

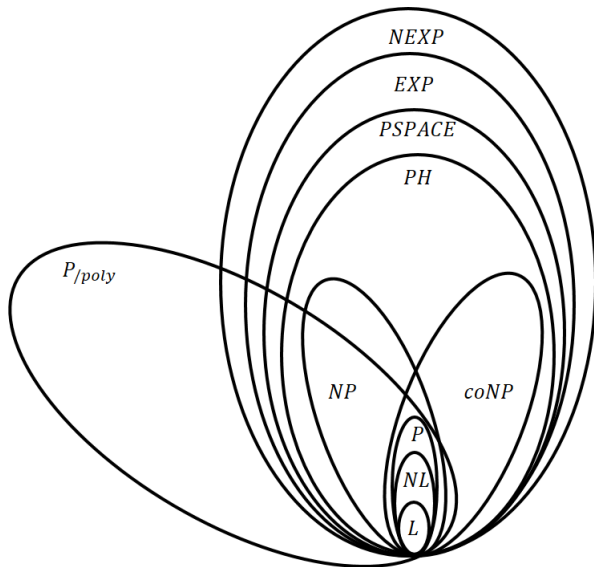
CSL853: Complexity Theory

Ragesh Jaiswal, CSE, IIT Delhi

Complexity Diagram

Complexity Diagram

Our current view of complexity classes



Randomized Computation

BPP examples: Perfect matching

Definition (MATCHING)

$\text{MATCHING} = \{ \langle G = (V_1, V_2, E) \rangle :$

G is a bipartite graph that has a perfect matching $\}$.

Theorem

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

Theorem

$\text{MATCHING} \in \text{randomNC}$.

Proof idea

- Let A be a matrix such that A_{ij} contains the variable x_{ij} if there is an edge from vertex i on the left to vertex j on the right, else A_{ij} has 0.
- Claim 1: G has a perfect matching if and only if $\det(A)$ is not identically zero polynomial.
- Claim 2: Language $\{ \langle M, k \rangle : M \text{ is a matrix with integer entries with determinant } k \}$ is in NC.

Randomized Computation

BPP examples: Perfect matching

Definition (MATCHING)

$\text{MATCHING} = \{ \langle G = (V_1, V_2, E) \rangle : G \text{ is a bipartite graph that has a perfect matching} \}.$

Theorem

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

Theorem

$\text{MATCHING} \in \text{randomNC}.$

Open Problem

$\text{MATCHING} \stackrel{?}{\in} \text{NC}.$

Randomized Computation

RP, coRP, ZPP

- BPP captures probabilistic algorithms with *two-sided* error.
- In certain applications, we might need *one-sided* error.

Definition (RTIME)

$\text{RTIME}(T(n))$ contains every language L for which there is a probabilistic TM M running in $T(n)$ times such that

$$x \in L \Rightarrow \Pr[M(x) = 1] \geq 2/3$$

$$x \notin L \Rightarrow \Pr[M(x) = 1] = 0.$$

Definition (RP)

$$\text{RP} = \cup_{c>0} \text{RTIME}(n^c).$$

Definition (coRP)

$$\text{coRP} = \{L : \bar{L} \in \text{RP}\}.$$

Randomized Computation

RP, coRP, ZPP

- BPP captures probabilistic algorithms with *two-sided* error.
- In certain applications, we might need *one-sided* error.
- In certain applications, we might need *zero-sided* error.
 - For any PTM M on input x , the running time of the machine on x denoted by $T_{M,x}$ is a random variable.
 - A PTM M is said to have expected running time $T(n)$ if the expectation $\mathbf{E}[T_{M,x}]$ is at most $T(|x|)$ for any $x \in \{0,1\}^*$.

Definition (ZTIME)

The class $\text{ZTIME}(T(n))$ contains all the languages L for which there is a PTM M that runs in expected time $O(T(n))$ such that for every input x , whenever M halts on x , the output $M(x)$ it produces is exactly $L(x)$.

Definition (ZPP)

$\text{ZPP} = \bigcup_{c>0} \text{ZTIME}(n^c)$.

