CS 121: Lecture 24
Intro to Randomized Algorithms

Adam Hesterberg

https://madhu.seas.Harvard.edu/courses/Fall2020

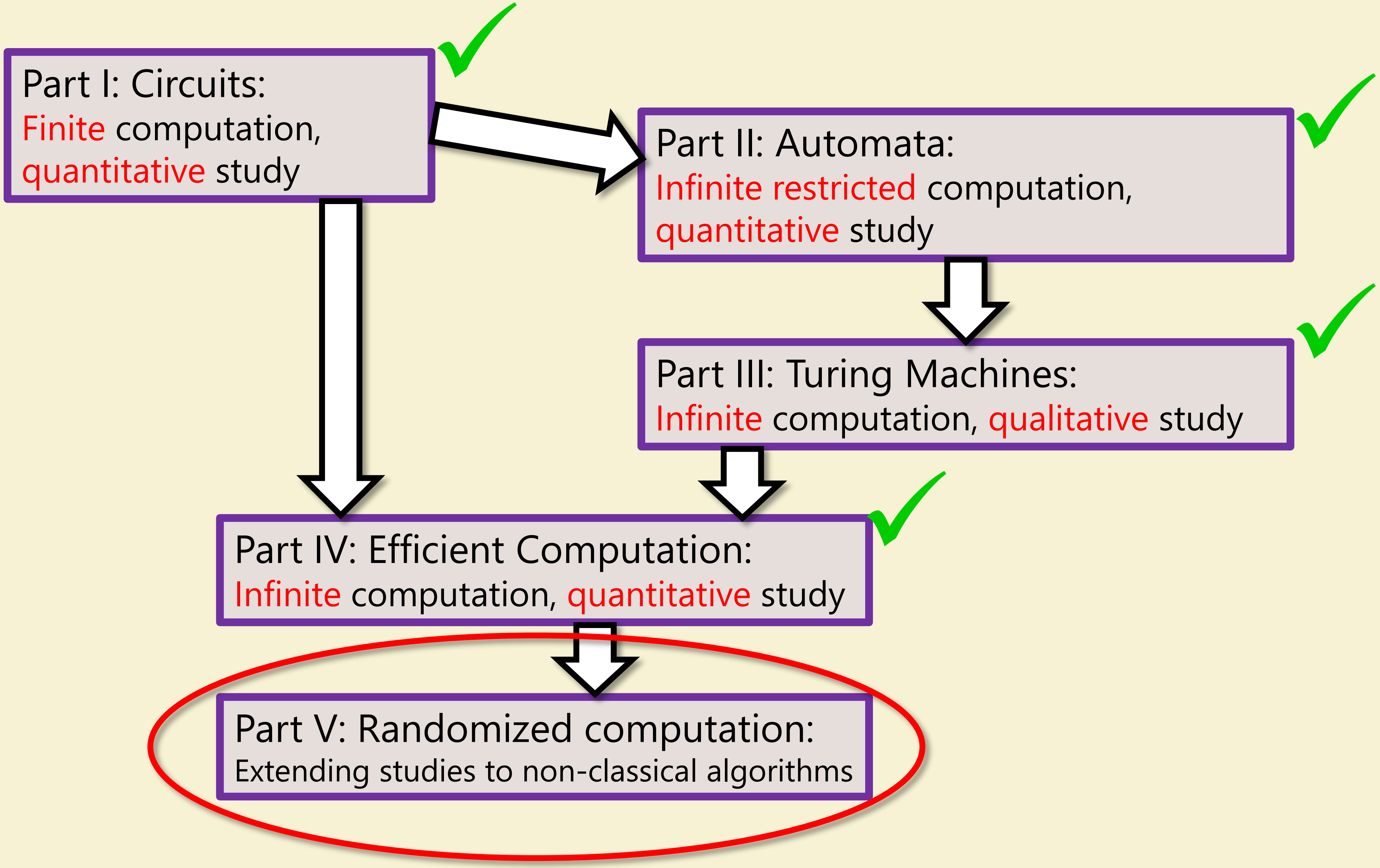
Book: https://introtcs.org

How to contact us
- The whole staff (faster response): CS 121 Piazza
- Only the course heads (slower): cs121.fall2020.course.heads@gmail.com

Announcements:

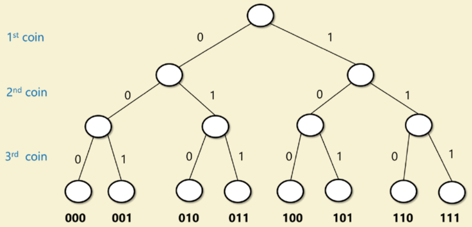
- Midterm 2 graded. Solutions to be posted today-ish.
- Thanks for participating in Midterm Feedback Survey.
- Happy Thanksgiving! (Next lecture Tuesday.)

