香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

SCHOOL OF DATA SCIENCE
數據科學學院

# CSC 6011 - Theory of Computation

# Lecture 19: Probabilistic Computation, BPP

**Time:** Mon & Wed, 4:00 - 5:20 PM

**Location:** Teaching A, 107

**Instructor:** Tao LIN (林涛)

**TA:** Yefan GAO (高叶繁)

Fall 2026

# Teaching plan

| Lecture | Topic |
|---|---|
| | **Computability Theory** |
| 1–2 | Introduction. Finite automata (DFA and NFA). Regular languages. |
| 3–4 | Nonregular languages, pumping lemma. Context-free languages, pushdown automata. |
| 5–6 | Turing machines, Church-Turing thesis. |
| 7–8 | Decidability, diagonalization. |
| 9 | Reduction. |
| | **Complexity Theory - Part 1** |
| 10 | Time complexity. |
| 11–12 | P and NP, NP-completeness. |
| 13 | Cook-Levin theorem. |
| 14 | Co-NP. Optimization problems. |

| Lecture | Topic |
|---|---|
| | **Complexity Theory - Part 2** |
| 15–16 | **Space complexity**, PSPACE, Savitch's theorem. |
| 17 | **[Midterm Exam] (Nov 9).** |
| 18 | L, NL. |
| | **Other Topics** |
| **19–20** | **Probabilistic (randomized) computation, BPP.** |
| 21–22 | Interactive proofs, IP=PSPACE. |
| 23–24 | Formal language and modern AI models. |
| 25–26 | Approximation algorithms, hardness of approximation, PCP theorem. |
| 27–28 | Communication complexity. |

# Algorithmic Ideas

Important algorithmic ideas seen so far:

- Dynamic Programming (Lecture 11: $A_{\text{CFG}} \in \text{P}$ ).
- Bisection and Recursion (Lecture 16: $\text{LADDER}_{\text{DFA}}$ and Savitch's Theorem).
- Randomization

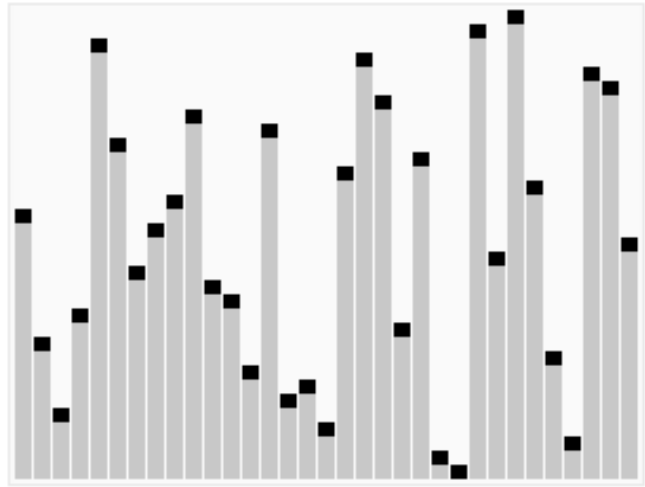
**Examples of randomized algorithms:**

(1) Quicksort:

- Pick an element ("pivot") in an array;
- Put smaller elements before the pivot and larger elements after it;
- Recurse.

Bad choices of pivot can cause $\Omega(n^2)$ running time;

Picking the pivot randomly ensures an $O(n \log n)$ expected running time.



Source: Wikipedia

(2) Primality test (Miller-Rabin algorithm).

(3) Cryptography (secret key generation).

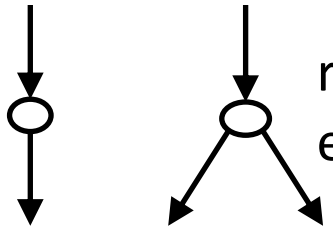
(4) Stochastic gradient descent.

……

# Probabilistic TMs

**Defn:** A probabilistic Turing machine (PTM) is a variant of a NTM where each computation step is either a deterministic step or a random step (with 2 choices).

