Randomized Complexity Classes

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error
- **coRP** (complement of **RP**)

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error
- **coRP** (complement of **RP**)
  - One-sided error

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error
- **coRP** (complement of **RP**)
  - One-sided error
- **BPP** (Bounded-error Probabilistic Polynomial time)

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error
- **coRP** (complement of **RP**)
  - One-sided error
- **BPP** (Bounded-error Probabilistic Polynomial time)
  - Two-sided error

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error
- **coRP** (complement of **RP**)
  - One-sided error
- **BPP** (Bounded-error Probabilistic Polynomial time)
  - Two-sided error
- **ZPP** (Zero-error Probabilistic Polynomial time)

Randomized Complexity Classes

- **RP** (Randomized Polynomial time)
  - One-sided error
- **coRP** (complement of **RP**)
  - One-sided error
- **BPP** (Bounded-error Probabilistic Polynomial time)
  - Two-sided error
- **ZPP** (Zero-error Probabilistic Polynomial time)
  - Zero error

Randomized Polynomial time

Randomized Polynomial time

Definition: **RP**

The complexity class **RP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies Prob[M(x) = 1] \geq \frac{1}{2}$$

$$x \notin L \implies Prob[M(x) = 0] = 1$$

Randomized Polynomial time

Definition: RP

The complexity class **RP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies Prob[M(x) = 1] \geq \frac{1}{2}$$

$$x \notin L \implies Prob[M(x) = 0] = 1$$

Definition: coRP

The complexity class **coRP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies Prob[M(x) = 1] = 1$$

$$x \notin L \implies Prob[M(x) = 0] \geq \frac{1}{2}$$

Randomized Polynomial time

Randomized Polynomial time

Let PTM M decide a language $L \in \mathbf{RP}$

$$x \in L \implies Prob[M(x) = 1] \geq \frac{1}{2}$$

$$x \notin L \implies Prob[M(x) = 0] = 1$$

- If $M(x) = 1$, then $x \in L$.
- If $M(x) = 0$, then $x \in L$ or $x \notin L$, we can't really tell.

Randomized Polynomial time

Let PTM M decide a language $L \in$ RP

$$x \in L \implies Prob[M(x) = 1] \geq \frac{1}{2}$$

$$x \notin L \implies Prob[M(x) = 0] = 1$$

- If $M(x) = 1$, then $x \in L$.
- If $M(x) = 0$, then $x \in L$ or $x \notin L$, we can't really tell.

Let PTM M decide a language $L \in$ coRP

$$x \in L \implies Prob[M(x) = 1] = 1$$

$$x \notin L \implies Prob[M(x) = 0] \geq \frac{1}{2}$$

- If $M(x) = 0$, then $x \notin L$.
- If $M(x) = 1$, then $x \in L$ or $x \notin L$, we can't really tell.

RP is in NP

$\mathbf{RP}$ is in $\mathbf{NP}$

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

RP is in NP

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

Proof

RP is in **NP**

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

Proof

- Let L be a language in **RP** . Let M be a polynomial probabilistic Turing Machine that decides L.

RP is in NP

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

Proof

- Let L be a language in **RP** . Let M be a polynomial probabilistic Turing Machine that decides L.
- If $x \in L$, then there exists a sequence of coin tosses y such that M accepts x with y as the random string.

RP is in NP

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

Proof

- Let L be a language in **RP** . Let M be a polynomial probabilistic Turing Machine that decides L.
- If $x \in L$, then there exists a sequence of coin tosses y such that M accepts x with y as the random string.
- So we can consider y as a certificate instead of a random string. y can be verified in polynomial time by the same machine.

RP is in **NP**

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

Proof

- Let L be a language in **RP** . Let M be a polynomial probabilistic Turing Machine that decides L.
- If $x \in L$, then there exists a sequence of coin tosses y such that M accepts x with y as the random string.
- So we can consider y as a certificate instead of a random string. y can be verified in polynomial time by the same machine.
- If $x \notin L$, then $\mathrm{Prob}[M(x) = 1] = 0$. So there is no certificate. M rejects x.