Where we are:

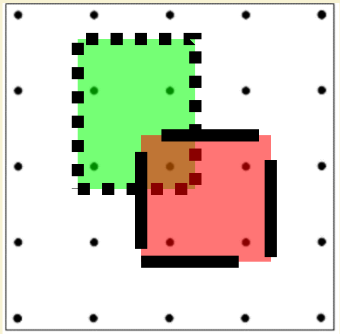

Part I: Circuits:
Finite computation,
quantitative study
Part II: Automata:
Infinite restricted computation,
quantitative study
Part III: Turing Machines:
Infinite computation, qualitative study
Part IV: Efficient Computation:
Infinite computation, quantitative study
Part V: Randomized computation:
Extending studies to non-classical algorithms

Last lecture

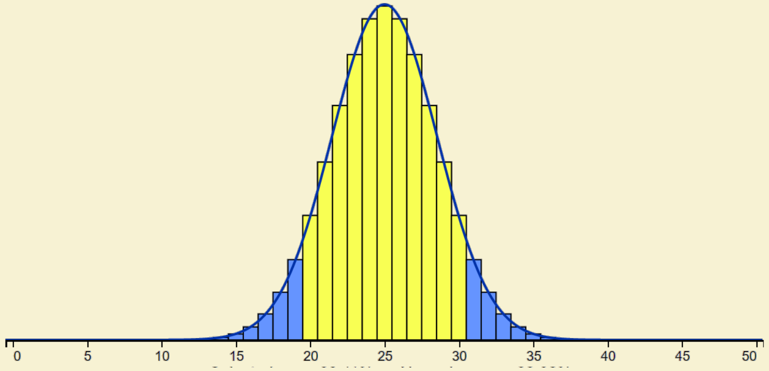
- Sample space



- Events



- Union/intersection/negation – AND/OR/NOT of events
- Random variables $X: \{0,1\}^n \to \mathbb{R}$
- Expectation Average value of X : $\mathbb{E}[X] = \sum_{x \in \{0,1\}^n} 2^{-n} X(x) = \sum_{v \in \mathbb{R}} v \cdot \Pr[X = v]$
- Concentration / tail bounds



Today:

- Randomized Algorithms
  - Polynomial Identity Testing
  - Approximation for maximum cut
- Randomized Complexity Class BPP
- Properties of randomized computation (Reducing error …)

Informal

A randomized algorithm has a special operation:

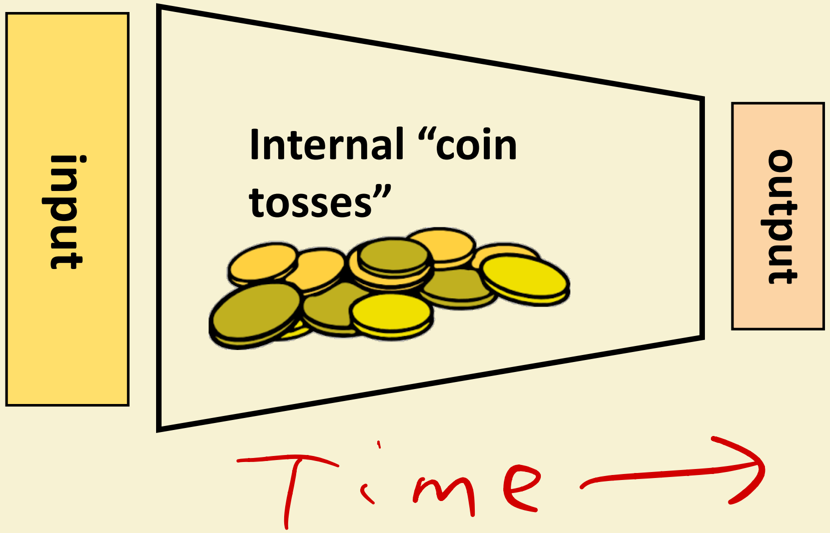
`foo` =

i.e. `foo` $\sim \{0,1\}$

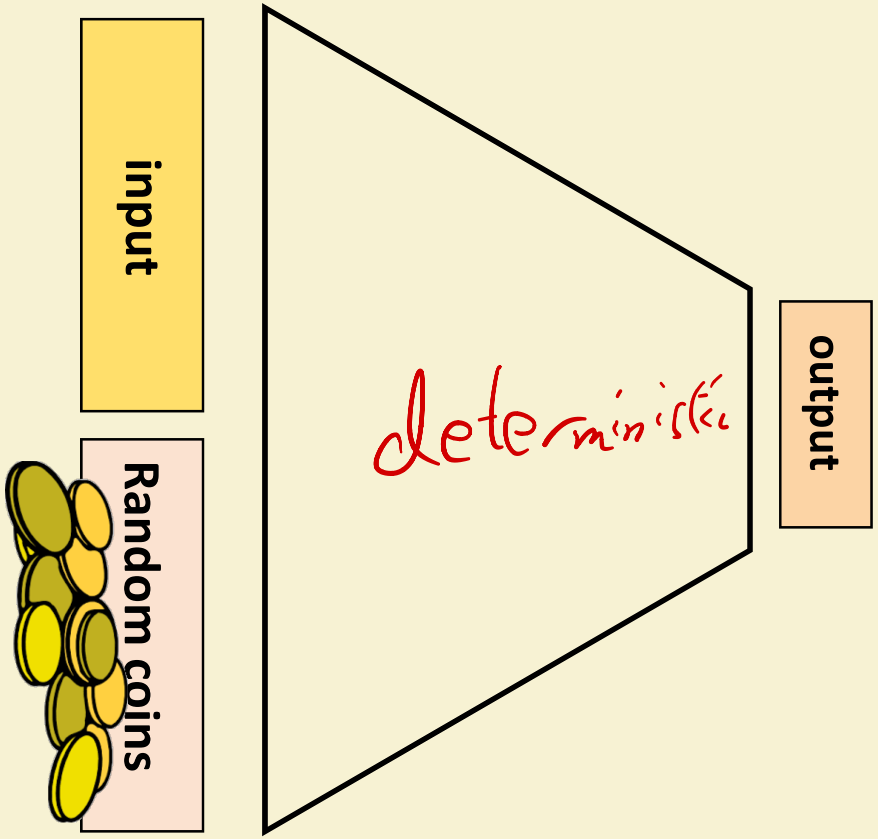
By repeating can choose `foo` $\sim \{0,1\}^n$ or $\sim [0,1]$

Randomized algorithms

Two equivalent views:



1. Get input $x \in \{0,1\}^n$
2. Run alg $A(x)$ that has special operation $r_i \leftarrow RAND(\,)$ ($r_i \sim \{0,1\}$)



1. Get input $x \in \{0,1\}^n$
2. Choose $r \sim \{0,1\}^m$
3. Run deterministic algorithm $A(x, r)$

$$output = ALG(input, randomness)$$

Computing a function

Not random input –
has to work in the worst case

Randomized algorithm ALG computes F if for every input x

$$\Pr[\,ALG(x) = F(x)] \geq \frac{2}{3}$$

Probability over the randomness of the algorithm, not the input

The constant $2/3$ is arbitrary – can be replaced by 0.51, 0.99, even $1 - 2^{-n}$. Not by $1/2$.

BPP: {Boolean functions computable by some randomized poly-time algorithm} $\supseteq P$

Polynomial Identity Testing: Problem

Q: $(x+yz)^7 - x^7 - y^7z^7 = 7x(x+yz)(x^2+y^2z^2)(x^2+xyz+y^2z^2)$?

Standard form: $(x+yz)(x+yz)(x+yz)(x+yz)(x+yz)(x+yz)(x+yz) - xxxxxxx - yyyyyyyzzzzzzz - 7x(x+yz)(xx+yyzz)(xx+xyz+yyzz) = 0$?

$x^2 + 2xyz + y^2z^2 \rightarrow$

Input φ: an expression like the above, with sums/products of variables.

Output $PIT(\varphi)$: 1 iff φ is the 0 polynomial.

Why is the following not a polynomial-time algorithm for PIT?

Alg-PIT(φ):

Multiply everything out,

Add/subtract like terms,

Return 1 iff all terms cancel.