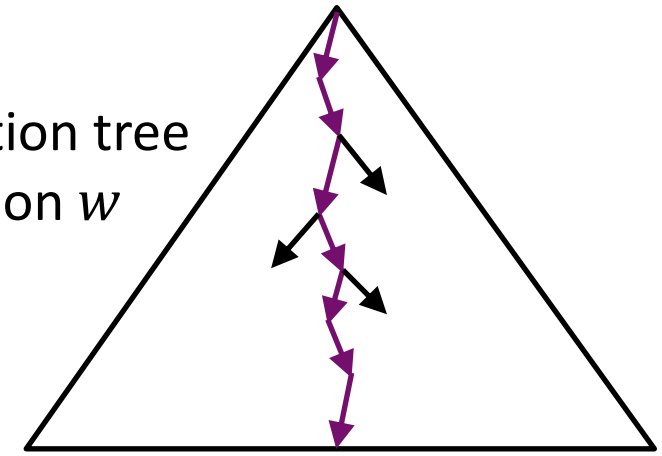
deterministic step





random step - each choice has 50% probability

computation tree for $M$ on $w$



branch $b$: accept/reject

Pr[ branch $b$ ] $= 2^{-k}$ if $b$ has $k$ random steps

$$\Pr[\ M \text{ accepts } w\ ] = \sum_{b \text{ is an accepting branch}} \Pr[\ \text{branch } b\ ]$$

Pr[ $M$ rejects $w$ ] $= 1 -$ Pr[ $M$ accepts $w$ ]

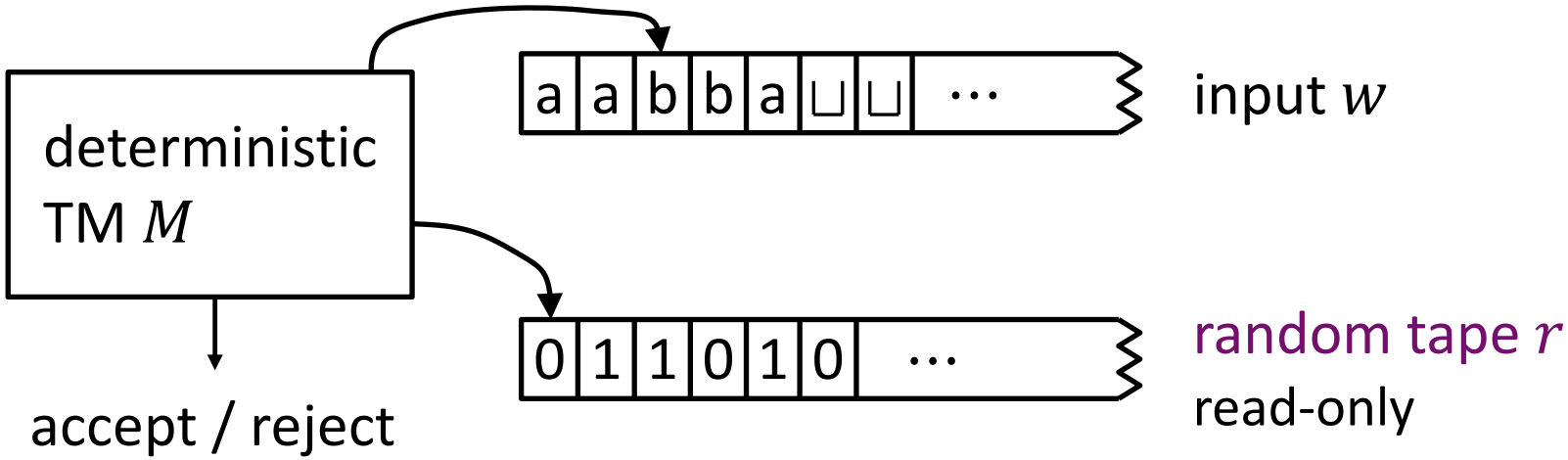
**Defn:** For $\varepsilon \geq 0$, say PTM $M$ decides language $A$ with error probability $\varepsilon$ if for every $w$, Pr[ $M$ gives the wrong answer about $w \in A$ ] $\leq \varepsilon$

