Lecture 23: Randomized Complexity
Practice Problems for Final Exam

Find them on Piazza!

Lots of problems to keep you busy... proofs, multiple choice, etc.

Solutions will come out in a few days

I strongly recommend trying to solve them before looking at the solutions!
TQBF = \{ \phi \mid \phi \text{ is a true quantified Boolean formula} \}

**Theorem:** TQBF is PSPACE-Complete

GG = \{ (G, a) \mid \text{Player 1 has a winning strategy for geography on graph G starting at node a} \}

**Theorem:** GG is PSPACE-Complete
Theorem [Hopcroft, Paul, and Valiant, ...]

\[ \text{TIME}(t(n)) \subseteq \text{SPACE}\left(\frac{t(n)}{\log t(n)}\right) \]

For every algorithm that runs in \( t(n) \) time, there is another algorithm with the same input/output behavior, using \( O(t(n)/\log t(n)) \) space (and exp time)

Holds for any reasonable algorithm model!

Amazing result! But we won’t prove it here...
**Theorem** [Hopcroft, Paul, and Valiant, ...]

\[
\text{TIME}(t(n)) \subseteq \text{SPACE}\left(\frac{t(n)}{\log t(n)}\right)
\]

**Corollary:** \(\text{SPACE}(n)\) is not in \(\text{TIME}(n \log \log n)\)

There are problems solvable in \(O(n)\) space that cannot be solved in \(O(n \log \log n)\) time.

A baby step towards \(P \neq \text{PSPACE}\).

**Proof Sketch:** If \(\text{SPACE}(n) \subseteq \text{TIME}(n \log \log n)\), then

\(\text{SPACE}(n^2) \subseteq \text{TIME}(n^2 \log \log n)\) by “padding”

By HPV theorem,

\(\text{TIME}(n^2 \log \log n) \subseteq \text{SPACE}((n^2 \log \log n)/ \log n)\)

This contradicts the Space Hierarchy Theorem!
For your favorite course on automata and complexity

Please complete the online subject evaluation for 6.045
Randomized / Probabilistic Complexity
A probabilistic TM $M$ is a nondeterministic TM where:

Each nondeterministic step is called a coin flip.

Each nondeterministic step has only two legal next moves (heads or tails).

The probability that $M$ runs on a branch $b$ is: $Pr[b] = 2^{-k}$

where $k$ is the number of coin flips that occur on branch $b$. 
Probabilistic/Randomized Algorithms

Why study randomized algorithms?

1. They can be **simpler** than deterministic algorithms

2. They can be **more efficient** than deterministic algorithms

3. **Can randomness be used to solve problems provably** much faster than deterministic algorithms?

*This is an open question!*
Pr [ M accepts w ] = \sum \Pr [ b ]

b is a branch on which M on w accepts

**Theorem:** A language A is in NP if there is a nondeterministic polynomial time TM M such that for all strings w:

- \( w \in A \Rightarrow \Pr[ M \text{ accepts } w ] > 0 \)
- \( w \notin A \Rightarrow \Pr[ M \text{ accepts } w ] = 0 \)
Theorem: A language \( A \) is in \( \text{coNP} \) if there is a nondeterministic polynomial time TM \( M \) such that for all strings \( w \):

\[
\begin{align*}
w \in A & \Rightarrow \Pr[ M \text{ accepts } w ] = 0 \\
w \notin A & \Rightarrow \Pr[ M \text{ accepts } w ] > 0
\end{align*}
\]

Theorem: A language \( A \) is in \( \text{NP} \) if there is a nondeterministic polynomial time TM \( M \) such that for all strings \( w \):

\[
\begin{align*}
w \in A & \Rightarrow \Pr[ M \text{ accepts } w ] > 0 \\
w \notin A & \Rightarrow \Pr[ M \text{ accepts } w ] = 0
\end{align*}
\]
Definition. A probabilistic TM $M$ decides a language $A$ with error $\varepsilon$ if for all strings $w$,

- $w \in A \implies \Pr [ M \text{ accepts } w ] \geq 1 - \varepsilon$
- $w \notin A \implies \Pr [ M \text{ doesn't accept } w ] \geq 1 - \varepsilon$
**Error Reduction Lemma**

**Lemma:** Let $\varepsilon$ be a constant, $0 < \varepsilon < 1/2$, let $k \in \mathbb{N}$. If $M_1$ has error $1/2 - \varepsilon$ and runs in $t(n)$ time then there is an equivalent machine $M_2$ such that $M_2$ has error $< 1/2^{n^k}$ and runs in $O(n^k \cdot t(n)/\varepsilon^2)$ time.