Randomized Computation

RP, coRP, ZPP

Definition (ZTIME)

The class $\text{ZTIME}(T(n))$ contains all the languages L for which there is a PTM M that runs in expected time $O(T(n))$ such that for every input x , whenever M halts on x , the output $M(x)$ it produces is exactly $L(x)$.

Definition (ZPP)

$\text{ZPP} = \bigcup_{c>0} \text{ZTIME}(n^c)$.

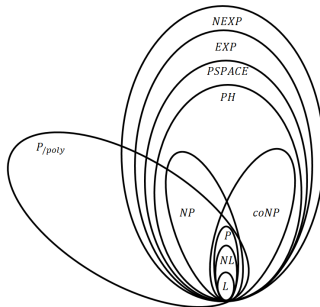
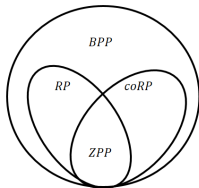
Theorem

$\text{ZPP} = \text{RP} \cap \text{coRP}$.

Randomized Computation

BPP, RP, coRP, ZPP

- Here is the relationship between the randomised classes.
- How does this new structure fit into our known complexity diagram?



Randomized Computation

BPP, RP, coRP, ZPP

- Claim 1: $P \subseteq ZPP$.
- Claim 2: $RP \subseteq NP$.
- Claim 3: $coRP \subseteq coNP$.
- How is BPP related to NP? Is it possible that $NP \subseteq BPP$?

Randomized Computation

BPP, RP, coRP, ZPP

- Claim 1: $P \subseteq BPP$.
- Claim 2: $RP \subseteq NP$.
- Claim 3: $coRP \subseteq coNP$.
- How is BPP related to NP?

Theorem (Adleman's Theorem, 1978)

$$BPP \subseteq P_{/poly}.$$

- Is it possible that $NP \subseteq BPP$? **This is unlikely.**

Randomized Computation

BPP, RP, coRP, ZPP

Theorem (Adleman's Theorem, Adleman, 1978)

$$\text{BPP} \subseteq \text{P}_{/\text{poly}}.$$

Proof.

- Claim 1: For any $L \in \text{BPP}$, there is a M and a polynomial p such that M runs in polynomial time such that

$$\Pr_{r \in_R \{0,1\}^{p(|x|)}} [M(x, r) = L(x)] > 1 - \frac{1}{2^{|x|+1}}.$$

Theorem (Chernoff Bounds)

Let $X_1, X_2, \dots, X_n$ be mutually independent random variables over $\{0, 1\}$ and let $\mu = \sum_{i=1}^n \mathbf{E}[X_i]$. Then for every $c > 0$,

$$\Pr \left[\left| \sum_{i=1}^n X_i - \mu \right| \geq c\mu \right] \leq 2 \cdot e^{-\mu \cdot \min(c^2/4, c/2)}$$

Randomized Computation

BPP, RP, coRP, ZPP

Theorem (Adleman's Theorem, Adleman, 1978)

$$\text{BPP} \subseteq \text{P}_{/\text{poly}}.$$

Proof.

- Claim 1: For any $L \in \text{BPP}$, there is a M and a polynomial p such that M runs in polynomial time such that

$$\Pr_{r \in_R \{0,1\}^{p(|x|)}} [M(x, r) = L(x)] > 1 - \frac{1}{2^{|x|+1}}.$$

- Consider inputs of size n . Let $m = p(n)$. Let $r \in \{0,1\}^m$ be called “bad” if there is an input $x \in \{0,1\}^n$ such that $M(x, r) \neq L(x)$, otherwise it is called “good”.
- Claim 2: For every n , there is a string $r \in \{0,1\}^m$ that is good for *all* inputs of length n .



Randomized Computation

BPP, RP, coRP, ZPP

Theorem (Adleman's Theorem, Adleman, 1978)

$$\text{BPP} \subseteq \text{P}_{/\text{poly}}.$$

Proof.

- Claim 1: For any $L \in \text{BPP}$, there is a M and a polynomial p such that M runs in polynomial time such that

$$\Pr_{r \in_R \{0,1\}^{p(|x|)}} [M(x, r) = L(x)] > 1 - \frac{1}{2^{|x|+1}}.$$

- Consider inputs of size n . Let $m = p(n)$. Let $r \in \{0,1\}^m$ be called “bad” if there is an input $x \in \{0,1\}^n$ such that $M(x, r) \neq L(x)$, otherwise it is called “good”.
- Claim 2: For every n , there is a string $r \in \{0,1\}^{p(n)}$ that is good for *all* inputs of length n .
- This shows that $\text{BPP} \subseteq \text{P}_{/\text{poly}}$ since such strings may be used as polynomial size “advice”.