*i.e.*, $w \in A$ ("YES" instance) $\rightarrow$ Pr[ $M$ accepts $w$ ] $\geq 1 - \varepsilon$

$w \notin A$ ("NO" instance) $\rightarrow$ Pr[ $M$ rejects $w$ ] $\geq 1 - \varepsilon$.

# Equivalent Definition: DTM with Random Tape

**Defn:** A PTM is equivalent to a deterministic TM with an extra read-only random tape containing independent, uniformly random bits.




Read one bit at each random step.

Fixing $w$ and $r$, the computation $M(w, r)$ becomes deterministic.

A DTM with a random tape $r$ generates the same distribution over outcomes as the PTM.

# The Class BPP (Bounded-error Probabilistic Polynomial Time)

**Defn:** BPP $= \{ A \mid$ some poly-time PTM decides $A$ with error probability $\varepsilon \leq 1/3 \}$

**Amplification lemma:** If $M_1$ is a poly-time PTM with error probability $\varepsilon_1 < 1/2$, then, for any $0 < \varepsilon_2 < 1/2$, there is a poly-time PTM $M_2$ with error probability $\varepsilon_2$.

Can strengthen to make $\varepsilon_2 < 2^{-\text{poly}(n)}$.

**Proof idea:** $M_2 =$ "On input $w$,

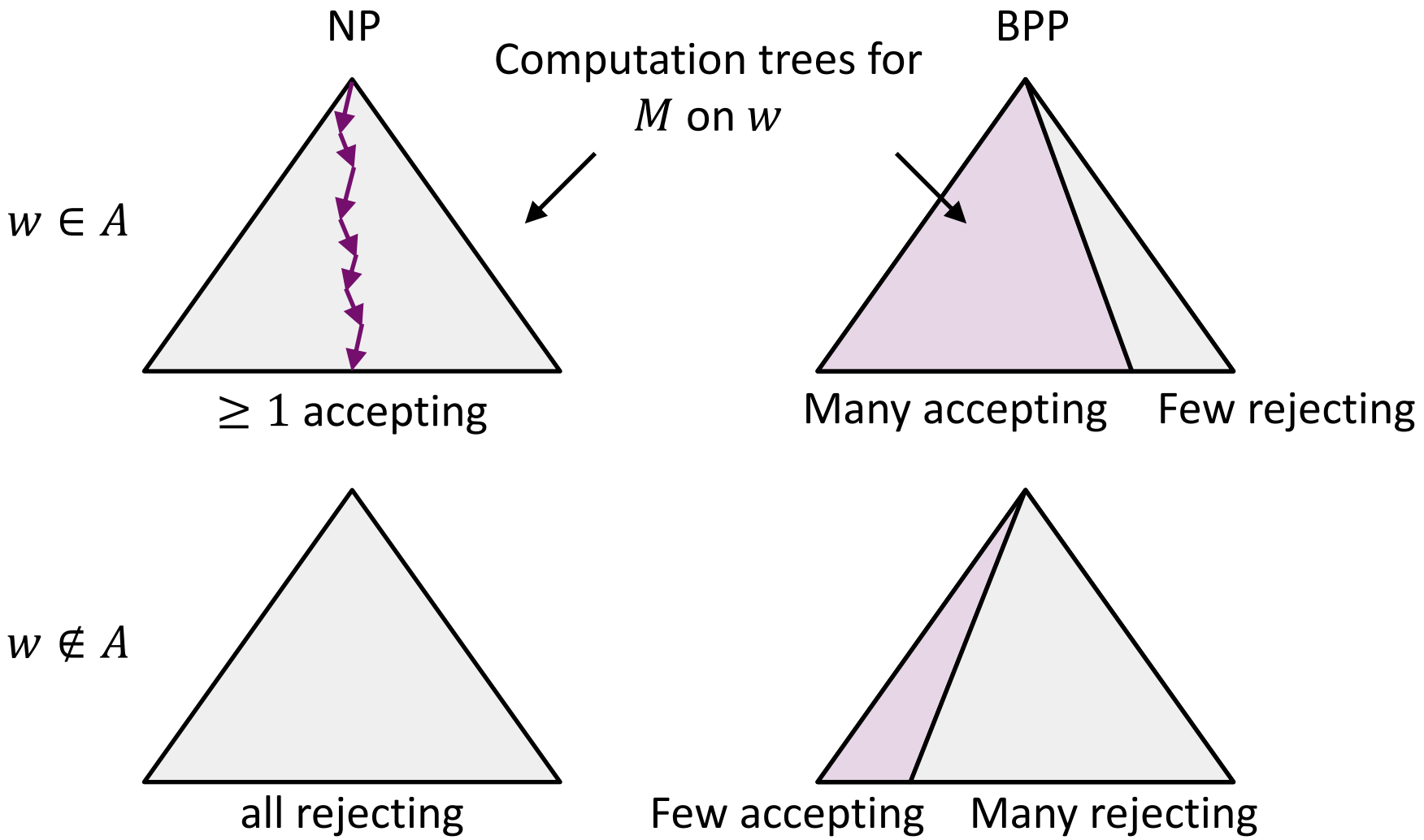
Run $M_1$ on $w$ for $k$ times (independently) and output the majority answer."