**Proof Idea:**

On input $w$, $M_2$ runs $M_1$ on $w$ for $m = 10 \frac{n^k}{\varepsilon^2}$ random independent trials, records the $m$ answers of $M_1$ on $w$, returns most popular answer (accept or reject).

Can use Chernoff Bound to show the error is $< 1/2^{n^k}$

Probability that the Majority answer over $10m/\varepsilon^2$ trials is different from the $1/2 + \varepsilon$ prob event is $< 1/2^m$. 
**Error Reduction Lemma**

Lemma: Let \( \varepsilon \) be a constant, \( 0 < \varepsilon < 1/2 \), let \( k \in \mathbb{N} \). If \( M_1 \) has error \( 1/2 - \varepsilon \) and runs in \( t(n) \) time then there is an equivalent machine \( M_2 \) such that

- \( M_2 \) has error \( < 1/2^{n^k} \) and runs in \( O(n^k \cdot t(n)/\varepsilon^2) \) time

Proof Idea:

On input \( w \), \( M_2 \) runs \( M_1 \) on \( w \) for \( m = 10 \cdot n^k/\varepsilon^2 \) random independent trials, records the \( m \) answers of \( M_1 \) on \( w \), returns most popular answer (accept or reject)

Define indicator \( X_i = 1 \) iff \( M_1 \) outputs right answer in trial \( i \)
Set \( X = \sum_i X_i \). Then \( E[X] = \sum_i E[X_i] \geq (1/2 + \varepsilon)m \)

Show: \( \Pr[M_2 \text{ outputs wrong}] = \Pr[X < m/2] < 1/2 \varepsilon^2 m/10 \)
BPP = Bounded Probabilistic P

BPP = \{ L \mid L \text{ is recognized by a probabilistic polynomial-time TM with error at most } \frac{1}{3} \}

Why \frac{1}{3}?

It doesn’t matter what error value we pick, as long as the error is smaller than \frac{1}{2} - \frac{1}{n^k} for some constant k

When the error is smaller than \frac{1}{2}, we can apply the error reduction lemma and get \frac{1}{2^{nc}} error
CHECK = \{ (M_1, M_2, N) \mid M_1, M_2 \text{ and } N \text{ are matrices and } M_1 \cdot M_2 = N \}

If \( M_1 \) and \( M_2 \) are \( n \times n \) matrices, computing \( M_1 \cdot M_2 \) takes \( O(n^3) \) time normally, and \( O(n^{2.373}) \) time using very sophisticated methods.

Here is an \( O(n^2) \)-time randomized algorithm for CHECK:

Pick a 0-1 bit vector \( r \) at random, test if \( M_1 \cdot M_2 r = N r \)

Claim: If \( M_1 \cdot M_2 = N \), then \( \Pr [M_1 \cdot M_2 r = N r] = 1 \)

If \( M_1 \cdot M_2 \neq N \), then \( \Pr [M_1 \cdot M_2 r = N r] \leq 1/2 \)

WHY?
Checking Matrix Multiplication

CHECK = \{ (M_1, M_2, N) \mid M_1, M_2 \text{ and } N \text{ are matrices and } M_1 \cdot M_2 = N \} 

If M_1 and M_2 are n x n matrices, computing M_1 \cdot M_2 takes \( O(n^3) \) time normally, and \( O(n^{2.373}) \) time using very sophisticated methods.

Here is an \( O(n^2) \)-time randomized algorithm for CHECK:

Pick a 0-1 bit vector \( r \) at random, test if \( M_1 \cdot M_2 r = Nr \)

Claim: If \( M_1 \cdot M_2 = N \), then \( \Pr [M_1 \cdot M_2 r = Nr ] = 1 \)
If \( M_1 \cdot M_2 \neq N \), then \( \Pr [M_1 \cdot M_2 r = Nr ] \leq 1/2 \)

If we pick 20 random vectors and test them all, what is the probability of incorrect output? \( 1/2^{20} < 0.000001 \)
Checking Matrix Multiplication

CHECK = \{ (M_1, M_2, N) \mid M_1, M_2 and N are matrices and M_1 \cdot M_2 = N \} 

Pick a 0-1 bit vector \( r \) at *random*, test if \( M_1 \cdot M_2 r = Nr \)

**Claim:** If \( M_1 \cdot M_2 \neq N \), then \( \Pr [M_1 \cdot M_2 r = Nr] \leq 1/2 \)

**Proof:** Define \( M' = N - (M_1 \cdot M_2) \). \( M' \) is a non-zero matrix. Some row \( M'_i \) is non-zero, some entry \( M'_{i,j} \) is non-zero.