RP is in NP

Theorem

$$\mathbf{RP} \subseteq \mathbf{NP}$$

Proof

- Let L be a language in **RP** . Let M be a polynomial probabilistic Turing Machine that decides L.
- If $x \in L$, then there exists a sequence of coin tosses y such that M accepts x with y as the random string.
- So we can consider y as a certificate instead of a random string. y can be verified in polynomial time by the same machine.
- If $x \notin L$, then $\text{Prob}[M(x) = 1] = 0$. So there is no certificate. M rejects x.
- So $L \in \mathbf{NP}$

Boosting for **RP**

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- We accept the string x if any of the t runs accept, and reject x otherwise

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- We accept the string x if any of the t runs accept, and reject x otherwise
- Clearly, if $x \notin L$, all t runs will return a 0

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- We accept the string x if any of the t runs accept, and reject x otherwise
- Clearly, if $x \notin L$, all t runs will return a 0
- If $x \in L$, if *any* of the t runs returns a 1, we return the correct answer

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- We accept the string x if any of the t runs accept, and reject x otherwise
- Clearly, if $x \notin L$, all t runs will return a 0
- If $x \in L$, if *any* of the t runs returns a 1, we return the correct answer
- If $x \in L$, if *all* t runs returns 0, we make mistake

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- We accept the string x if any of the t runs accept, and reject x otherwise
- Clearly, if $x \notin L$, all t runs will return a 0
- If $x \in L$, if *any* of the t runs returns a 1, we return the correct answer
- If $x \in L$, if *all* t runs returns 0, we make mistake
- Pr[algorithm makes a mistake t times] $\leq \frac{1}{2^t}$

Boosting for **RP**

- The constant $1/2$ in the definition of **RP** can be replaced by any constant k, $0 < k < 1$
- Since the error is one-sided, we can repeat the algorithm t times independently:
- We accept the string x if any of the t runs accept, and reject x otherwise
- Clearly, if $x \notin L$, all t runs will return a 0
- If $x \in L$, if *any* of the t runs returns a 1, we return the correct answer
- If $x \in L$, if *all* t runs returns 0, we make mistake
- Pr[algorithm makes a mistake t times] $\leq \frac{1}{2^t}$
- Thus, we can make the error exponentially small by polynomial number of repetitions.

Examples of **RP** and **coRP**

Examples of **RP** and **coRP**

Example of **RP**

Primes, the set of all prime numbers, is in **RP** .
This was shown by Adelman and Huang (1992), who gave a primality testing algorithm that always rejected composite numbers, and accepted primes with probability at least $1/2$

Examples of **RP** and **coRP**

Example of **RP**

Primes, the set of all prime numbers, is in **RP** .
This was shown by Adelman and Huang (1992), who gave a primality testing algorithm that always rejected composite numbers, and accepted primes with probability at least $1/2$

Example of **coRP**

Primes, the set of all prime numbers, is in **coRP** also.
Miller and Rabin gave a primality test that always accepts prime numbers, but rejects composites with probability at least $1/2$

Bounded-error Probabilistic Polynomial time

Bounded-error Probabilistic Polynomial time

Definition: BPP

The complexity class **BPP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies Prob[M(x) = 1] \geq \frac{2}{3}$$

$$x \notin L \implies Prob[M(x) = 1] < \frac{1}{3}$$

M answers correctly with probability $2/3$ on any input x regardless if $x \in L$ or $x \notin L$

Bounded-error Probabilistic Polynomial time

Definition: **BPP**

The complexity class **BPP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies Prob[M(x) = 1] \geq \frac{2}{3}$$

$$x \notin L \implies Prob[M(x) = 1] < \frac{1}{3}$$

M answers correctly with probability $2/3$ on any input x regardless if $x \in L$ or $x \notin L$

Example of **BPP**

Primes, the set of all prime numbers, is in **BPP** .
BPP is closed under complement.

$$\mathbf{BPP} = \mathbf{coBPP}$$

Boosting for **BPP**

Boosting for BPP

- $2/3$ is arbitrary and can be improved as follows:

Boosting for **BPP**

- 2/3 is arbitrary and can be improved as follows:
  - Repeat the algorithm t times, say it returns X_i at the i-th run

Boosting for **BPP**

- $2/3$ is arbitrary and can be improved as follows:
  - Repeat the algorithm t times, say it returns X_i at the i-th run
  - Take majority answer, i.e., if $\geq t/2$ times M returns 1, return 1, otherwise return 0

Boosting for **BPP**

- $2/3$ is arbitrary and can be improved as follows:
  - Repeat the algorithm t times, say it returns X_i at the i-th run
  - Take majority answer, i.e., if $\geq t/2$ times M returns 1, return 1, otherwise return 0

The Chernoff Bound

Suppose $X_1, \ldots, X_t$ are t independent random variables with values in $\{0, 1\}$ and for every i, $\Pr[X_i = 1] = p$.

Boosting for **BPP**

- 2/3 is arbitrary and can be improved as follows:
  - Repeat the algorithm t times, say it returns X_i at the i-th run
  - Take majority answer, i.e., if $\geq t/2$ times M returns 1, return 1, otherwise return 0

The Chernoff Bound

Suppose $X_1, \ldots, X_t$ are t independent random variables with values in $\{0, 1\}$ and for every i, $\Pr[X_i = 1] = p$. Then

$$Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - p > \epsilon] < e^{-\epsilon^2 \cdot \frac{t}{2p(1-p)}}$$

$$Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - p > -\epsilon] < e^{-\epsilon^2 \cdot \frac{t}{2p(1-p)}}$$

Boosting for **BPP**

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- $Pr[$ Algorithm outputs the wrong answer on $x]$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- Pr[Algorithm outputs the wrong answer on x]

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- Pr[Algorithm outputs the wrong answer on x]

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i < \frac{1}{2}]$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- $Pr[$ Algorithm outputs the wrong answer on $x]$

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i < \frac{1}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i - 2/3 < -\frac{1}{6}]$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- $Pr[$ Algorithm outputs the wrong answer on $x]$

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i < \frac{1}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i - 2/3 < -\frac{1}{6}]$

  $\leq e^{-\frac{t}{36 \cdot 2 \cdot 2/3 \cdot 1/3}}$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- $Pr[$ Algorithm outputs the wrong answer on $x]$

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

  $= Pr[\frac{1}{t}\sum_{i=1}^{t} X_i < \frac{1}{2}]$

  $= Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - 2/3 < -\frac{1}{6}]$

  $\leq e^{-\frac{t}{36 \cdot 2 \cdot 2/3 \cdot 1/3}}$

  $= e^{-ct}$, where c is some constant

Boosting for BPP

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- $Pr[$ Algorithm outputs the wrong answer on $x]$

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i < \frac{1}{2}]$

  $= Pr[\frac{1}{t} \sum_{i=1}^{t} X_i - 2/3 < -\frac{1}{6}]$

  $\leq e^{-\frac{t}{36 \cdot 2 \cdot 2/3 \cdot 1/3}}$

  $= e^{-ct}$, where c is some constant

  $\leq \frac{1}{2^n}$ for $t = (\ln 2^n)/c$

Boosting for **BPP**

- $x \in L$: M outputs correctly if $\sum_{i=1}^{t} X_i \geq \frac{t}{2}$
- The algorithm makes a mistake when $\sum_{i=1}^{t} X_i < \frac{t}{2}$
- Pr[Algorithm outputs the wrong answer on x]

  $= Pr[\sum_{i=1}^{t} X_i < \frac{t}{2}]$

  $= Pr[\frac{1}{t}\sum_{i=1}^{t} X_i < \frac{1}{2}]$

  $= Pr[\frac{1}{t}\sum_{i=1}^{t} X_i - 2/3 < -\frac{1}{6}]$

  $\leq e^{-\frac{t}{36 \cdot 2 \cdot 2/3 \cdot 1/3}}$

  $= e^{-ct}$, where c is some constant

  $\leq \frac{1}{2^n}$ for $t = (\ln 2^n)/c$
- So by running the algorithm $O(n)$ times, reduce probability exponentially