By the law of large numbers, the majority answer should be correct with high probability.

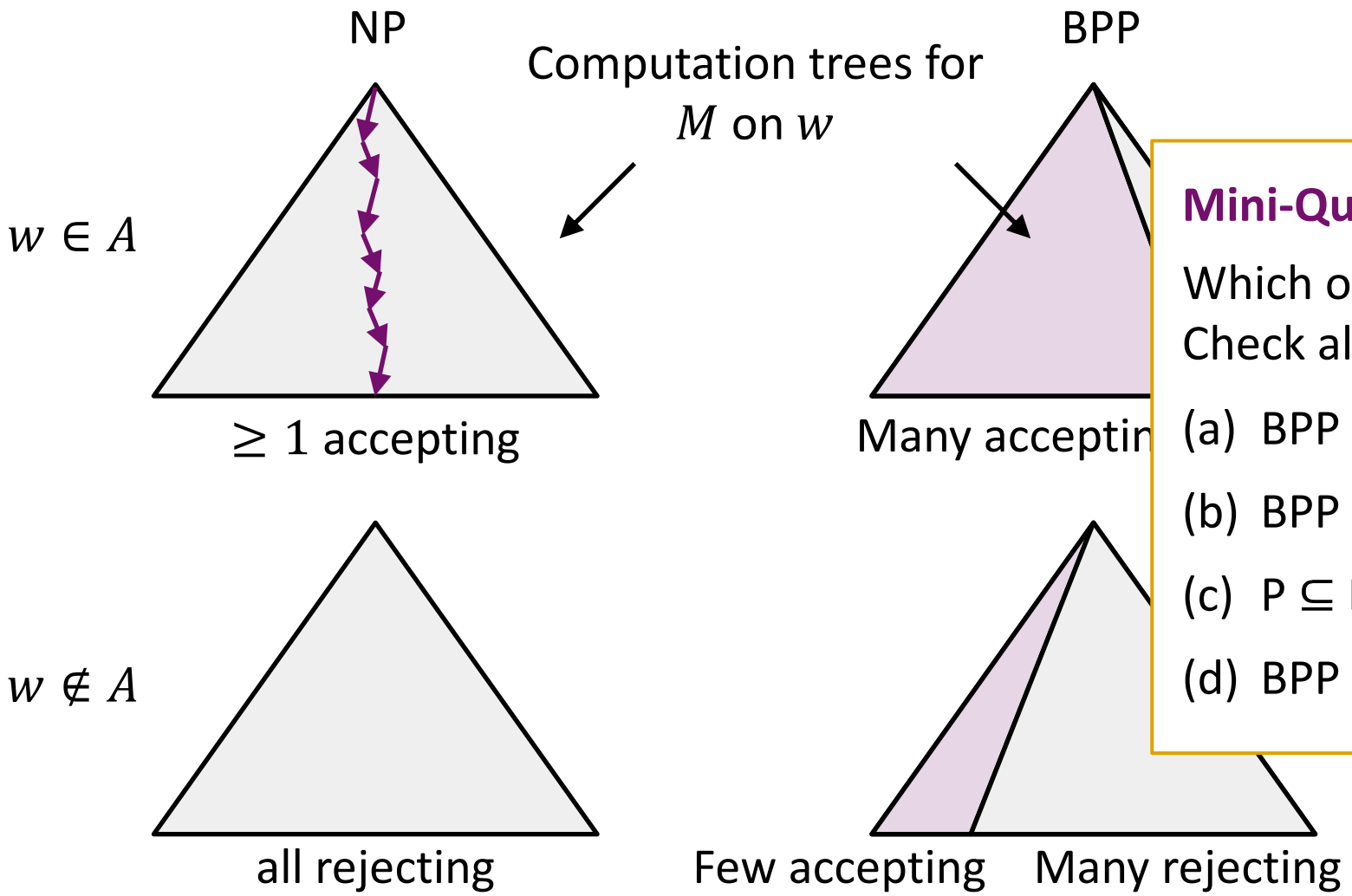
**Details:** Homework 4.