**Want:** \( \Pr[M'r = \vec{0}] \leq 1/2 \)

We have: \( \Pr[M'r = \vec{0}] \leq \Pr[\langle M'_i, r \rangle = 0] \)

\[ = \Pr[\sum_k M'_{i,k} \cdot r_k = 0] \text{ (def of inner product)} \]

\[ = \Pr[r_k = (\sum_{k \neq j} M'_{i,k} \cdot r_k)/M'_{i,j}] \leq 1/2 \]

Why \( \leq 1/2 \)? After everything else is assigned on RHS, there is at most one value of \( r_k \) that satisfies the equation!
An arithmetic formula is like a Boolean formula, except it has $+, -, \text{ and } \ast$ instead of \textbf{OR, NOT, AND}.

$\text{ZERO-POLY} = \{ p \mid p \text{ is an arithmetic formula that is } \text{identically zero} \}$

Identically zero means: all coefficients are 0

Two examples of polynomials in \text{ZERO-POLY}:

$(x + y)\cdot(x + y) - x\cdot x - y\cdot y - 2\cdot x\cdot y$

Abbreviate as: $(x + y)^2 - x^2 - y^2 - 2xy$

$(x^2 + a^2)\cdot(y^2 + b^2) - (x\cdot y - a\cdot b)^2 - (x\cdot b + a\cdot y)^2$

There is a rich history of polynomial identities in mathematics. Useful also in program testing!
Testing Univariate Polynomials

Let \( p(x) \) be a polynomial in one variable over \( \mathbb{Z} \)

\[
p(x) = a_0 + a_1x + a_2x^2 + \ldots + a_dx^d
\]

Suppose \( p \) is hidden in a black box – we can only see its inputs and outputs.

Want to determine if \( p \) is \textit{identically} 0

Simply evaluate \( p \) on \( d+1 \) distinct values!

Non-zero degree \( d \) polynomials have \( \leq d \) roots.

But the \textit{zero polynomial} has every value as a root.
Testing Multivariate Polynomials

Let $p(x_1,\ldots,x_n)$ be a polynomial in $n$ variables over $\mathbb{Z}$.

Suppose $p(x_1,\ldots,x_n)$ is given to us, but as a very complicated arithmetic formula. Can we efficiently determine if $p$ is identically 0?

If $p(x_1,\ldots,x_n)$ is a product of $m$ polynomials, each of which is a polynomial in $t$ terms, $\prod_m (\sum_t stuff)$

Then expanding the expression into a sum of products could take $t^m$ time!

Big Idea: Evaluate $p$ on random values
Theorem (Schwartz-Zippel-DeMillo-Lipton)
Let $p(x_1, x_2, \ldots, x_n)$ be a \textit{nonzero} polynomial, where each $x_i$ has \textit{degree at most} $d$. Let $F \subset \mathbb{Z}$ be finite.

If $a_1, \ldots, a_m$ are selected randomly from $F$, then:

$$\Pr [ p(a_1, \ldots, a_m) = 0 ] \leq dn/|F|$$

Proof (by induction on $n$):

Base Case ($n = 1$):

$$\Pr [ p(a_1) = 0 ] \leq d/|F|$$

Nonzero polynomials of degree $d$ have most $d$ roots, so at most $d$ elements in $F$ can make $p$ zero
Inductive Step \((n > 1)\): Assume true for \(n-1\) and prove for \(n\)

Let \(p(x_1,\ldots,x_n)\) be not identically zero.