Randomized Computation

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P$$

Theorem (Sipser-Gács-Lautemann theorem, 1983)

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P.$$

Proof.

- Claim 1: It is sufficient to show that $\text{BPP} \subseteq \Sigma_2^P$.

Theorem (Sipser-Gács-Lautemann theorem, 1983)

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P.$$

Proof.

- Claim 1: It is sufficient to show that $\text{BPP} \subseteq \Sigma_2^P$.
- Claim 2: For any $L \in \text{BPP}$, there is a TM M that on inputs of length n uses $m = \text{poly}(n)$ random bits such that

$$x \in L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \geq 1 - 2^{-n}$$

$$x \notin L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \leq 2^{-n}.$$

Randomized Computation

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P$$

Theorem (Sipser-Gács-Lautemann theorem, 1983)

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P.$$

Proof.

- Claim 1: It is sufficient to show that $\text{BPP} \subseteq \Sigma_2^P$.
- Claim 2: For any $L \in \text{BPP}$, there is a TM M that on inputs of length n uses $m = \text{poly}(n)$ random bits such that

$$x \in L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \geq 1 - 2^{-n}$$

$$x \notin L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \leq 2^{-n}.$$

- For a set $S \subseteq \{0,1\}^m$ and string $u \in \{0,1\}^m$, let $S + u = \{x + u : x \in S\}$, where $+$ denotes the bitwise XOR operation. Let $k = \lceil m/n \rceil + 1$.
- Claim 3: For every set $S \subseteq \{0,1\}^m$ with $|S| \leq 2^{m-n}$ and every k vectors $u_1, \dots, u_k \in \{0,1\}^m$, $\bigcup_{i=1}^k (S + u_i) \neq \{0,1\}^m$.

Randomized Computation

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P$$

Theorem (Sipser-Gács-Lautemann theorem, 1983)

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P.$$

Proof.

- Claim 1: It is sufficient to show that $\text{BPP} \subseteq \Sigma_2^P$.
- Claim 2: For any $L \in \text{BPP}$, there is a TM M that on inputs of length n uses $m = \text{poly}(n)$ random bits such that

$$x \in L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \geq 1 - 2^{-n}$$

$$x \notin L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \leq 2^{-n}.$$

- For a set $S \subseteq \{0,1\}^m$ and string $u \in \{0,1\}^m$, let $S + u = \{x + u : x \in S\}$, where $+$ denotes the bitwise XOR operation. Let $k = \lceil m/n \rceil + 1$.
- Claim 3: For every set $S \subseteq \{0,1\}^m$ with $|S| \leq 2^{m-n}$ and every k vectors $u_1, \dots, u_k \in \{0,1\}^m$, $\bigcup_{i=1}^k (S + u_i) \neq \{0,1\}^m$.
- Claim 4: For every set $S \subseteq \{0,1\}^m$ with $|S| \geq (1 - 2^{-n})2^m$, there exists $u_1, \dots, u_k \in \{0,1\}^m$ such that $\bigcup_{i=1}^k (S + u_i) = \{0,1\}^m$.

Randomized Computation

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P$$

Theorem (Sipser-Gács-Lautemann theorem, 1983)

$$\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P.$$

Proof.