**Therefore,** BPP can be equivalently defined using any error probability that is a fixed constant $0 < \varepsilon < 1/2$ or exponentially small.

# NP and BPP


NP
Computation trees for
$M$ on $w$
BPP
$w \in A$
$\geq 1$ accepting
Many accepting
Few rejecting
$w \notin A$
all rejecting
Few accepting
Many rejecting

# NP and BPP




**Mini-Quiz 19.1**

Which of the following are true?
Check all that apply.

(a) BPP is closed under union.

(b) BPP is closed under complement.

(c) P $\subseteq$ BPP

(d) BPP $\subseteq$ PSPACE

# BPP ⊆ PSPACE

**Theorem:** BPP ⊆ PSPACE

**Proof:** Let $A \in$ BPP be decided by a PTM $M$ with polynomial running time $t(n)$.
$M$ uses at most $t(n)$ bits on the random tape.

Simulate $M$ by a PSPACE TM $N$:

$N =$ "On input $w$,

1. Enumerate all $r \in \{0,1\}^{t(n)}$.
2. Simulate $M(w, r)$ and count the accepting runs.
3. Accept if more than half of the runs accept. "

**Space needed:** storing $r$ and simulating $M(w, r)$ both take polynomial space.

**Correctness:**
For $w \in A$, at least 2/3 of the runs accept, so $N$ accepts $w$;
For $w \notin A$, at most 1/3 of the runs accept, so $N$ rejects $w$.

# Polynomial Identity Testing

**Input:** Two arithmetic formulas $p_1, p_2$ with variables $x_1, \ldots, x_m$ and integer coefficients, combined by operations $+, -, \times$.

**Example:** $p_1(x, y) = (x + y)(x - y)$ and $p_2(x, y) = x^2 - y^2$.

**Problem:** Are polynomials $p_1$ and $p_2$ identical (denoted by $p_1 \equiv p_2$)?

$$\mathrm{EQ}_{\mathrm{poly}} = \{ \langle p_1, p_2 \rangle \mid p_1(x_1, \ldots, x_m) = p_2(x_1, \ldots, x_m) \ \forall x_1, \ldots, x_m \in \mathbb{R} \}$$

(Equivalently, “identical" means that the coefficients of $p_1$ and $p_2$ agree after expansion.)