Polynomial Identity Testing: Algorithm

$x = 3 \quad y = 0 \quad z = 121$

Q: $(x + yz)(x + yz)(x + yz)(x + yz)(x + yz)(x + yz)(x + yz) - xxxxxxx - yyyyyyyzzzzzzz - 7x(x + yz)(xx + yyzz)(xx + xyz + yyzz) = 0?$

$x - y \stackrel{?}{=} 0 \quad yz \quad 3^7 - 3^7 - 7 \cdot 3^6 \neq 0$

Randomized algorithm for PIT (note: polynomial time!):

RandAlg-PIT(φ):

For each variable, choose a random number between 0 and 3n.

Plug in those values and do all the integer arithmetic. ← poly time

Return 1 iff the result is 0.

Can give the wrong answer! Give an example.

(Give an input and the random choices.)

input ↑ polynomial; random choices ↑ inputs

•

Polynomial Identity Testing: Correctness (1/2)

Randomized algorithm for PIT:

RandAlg-PIT(φ):

For each variable, choose a random number between 0 and 3n.

Plug in those values and do all the integer arithmetic.

Return 1 iff the result is 0.

Goal: $\Pr_r[\, RandAlg - PIT(x, r) = PIT(x)] \geq \frac{2}{3}$ for all x

If $PIT(\varphi) = 1$, $\Pr[\, RandAlg - PIT(\varphi) = 1] = 1$ no matter what you plug in to 0 poly, get 0.

If $PIT(\varphi) = 0$…

Polynomial Identity Testing: Correctness (2/2)

$(x+yz)(x+yz)(x+yz)(x+yz)(x+yz)(x+yz)(x+yz) - xxxxxxx - yyyyyyyzzzzzzz = 7x(x+yz)(xx+yyzz)(xx+xyz+yyzz)$?

RandAlg-PIT(φ):
For each variable, choose a random number between 0 and 3n.
Plug in those values and do all the integer arithmetic.
Return 1 iff the result is 0.

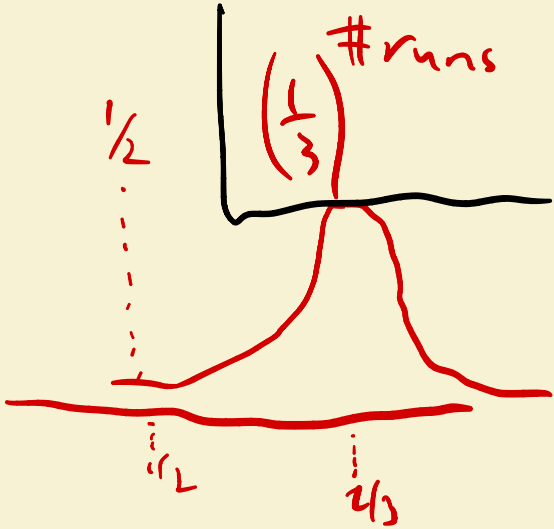
If $PIT(\varphi) = 0$: note that the <u>degree</u> is at most n.

Fact: A 1-variable polynomial $p \neq 0$ is 0 for $\leq \deg(p)$ inputs in $\{0, \ldots, 3n\}$

Fact: A k-variable polynomial $p \neq 0$ is 0 for $\leq \deg(p)(3n+1)^{k-1}$ inputs in $\{0, \ldots, 3n\}^k$

So $\Pr[\, RandAlg - PIT(\varphi) = 1] = \Pr[\varphi(x) = 0] \leq \frac{\deg(p)}{3n+1} < \frac{1}{3}$.

Success amplification



We have an algorithm RandAlg-PIT for which:

$$\Pr[RandAlg - PIT(\varphi) = PIT(\varphi)] \geq \frac{2}{3}$$

Give an algorithm BetterRandAlg-PIT for which:

$$\Pr[RandAlg - PIT(\varphi) = F(x)] \geq 1 - 2^{-60n}$$

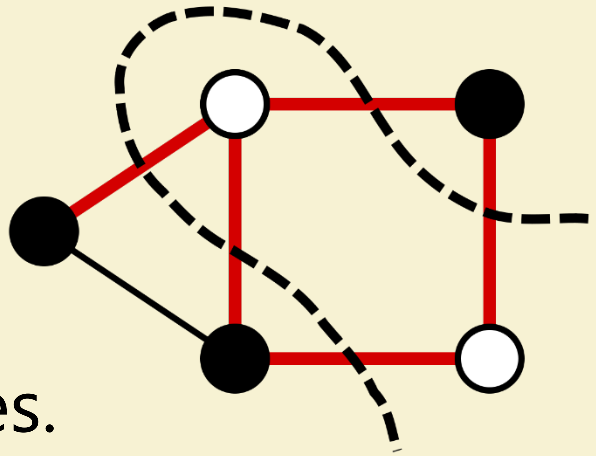
A: run the above $O(60n)$ times; return the majority answer.

