

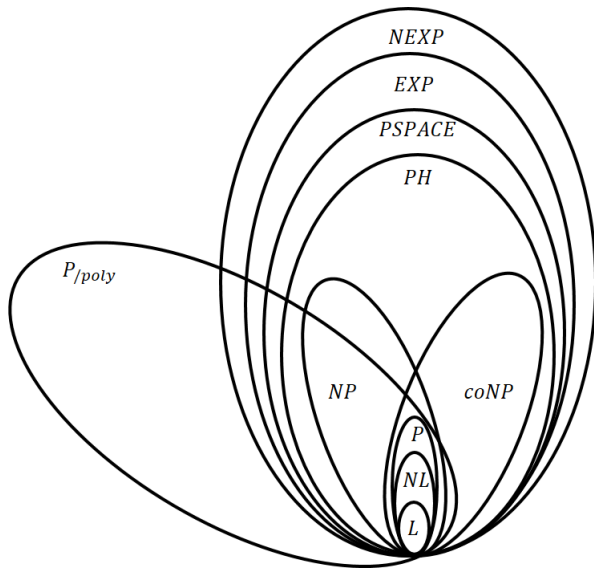
CSL853: Complexity Theory

Ragesh Jaiswal, CSE, IIT Delhi

Complexity Diagram

Complexity Diagram

Our current view of complexity classes



Randomized Computation

Randomized Computation

Probabilistic Turing Machines

Definition (Probabilistic Turing Machine (PTM))

A probabilistic Turing machine is a Turing machine with two transition functions δ_0, δ_1 . To execute a PTM M on an input x , we choose in each step with probability $1/2$ to apply the transition function δ_0 and with probability $1/2$ to apply δ_1 . The choice is made independently of all previous choices.

The machine only outputs 1 (accept) or 0 (reject). We denote by $M(x)$ the random variable corresponding to the value M writes at the end of this process. For a function $T : \mathbb{N} \rightarrow \mathbb{N}$, we say that M runs in $T(n)$ -time if for any input x , M halts on x within $T(|x|)$ steps regardless of the random choices it makes.

Randomized Computation

Probabilistic Turing Machines

Definition (BPTIME)

For $T : \mathbb{N} \rightarrow \mathbb{N}$ and $L \subseteq \{0,1\}^*$ we say that a PTM M decides L in time $T(n)$ if for every $x \in \{0,1\}^*$, M halts in $T(|x|)$ steps regardless of its random choices, and $\Pr[M(x) = L(x)] \geq 2/3$. $\text{BPTIME}(T(n))$ is the class of languages decided by PTMs in $O(T(n))$ time.

Definition (BPP)

$\text{BPP} = \cup_c \text{BPTIME}(n^c)$.

Randomized Computation

Probabilistic Turing Machines

Definition (BPTIME)

For $T : \mathbb{N} \rightarrow \mathbb{N}$ and $L \subseteq \{0, 1\}^*$ we say that a PTM M decides L in time $T(n)$ if for every $x \in \{0, 1\}^*$, M halts in $T(|x|)$ steps regardless of its random choices, and $\Pr[M(x) = L(x)] \geq 2/3$.

$\text{BPTIME}(T(n))$ is the class of languages decided by PTMs in $O(T(n))$ time.

Definition (BPP)

$\text{BPP} = \bigcup_c \text{BPTIME}(n^c)$.

Definition (BPP, alternate definition)

A language L is in BPP if there exists a polynomial-time TM M and a polynomial $p : \mathbb{N} \rightarrow \mathbb{N}$ such that for every $x \in \{0, 1\}^*$,

$\Pr_{r \in_R \{0,1\}^{p(|x|)}} [M(x, r) = L(x)] \geq 2/3$.

Randomized Computation

Probabilistic Turing Machines

Definition (BPP, alternate definition)

A language L is in BPP if there exists a polynomial-time TM M and a polynomial $p : \mathbb{N} \rightarrow \mathbb{N}$ such that for every $x \in \{0, 1\}^*$,
 $\Pr_{r \in_R \{0,1\}^{p(|x|)}} [M(x, r) = L(x)] \geq 2/3$.

- Show that $P \subseteq \text{BPP}$.
- Show that $\text{BPP} \subseteq \text{EXP}$.