A short formula such as $(x_1 + 1)(x_2 + 1) \cdots (x_m + 1)$ has $2^m$ terms after expansion.

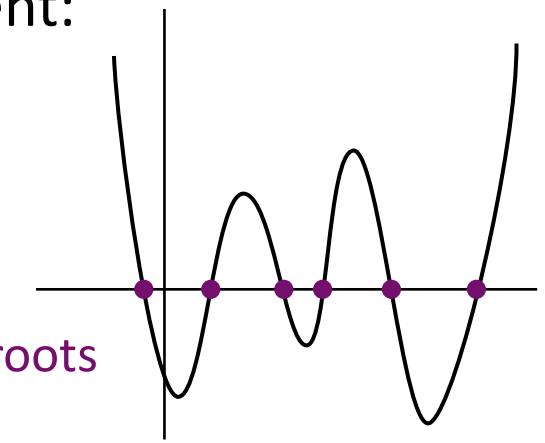
Can we test polynomial identity without expanding the formulas?

**Equivalent problem:** Testing if $p_1 - p_2 \equiv 0$.

# Background on algebra

Let $p(x) = a_0 + a_1 x + \cdots + a_d x^d$ be a single-variable polynomial.

- The degree of $p$ is the highest exponent of $x^i$ with a non-zero coefficient:
  - the degree is $d$ if $a_d \neq 0$
- A $z$ satisfying $p(z) = 0$ is called a root of $p$.




**Fundamental Theorem of Algebra:**
A non-zero single-variable polynomial $p$ of degree $\leq d$ has $\leq d$ roots.

**Finite field:** The above holds for any field $\mathbb{F}$ (a field is a set with $+, -, \times, \div$ operations).
We will use the finite field $\mathbb{F}_q = \{0, 1, \ldots, q-1\}$ with a prime $q$ and operations modulo $q$.

Example in $\mathbb{F}_7$: $5 + 4 = 2 \pmod 7$ $3 \times 5 = 1 \pmod 7$ $3^{-1} = 5 \pmod 7$

**Corollary 1:** If $p \not\equiv 0$ has degree $\leq d$ and we pick a random $r \in \mathbb{F}_q$, then

$$\Pr[\, p(r) = 0 \,] \leq \frac{d}{q}.$$

**Proof:** There are at most $d$ roots out of $q$ possibilities.

# The Schwartz–Zippel Lemma

**Schwartz-Zippel Lemma**: If $p(x_1, \ldots, x_m) \not\equiv 0$ has degree $\leq d$ for each $x_i$ and we pick random $r_1, \ldots, r_m \in \mathbb{F}_q$ (uniformly and independently), then

$$\Pr[\, p(r_1, \ldots, r_m) = 0 \,] \leq \frac{md}{q}$$

**Proof** (induction on $m$): For $m = 1$: use Corollary 1.

For $m \geq 2$, write $x' = (x_1, \ldots, x_{m-1})$ and

$$p(x', x_m) = a_0(x') + a_1(x')x_m + \cdots + a_k(x')x_m^k, \quad \text{where } a_k \not\equiv 0 \text{ and } k \leq d.$$

By induction, $\Pr[a_k(r') = 0] \leq \frac{(m-1)d}{q}$. Let $E$ be the event that $a_k(r') = 0$

If $a_k(r') \neq 0$, then $p(r', x_m)$ is a nonzero polynomial of $x_m$ of degree $k \leq d$ and has at most $d$ roots.