Note: $\Pr[\,failure] < \Pr[\,asteroid\ hits\ us\ this\ minute]$

Bottom line: randomized algorithms as good as deterministic for all practical purposes.

Recall: randomized algorithms – work on worst case inputs.
Randomness is only over the coins of the algorithm.

MAXCUT



Input: Graph $G = (V, E)$.

Output: Partition of V maximizing # of crossing edges.

Define: $OPT(G) = \max_{S \subseteq V} |E(S, \overline{S})|$ to be max # of cut edges.

If $P \neq NP$, no poly-time alg computes $OPT(G)$ / produces cut achieving it.

We'll show: Poly-time randomized algorithm that w/ probability ≥ 0.99 outputs cut S that cuts at least $0.5 \cdot OPT(G)$ edges.

Note: not computing a function!

computing a relation

Best known: Alg cutting $\alpha \cdot OPT(G)$ edges for $\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \cdot \frac{\theta}{1 - \cos\theta} \approx 0.87857$

Central open question: is this optimal?

Input: Graph $G = (V, E)$.

Output: Partition of V maximizing # of crossing edges.

Define: $OPT(G) = \max_{S \subseteq V} |\mathrm{E}(S, \overline{S})|$ to be max # of cut edges.

We'll show: Poly-time randomized algorithm that w/ probability ≥ 0.99 outputs cut S that cuts at least $0.5 \cdot OPT(G)$ edges.

Thm: $\exists$ randomized poly time algorithm A s.t. with prob ≥ 0.99

$$A(G) = S \text{ s.t. } \left|E\left(S, \overline{S}\right)\right| \geq |E|/2$$

Q: Why does Thm imply what we need to show?

Thm: $\exists$ randomized poly time algorithm A s.t. with prob ≥ 0.99

$$A(G) = S \text{ s.t. } |E(S, \overline{S})| \geq |E|/2$$

Lemma: $\exists$ randomized poly time algorithm A s.t. if $S = A(G)$ then

$$\mathbb{E}[|E(S, \overline{S})|] \geq |E|/2$$

Over randomness of A

Runs of A:
.499 |E|
.999 |E|
.499 |E|
.499 |E|

Q: Why does Lemma not immediately imply the theorem?

Lemma: $\exists$ randomized poly time algorithm A s.t. if $S = A(G)$ then

$$\mathbb{E}\left[\left|E\left(S,\overline{S}\right)\right|\right] \geq |E|/2$$

Proof: Given G on n vertices, A picks $x \sim \{0,1\}^n$ and output $S = \{\, i \mid x_i = 1 \,\}$

For every edge $(i,j) \in E$, define $X_{i,j} = \begin{cases} 1, & x_i \neq x_j \\ 0, & x_i = x_j \end{cases}$

Q: What is $\mathbb{E}[X_{i,j}]$? A: $1/2$

Q: Prove that $\left|E\left(S,\overline{S}\right)\right| = \sum_{(i,j)\in E} X_{i,j}$

$\mathbb{E}\left[\sum_{(i,j)\in E} X_{i,j}\right] = \sum_{(i,j)\in E} \mathbb{E}\left[X_{i,j}\right] = \frac{|E|}{2}$

∎

From expectation to high probability

Success amplification

Given: Poly-time alg A s.t. that $\mathbb{E}\left[val(A(G))\right] \geq k$

Goal: Poly-time alg B s.t. that $\Pr\left[val\big(B(G)\big) \geq k\right] \geq 0.99$

k-1 w/p .999

|E| w/p .001

Algorithm B

Input: G

m: # of edges

m = |E|

for $i = 1 \ldots 1000m$:

  $S_i \leftarrow A(G)$ # *fresh randomness each time*

return S_i maximizing edges cut

Given: Poly-time alg A s.t. that $\mathbb{E}\,[val(A(G))] \geq k$

Goal: Poly-time alg B s.t. that $\Pr[val\big(B(G)\big) \geq k] \geq 0.99$

Algorithm B

Input: G

for $i = 1 \ldots 1000m$:

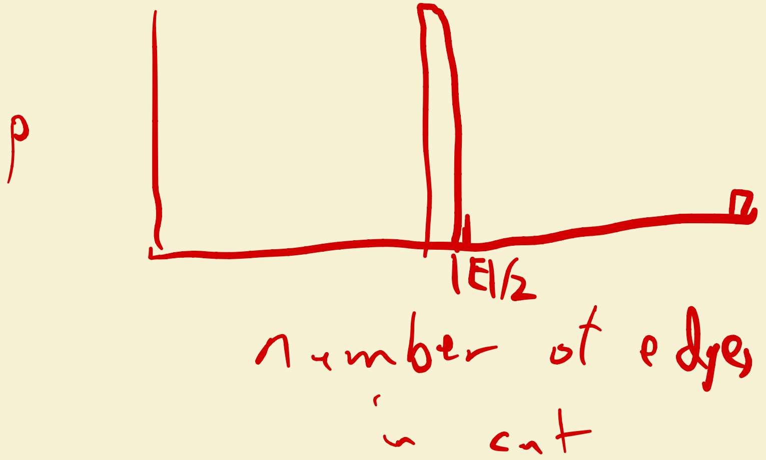
$\quad S_i \leftarrow A(G)$ # *fresh randomness each time*

return S_i maximizing edges cut

Lemma: $\Pr[val\big(A(G)\big) \geq k] \geq 1/m$

Q: Prove that Lemma $\Rightarrow \Pr[val\big(B(G)\big) \geq k] \geq 0.99$

Given: Poly-time alg A s.t. that $\mathbb{E}[val(A(G))] \geq k$



Lemma: $\Pr[val(A(G)) \geq k] \geq 1/m$

Proof: Suppose that $\Pr[\, val(A(G)) \geq k \,] < \frac{1}{m}$

then

prob $< 1/m$

val $\leq m$

prob ≤ 1

val $\leq k-1$

$\frac{|E|}{2} = k \leq \mathbb{E}[val(A(G)] < \frac{1}{m} \cdot m + 1 \cdot (k-1) = k$

$=$

$p[A(G) \geq k]\, E[val \text{ if } A(G) \geq k]$
$+ p[A(G) \leq k]\, E[val \text{ if } A(G) < k]$

Contribution from case that $val(A(G)) \geq k$

Contribution from case that $val(A(G)) < k-1$

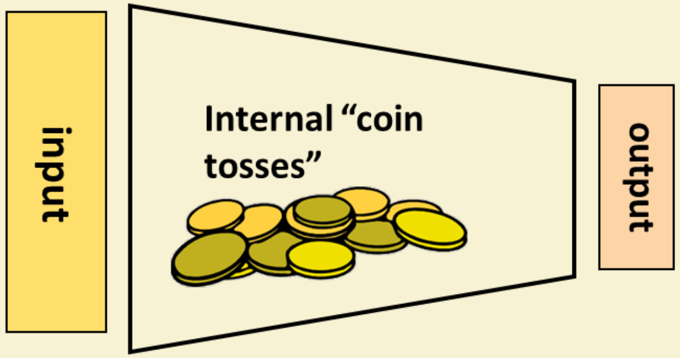
∎

Recap

Def: $F:\{0,1\}^* \rightarrow \{0,1\}$ is in BPP is there is a poly-time randomized algorithm A s.t. $\forall n\ \forall x \in \{0,1\}^n$

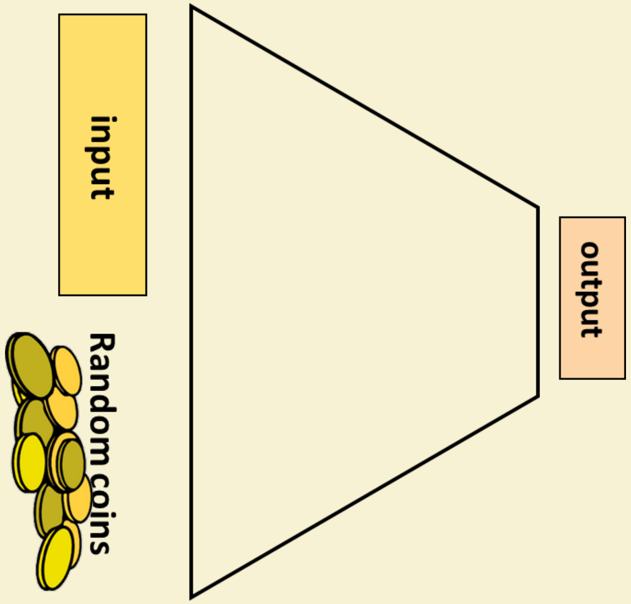
$$\Pr_{A's\ randomness}[A(x) = F(x)] \geq \frac{2}{3}$$

