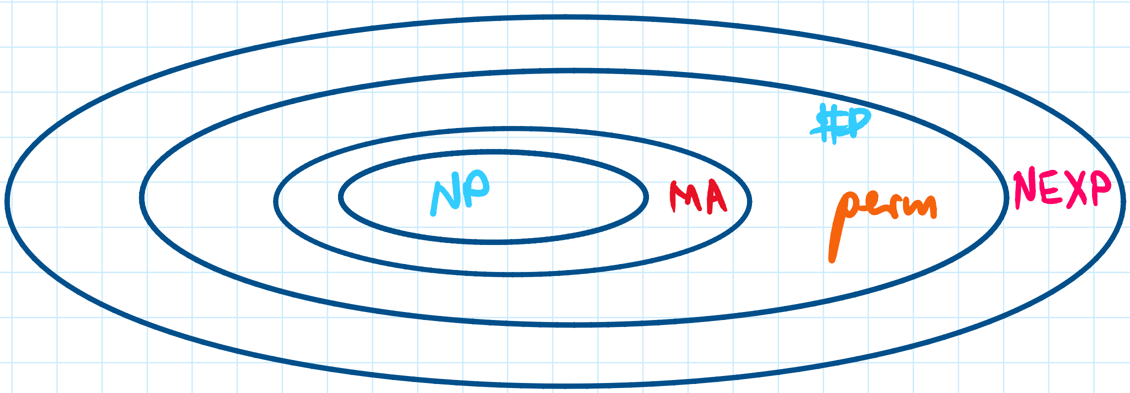
(2) PIT $\in$ P,

then perm $\in$ NP.

Application (derandomizing PIT requires proving circuit lower bounds):

If PIT $\in$ P, then (a) perm $\notin$ arithm. PolySize
or (b) NEXP $\notin$ PolySize.

Proof: If NEXP $\notin$ PolySize, done.
Otherwise, by the Easy Witness Lemma,
NEXP = MA.
If perm $\notin$ arithm. PolySize, done.
Otherwise, by Corollary 1 above,
perm $\in$ NP



Hence, NEXP = NP. Contradiction (Hierarchy Thm).

[AM] : - Graph Non-Isomorphism $\in AM$

- $\forall$ constant $k \geq 2$, $AM(k) = AM$.

- perm $\in AM(poly)$ [LFKN]

$IP = PSPACE$

<u>Given:</u>

prime $p \geq n^4$

matrix $A \in \mathbb{F}_p^{n \times n}$

$v \in \mathbb{Z}$

<u>Prove:</u> $v = \text{perm}(A)$.

$A : \quad A_{1,1}, A_{1,2}, \ldots, A_{1,n}$ (minors of A)

$(n-1) \times (n-1)$

for each $1 \leq j, k \leq n-1$, by interpolation, define a unique polynomial $f_{j,k}(x)$ of degree $\leq n$

polynomial $f_{j,k}(x)$ of degree $\leq n$

Such that $\forall\ 1 \leq i \leq n$, $f_{j,k}(i) = (A_{1,i})_{j,k}$

Note: $g(x) := \text{perm}\begin{bmatrix} f_{1,1}(x) & \cdots & f_{1,n-1}(x) \\ \vdots & & \\ f_{n-1,1}(x) & \cdots & f_{n-1,n-1}(x) \end{bmatrix}$ is a polynomial of degree $\leq n^2$.

$$\text{perm}(A) = \sum_{i=1}^{n} a_{1,i} \cdot \text{perm}(A_{1,i})$$
$$= \sum_{i=1}^{n} a_{1,i} \cdot g(i).$$

Merlin $\xrightarrow[\text{coefficients of } h(x) \overset{?}{\equiv} g(x)]{\text{"} V \overset{?}{=} \text{perm}(A) \text{"}}$ Arthur

Check $V = \sum_{i=1}^{n} a_{1,i} \cdot h(i)$

if not equal, Reject!

Otherwise,

$\xleftarrow{b \sim GF(p)}$

expect the proof from Merlin that

"$h(b) \stackrel{?}{=} \mathrm{perm}\begin{bmatrix} f_{11}(b) & \cdots & f_{1,n-1}(b) \\ \vdots & & \\ f_{n-1,1}(b) & \cdots & f_{n-1,n-1}(b) \end{bmatrix}$"

The Base Case if $n=1$:
check $v = \mathrm{perm}(A)$
yourself.

Correctness:

- Honest Merlin will always win.
- Merlin cheating in a given round will be forced to cheat in the next round with high probability.

(To get lucky, Merlin must get $b \in GF(p)$ such that,
for $g(x) \neq h(x)$,
it still holds
$g(b) = h(b)$.

By SZ Lemma, the probability of that is
$\leq \frac{\deg(g-h)}{p} \leq \frac{n^2}{n^4} = \frac{1}{n^2}$.)