- Claim 1: It is sufficient to show that $\text{BPP} \subseteq \Sigma_2^P$.
- Claim 2: For any $L \in \text{BPP}$, there is a TM M that on inputs of length n uses $m = \text{poly}(n)$ random bits such that

$$x \in L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \geq 1 - 2^{-n}$$

$$x \notin L \Rightarrow \Pr_{r \in \{0,1\}^m} [M(x, r) \text{ accepts}] \leq 2^{-n}.$$

- For a set $S \subseteq \{0,1\}^m$ and string $u \in \{0,1\}^m$, let $S + u = \{x + u : x \in S\}$, where $+$ denotes the bitwise XOR operation. Let $k = \lceil m/n \rceil + 1$.
- Claim 3: For every set $S \subseteq \{0,1\}^m$ with $|S| \leq 2^{m-n}$ and every k vectors $u_1, \dots, u_k \in \{0,1\}^m$, $\bigcup_{i=1}^k (S + u_i) \neq \{0,1\}^m$.
- Claim 4: For every set $S \subseteq \{0,1\}^m$ with $|S| \geq (1 - 2^{-n})2^m$, there exists $u_1, \dots, u_k \in \{0,1\}^m$ such that $\bigcup_{i=1}^k (S + u_i) = \{0,1\}^m$.
- So,

$$x \in L \Leftrightarrow \exists u_1, \dots, u_k \in \{0,1\}^m \forall r \in \{0,1\}^m \vee_{i=1}^k M(x, r \oplus u_i) \text{ accepts.}$$



Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem?

Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem? **Yes.**
 - $L = \{ \langle M, x, 1^t \rangle : M \text{ outputs } 1 \text{ within } t \text{ steps with probability at least } 2/3 \}$
- Does there exist a BPP-complete problem?
 - Is $L \in \text{BPP}$?

Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem? **Yes**.
 - $L = \{ \langle M, x, 1^t \rangle : M \text{ outputs 1 within } t \text{ steps with probability at least } 2/3 \}$
- Does there exist a BPP-complete problem? **Not known**
 - Is $L \in \text{BPP}$? **Unlikely**.

Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem? **Yes.**
- Does there exist a BPP-complete problem? **Not known**

Definition (Promise Problems)

A promise problem consists of two disjoint set of strings $L_{yes}, L_{no} \subseteq \{0, 1\}^*$ with $L_{yes} \cap L_{no} = \emptyset$. We only care about inputs from the set $L_{yes} \cup L_{no}$. That is the machine are allowed behave arbitrarily on inputs $x \notin L_{yes} \cup L_{no}$.

- Example:
 - *Promise-P* is the class of promise problems (L_{yes}, L_{no}) for which there exists a polynomial time TM M such that

$$x \in L_{yes} \Rightarrow M(x) = 1$$

$$x \in L_{no} \Rightarrow M(x) = 0$$

Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem? **Yes.**
- Does there exist a BPP-complete problem? **Not known**

Definition (Promise Problems)

A promise problem consists of two disjoint set of strings $L_{yes}, L_{no} \subseteq \{0,1\}^*$ with $L_{yes} \cap L_{no} = \emptyset$. We only care about inputs from the set $L_{yes} \cup L_{no}$. That is the machine are allowed behave arbitrarily on inputs $x \notin L_{yes} \cup L_{no}$.

Definition (*Promise*-BPP)

A promise problem (L_{yes}, L_{no}) is said to be in *Promise*-BPP if there a PTM M such that

$$x \in L_{yes} \Rightarrow \Pr[M(x) = 1] \geq 2/3$$

$$x \in L_{no} \Rightarrow \Pr[M(x) = 1] \leq 1/3$$

Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem? **Yes.**
- Does there exist a BPP-complete problem? **Not known**

Definition (*Promise*-BPP)

A promise problem (L_{yes}, L_{no}) is said to be in *Promise*-BPP if there a PTM M such that

$$x \in L_{yes} \Rightarrow \Pr[M(x) = 1] \geq 2/3$$

$$x \in L_{no} \Rightarrow \Pr[M(x) = 1] \leq 1/3$$

- How is *Promise*-BPP related to BPP?

Randomized Computation

Complete problems for BPP

- Does there exist a BPP-hard problem? **Yes**.
- Does there exist a BPP-complete problem? **Not known**

Definition (*Promise-BPP*)

A promise problem $(L_{\text{yes}}, L_{\text{no}})$ is said to be in *Promise-BPP* if there a PTM M such that

$$x \in L_{\text{yes}} \Rightarrow \Pr[M(x) = 1] \geq 2/3$$

$$x \in L_{\text{no}} \Rightarrow \Pr[M(x) = 1] \leq 1/3$$

- How is *Promise-BPP* related to BPP? **$\text{BPP} \subseteq \text{Promise-BPP}$** .
- Does there exist a *Promise-BPP*-complete problem? **Yes**.
 - $L_{\text{yes}} = \{\langle M, x, 1^t \rangle : M \text{ outputs 1 within } t \text{ steps with probability at least } 2/3\}$.
 - $L_{\text{no}} = \{\langle M, x, 1^t \rangle : M \text{ outputs 1 within } t \text{ steps with probability at most } 1/3\}$