Write: \(p(x_1,\ldots,x_n) = p_0 + x_n p_1 + x_n^2 p_2 + \ldots + x_n^d p_d\)

where \(x_n\) does not occur in any \(p_i(x_1,\ldots,x_{n-1})\)

Observe: At least one \(p_i\) is not identically zero

Suppose \(p(a_1,\ldots,a_n) = 0\). Let \(q(x_n) = p(a_1,\ldots,a_{n-1},x_n)\). Two cases:

1. \(q \equiv 0\). That is, for all \(j\), \(p_j(a_1,\ldots,a_{n-1}) = 0\) (including \(p_i\))

\[
\Pr[(1)] \leq \Pr[p_i(a_1,\ldots,a_{n-1}) = 0] \leq (n-1)d/|F| \text{ by induction}
\]

2. \(q\) is not identically zero, but \(q(a_n) = 0\).

Note \(q\) is a univariate degree-\(d\) polynomial!

\[
\Pr[(2)] \leq \Pr[q(a_n) = 0] \leq d/|F| \text{ by univariate case}
\]

\[
\Pr[(1) \text{ or } (2)] \leq \Pr[(1)] + \Pr[(2)] \leq nd/|F|
\]
ZERO-POLY = \{ p \mid p \text{ is an arithmetic formula that is identically zero}\}

Theorem: ZERO-POLY ∈ BPP

Proof: Suppose \( n = |p| \). Then \( p \) has \( k \leq n \) variables, and the degree of each variable is at most \( n \).

Algorithm A: Given polynomial \( p \),

For all \( i = 1, \ldots, k \), choose \( r_i \) randomly from \( \{1, \ldots, 3n^2\} \)

If \( p(r_1, \ldots, r_k) = 0 \) then output \text{zero}

else output \text{nonzero}

Observe A runs in polynomial time.

If \( p \equiv 0 \), then \( \Pr[A(p) \text{ outputs zero}] = 1 \)

If \( p \not\equiv 0 \), then by the Schwartz-Zippel lemma,

\[
\Pr[A(p) \text{ outputs zero}] = \Pr_r[p(r) = 0] \leq \frac{n^2}{3n^2} \leq \frac{1}{3}
\]
Checking Equivalence of Arithmetic Formulas

**Theorem:** \( \text{ZERO-POLY} \in \text{BPP} \)

\( \text{ZERO-POLY} = \{ p \mid p \text{ is an arithmetic formula that is identically zero} \} \)

**Corollary:** \( \text{EQUIV-POLY} \in \text{BPP} \)

\( \text{EQUIV-POLY} = \{ (p,q) \mid p \text{ and } q \text{ are arithmetic formulas computing the same polynomial} \} \)

**Proof:** \((p,q) \in \text{EQUIV-POLY} \iff p-q \in \text{ZERO-POLY} \)

Therefore \( \text{EQUIV-POLY} \leq_p \text{ZERO-POLY} \) and we get a BPP algorithm for \( \text{EQUIV-POLY} \).

See Sipser 10.2 for an application to testing equivalence of simple programs!
Equivalence of Arithmetic Formulas

EQUIV-POLY = \{ (p,q) \mid p \text{ and } q \text{ are arithmetic formulas computing the same polynomial} \}

Corollary: EQUIV-POLY \in BPP

There is a big contrast with Boolean formulas!

EQUIV = \{ (\phi,\psi) \mid \phi \text{ and } \psi \text{ are Boolean formulas computing the same function} \}

We showed EQUIV is in coNP. It’s also coNP-complete!

TAUTOLOGY \leq_p EQUIV: map \phi \text{ to } (\phi, T)
Theorem: \text{ZERO-POLY} \in \text{BPP}

\text{ZERO-POLY} = \{ p \mid p \text{ is an arithmetic formula that is identically zero} \}

It is not known how to solve \text{ZERO-POLY} efficiently \textit{without} randomness!

Thm [KI’04, AvM’11] IF \text{ZERO-POLY} \in \text{P} THEN NEW \text{LOWER BOUNDS FOLLOW} (not \text{P} \neq \text{NP}, but still a breakthrough!)
BPP = \{ L \mid L \text{ is recognized by a probabilistic polynomial-time TM with error at most } \frac{1}{3} \}

Is BPP \subseteq NP?

THIS IS AN OPEN QUESTION!
Is BPP $\subseteq$ PSPACE?

Yes! Run through all possible sequences of coin flips one at a time, and count the number of branches that accept.

Known: BPP $\subseteq$ NP$^{|NP|}$ and BPP $\subseteq$ coNP$^{|NP|}$, but BPP $\subseteq$ P$^{|NP|}$ is still open!
Is $\text{NP} \subseteq \text{BPP}$?

THIS IS AN OPEN QUESTION!