Randomized Computation

BPP: examples

Definition (PRIMES)

$\text{PRIMES} = \{ \langle N \rangle : N \text{ is a prime number} \}.$

Theorem

$\text{PRIMES} \in \text{BPP}.$

Proof.

- For any N and $A \in [N-1]$, $QR_N(A) = \begin{cases} 0 & \text{if } \gcd(A, N) \neq 1 \\ +1 & \text{if } A = B^2 \pmod{N} \text{ for some } B \text{ with } \gcd(B, N) = 1 \\ -1 & \text{if otherwise} \end{cases}$
- Claim 1: For every odd prime N and $A \in [N-1]$, $QR_N(A) = A^{(N-1)/2} \pmod{N}$.
- Claim 2: For every odd N , A define the Jacobi symbol $(\frac{N}{A})$ as $\prod_{i=1}^k QR_{p_i}(A)$ where $p_1, \dots, p_k$ are all (not necessarily distinct) prime factors of N (i.e., $N = \prod_{i=1}^k p_i$). Then, the Jacobi symbol is computable in time $O(\log A \cdot \log N)$.
- Claim 3: For every odd composite N , among all $A \in [N-1]$ such that $\gcd(N, A) = 1$, at most half of the A 's satisfy $(\frac{N}{A}) = A^{(N-1)/2} \pmod{N}$. □

Randomized Computation

BPP examples: PRIMES

Definition (PRIMES)

$\text{PRIMES} = \{\langle N \rangle : N \text{ is a prime number}\}.$

Theorem

$\text{PRIMES} \in \text{BPP}.$

Theorem (Agarwal, Kayal, and Saxena, 2003)

$\text{PRIMES} \in \text{P}.$

Randomized Computation

BPP examples: Polynomial Identity Testing

- We will need the definition of an *algebraic circuit* for defining the problem.
- Informally, algebraic circuits are analogous to Boolean circuits except that instead of the operators $\wedge, \vee, \neg$, we have the arithmetic operators $+, -, \times$.

Definition (Algebraic circuit)

An n -variable arithmetic circuit is a directed acyclic graph with the sources labeled by a variable name from the set $x_1, \dots, x_n$, and each non-source node has in-degree two and is labeled by an operator from the set $\{+, -, \times\}$. There is a single sink in the graph which we call the output node. The arithmetic circuit defines a polynomial from $\mathbb{Z}_n$ to $\mathbb{Z}$ by placing the inputs on the sources and computing the value of each node using the appropriate operator.

Randomized Computation

BPP examples: Polynomial Identity Testing

- Informally, algebraic circuits are analogous to Boolean circuits except that instead of the operators $\wedge, \vee, \neg$, we have the arithmetic operators $+, -, \times$.

Definition (ZEROP)

$\text{ZEROP} = \{C :$

algebraic circuit C computes the identically zero polynomial}

- Let $\text{SAMEP} = \{\langle C, C' \rangle :$
algebraic circuits C and C' compute the same polynomial}.
- Show that $\text{SAMEP} \leq_p \text{ZEROP}$.

Randomized Computation

BPP examples: Polynomial Identity Testing

- Informally, algebraic circuits are analogous to Boolean circuits except that instead of the operators $\wedge, \vee, \neg$, we have the arithmetic operators $+, -, \times$.

Definition (ZEROP)

$\text{ZEROP} = \{C :$
algebraic circuit C computes the identically zero polynomial}

Theorem

$\text{ZEROP} \in \text{BPP}$.

Proof idea

- Evaluate the circuit on large number of inputs and see if all of them are 0.

Lemma (Schwartz-Zippel Lemma)

Let $p(x_1, x_2, \dots, x_m)$ be a non-zero polynomial to total degree at most d . Let S be a finite set of integers. Then, if $a_1, \dots, a_m$ are randomly chosen with replacement from S , then $\Pr[p(a_1, \dots, a_m) \neq 0] \geq 1 - \frac{d}{|S|}$.

Randomized Computation

BPP examples: Polynomial Identity Testing

- Informally, algebraic circuits are analogous to Boolean circuits except that instead of the operators $\wedge, \vee, \neg$, we have the arithmetic operators $+, -, \times$.

Definition (ZEROP)

$\text{ZEROP} = \{C :$
algebraic circuit C computes the identically zero polynomial}

Theorem

$\text{ZEROP} \in \text{BPP}.$

Open Problem

$\text{ZEROP} \stackrel{?}{\in} \text{P}.$

End