Randomized Computation

Complete problems for BPP

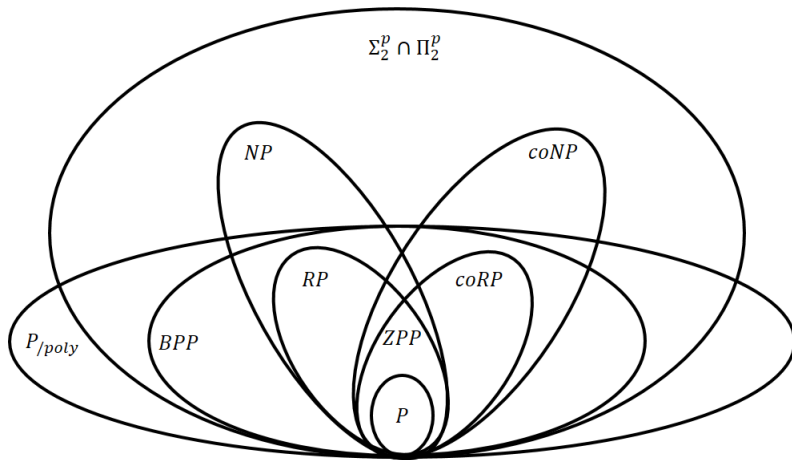
- Does there exist a BPP-hard problem? **Yes.**
- Does there exist a BPP-complete problem? **Not known**
- How is *Promise*-BPP related to BPP? **$BPP \subseteq \text{Promise-BPP}$.**
- Does there exist a *Promise*-BPP-complete problem? **Yes.**
- Is there a hierarchy (similar to that within P) within BPP?
Not known.

Theorem (Impagliazzo-Wigderson, 1997)

If SAT (or any problem in $\text{DTIME}(2^{O(n)})$) requires exponentially large circuits, i.e., $2^{\Omega(n)}$, then $P = BPP$.

Complexity Diagram

Randomized Computation



Randomized Computation

Randomized Space-bounded Computations

Definition (BPL)

A language L is in BPL if there is an $O(\log n)$ -space PTM M such that $\Pr[M(x) = L(x)] \geq 2/3$.

Definition (RL)

A language L is in RL if there is an $O(\log n)$ -space PTM M such that if $x \in L$, then $\Pr[M(x) = 1] \geq 2/3$ and if $x \notin L$, then $\Pr[M(x) = 1] = 0$.

- Argue that $RL \subseteq NL$, $RL \subseteq BPL$.
- Let $UPATH = \{ \langle G, s, t \rangle : \text{There is a path from between vertices } s \text{ and } t \text{ in undirected graph } G \}$.

Theorem (Aleliunas, Karp, Lipton, 1979)

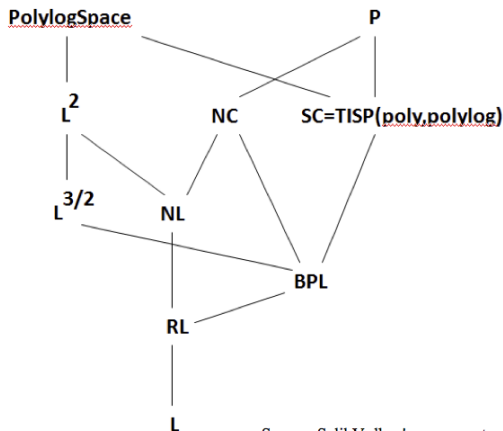
$UPATH \in RL$.

Proof.

- Algorithm: Take a *random walk* of length $l = 100 \cdot n^4$ starting from s . If the random walk visits t , then output accept, otherwise output reject.
- Claim: If s and t are connected, then the expected number of steps it takes for a walk from s to t is at most $10 \cdot n^4$. □

Complexity Diagram

Randomized Space-bounded Computations



End