Again by Corollary 1, $\Pr[\, p(r', r_m) = 0 \mid a_k(r') \neq 0] \leq \frac{d}{q}$.

$$\begin{aligned}
\Pr[p(r) = 0] &= \Pr[\, p(r) = 0 \mid E \,] \cdot \Pr[E] + \Pr[\, p(r) = 0 \mid \overline{E} \,] \cdot \Pr[\overline{E}] \\
&\leq 1 \cdot \Pr[E] + \Pr[\, p(r) = 0 \mid \overline{E} \,] \cdot 1 \\
&\leq \frac{(m-1)d}{q} + \frac{d}{q} = \frac{md}{q}
\end{aligned}$$

# A BPP algorithm for polynomial identity testing

Let $p_1, p_2$ be polynomials with integer coefficients and degree $\leq d$ for each of $m$ variables.
Let $B$ be an upper bound on the absolute value of every coefficient (after expansion).

1. Choose a prime $q > \max\{3md, 2B\}$.
2. Pick $r_1, \ldots, r_m$ independently and uniformly from $\mathbb{F}_q = \{0, 1, \ldots, q-1\}$.
3. Evaluate $p_1(r)$ and $p_2(r)$ modulo $q$, without expanding the formulas.
4. If $p_1(r) = p_2(r)$, *accept*. Otherwise, *reject*.

**Error probability:**

If $p_1 \equiv p_2$, the algorithm always accepts.

If $p_1 \not\equiv p_2$, some coefficient $c$ of $p = p_1 - p_2$ satisfies $0 < |c| \leq 2B < q$.
So $p \not\equiv 0$ over $\mathbb{F}_q$. By Schwartz–Zippel lemma,

$$\Pr[p_1(r) = p_2(r) \pmod q] \leq \frac{md}{q} < \frac{1}{3}.$$

**Running time:** Polynomial in the length of $p_1, p_2$ (before expansion) and $\log q$.

**Conclusion:** $\mathrm{EQ}_{\mathrm{poly}} \in$ BPP.

# A BPP algorithm for polynomial identity testing

Let $p_1, p_2$ be polynomials with integer coefficients and degree $\leq d$ for each of $m$ variables.
Let $B$ be an upper bound on the absolute value of every coefficient (after expansion).

1. Choose a prime $q > \max\{3md, 2B\}$.
2. Pick $r_1, \ldots, r_m$ independently and uniformly from
3. Evaluate $p_1(r)$ and $p_2(r)$ modulo $q$, without exp
4. If $p_1(r) = p_2(r)$, *accept*. Otherwise, *reject*.

**Error probability:**

If $p_1 \equiv p_2$, the algorithm always accepts.

If $p_1 \not\equiv p_2$, some coefficient $c$ of $p = p_1 - p_2$ satisfies
So $p \not\equiv 0$ over $\mathbb{F}_q$. By Schwartz–Zippel lemma,

$$\Pr[p_1(r) = p_2(r) \pmod q] \leq \frac{md}{q} < \frac{1}{3}.$$

**Mini-Quiz 19.2**

Let $p(x_1, \ldots, x_m) = \prod_{j=1}^{n}(a_j\, x_{i_j} + b_j)$.

where $a_j, b_j \in \mathbb{Z}$ and $x_{i_j} \in \{x_1, \ldots, x_m\}$.
Each variable may appear more than once.

Which $d$ and $B$ can we choose?

a) $d = 1, \quad B = \prod_{j=1}^{n}(|a_j| + |b_j|)$

b) $d = n, \quad B = \prod_{j=1}^{n} \max\{|a_j|, |b_j|\}$

c) $d = n, \quad B = \prod_{j=1}^{n}(|a_j| + |b_j|)$

**Running time:** Polynomial in the length of $p_1, p_2$ (before expansion) and $\log q$.

**Conclusion:** $\mathrm{EQ}_{\mathrm{poly}} \in$ BPP.

# Quick Review of Today

1. Probabilistic TM, DTM with random tape

2. BPP (Bounded-error Probabilistic Polynomial Time)

3. Testing polynomial identity

These slides are adapted from Michael Sipser, *18.404J Theory of Computation*, Fall 2020. Massachusetts Institute of Technology:

- Source: MIT OpenCourseWare https://ocw.mit.edu

Except for separately identified third-party material, this adapted presentation is licensed under CC BY-NC-SA 4.0.