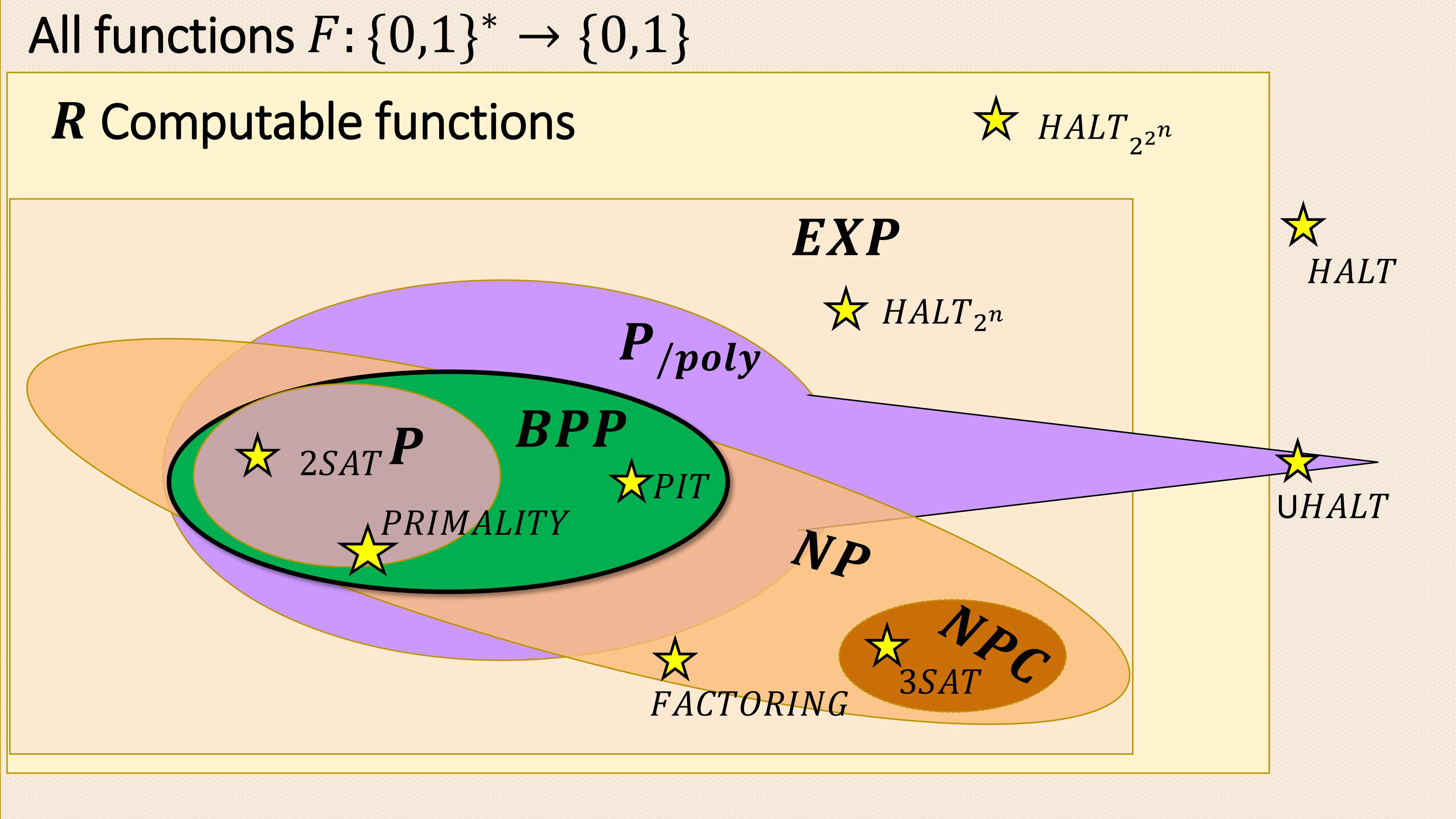
not over input; every input



Def: $F:\{0,1\}^* \rightarrow \{0,1\}$ is in BPP is there is a poly-time algorithm A , poly $q(n)$ s.t. $\forall n\ \forall x \in \{0,1\}^n$

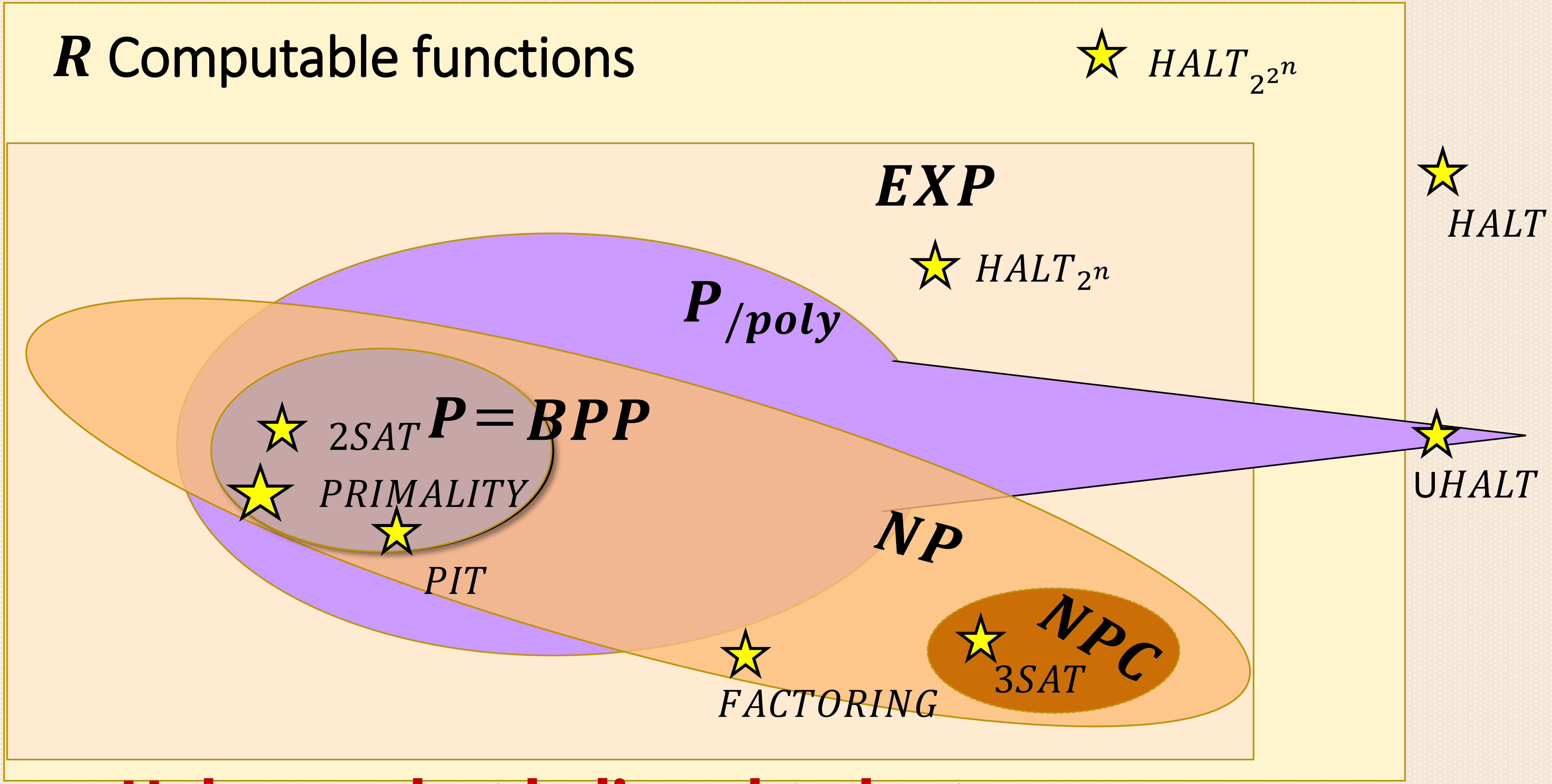
$$\Pr_{r\sim\{0,1\}^{q(n)}}[A(x;r) = F(x)] \geq \frac{2}{3}$$





All functions $F:\{0,1\}^* \to \{0,1\}$
$\boldsymbol{R}$ Computable functions
$HALT_{2^{2^n}}$
$\boldsymbol{EXP}$
$HALT$
$HALT_{2^n}$
$\boldsymbol{P}_{/poly}$
$\boldsymbol{BPP}$
$2SAT$
$\boldsymbol{P}$
PIT
UHALT
PRIMALITY
$\boldsymbol{NP}$
$\boldsymbol{NPC}$
$3SAT$
FACTORING

All functions $F: \{0,1\}^* \to \{0,1\}$



Unknown but believed to be true

Next Lecture

- BPP vs EXP
- BPP vs P/poly
- BPP vs NP