RP $\subseteq$ BPP

RP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{1}{2}$
- $x \notin L \Rightarrow Prob[M(x) = 1] = 0$

RP $\subseteq$ BPP

RP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{1}{2}$
- $x \notin L \Rightarrow Prob[M(x) = 1] = 0$

BPP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{2}{3}$
- $x \notin L \Rightarrow Prob[M(x) = 1] < \frac{1}{3}$

RP $\subseteq$ BPP

RP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{1}{2}$
- $x \notin L \Rightarrow Prob[M(x) = 1] = 0$

BPP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{2}{3}$
- $x \notin L \Rightarrow Prob[M(x) = 1] < \frac{1}{3}$

- Can boost **RP** probability by running twice:

RP $\subseteq$ BPP

RP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{1}{2}$
- $x \notin L \Rightarrow Prob[M(x) = 1] = 0$

BPP

- $x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{2}{3}$
- $x \notin L \Rightarrow Prob[M(x) = 1] < \frac{1}{3}$

- Can boost **RP** probability by running twice:

$$x \in L \Rightarrow Prob[M(x) = 1] \geq \frac{3}{4}$$

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

- Clearly the above definition satisfies **BPP**

Randomized Computation

Nabil Mustafa

Computational Complexity

Zero-error Probabilistic Polynomial time

Zero-error Probabilistic Polynomial time

Definition: **ZPP**

The complexity class **ZPP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies M(x) = 1 \text{ or } M(x) = \textbf{`Don't know'}$$

Zero-error Probabilistic Polynomial time

Definition: ZPP

The complexity class **ZPP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies M(x) = 1 \textit{ or } M(x) = \textbf{`Don't know'}$$

$$x \notin L \implies M(x) = 0 \textit{ or } M(x) = \textbf{`Don't know'}$$

Zero-error Probabilistic Polynomial time

Definition: ZPP

The complexity class **ZPP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies M(x) = 1 \text{ } \mathit{or} \text{ } M(x) = \textbf{`Don't know'}$$

$$x \notin L \implies M(x) = 0 \text{ } \mathit{or} \text{ } M(x) = \textbf{`Don't know'}$$

$$\forall x, \ \mathit{Prob}[M(x) = \textbf{`Don't know'}] \leq 1/2$$

Zero-error Probabilistic Polynomial time

Definition: ZPP

The complexity class **ZPP** is the class of all languages L for which there exists a polynomial **PTM** M such that

$$x \in L \implies M(x) = 1 \textit{ or } M(x) = \textbf{`Don't know'}$$

$$x \notin L \implies M(x) = 0 \textit{ or } M(x) = \textbf{`Don't know'}$$

$$\forall x,\ \mathit{Prob}[M(x) = \textbf{`Don't know'}] \leq 1/2$$

Whenever M answers with a 0 or a 1, it answers correctly. If M is not sure, it'll output a **'Don't know'**. On any input x, it outputs **'Don't know'** with probability at most $1/2$.

A Theorem on ZPP

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

A Theorem on ZPP

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

A Theorem on $\mathbf{ZPP}$

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.

A Theorem on **ZPP**

Theorem

$$\textbf{ZPP} = \textbf{RP} \cap \textbf{coRP}$$

Claim: $\textbf{ZPP} \subseteq \textbf{RP} \cap \textbf{coRP}$

- Show that $\textbf{ZPP} \subseteq \textbf{RP}$ and $\textbf{ZPP} \subseteq \textbf{coRP}$
- We prove $\textbf{ZPP} \subseteq \textbf{coRP}$, the other proof is similar
- Let $L \in \textbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$:

A Theorem on ZPP

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$

A Theorem on ZPP

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.

A Theorem on **ZPP**

Theorem

$$\textbf{ZPP} = \textbf{RP} \cap \textbf{coRP}$$

Claim: $\textbf{ZPP} \subseteq \textbf{RP} \cap \textbf{coRP}$

- Show that $\textbf{ZPP} \subseteq \textbf{RP}$ and $\textbf{ZPP} \subseteq \textbf{coRP}$
- We prove $\textbf{ZPP} \subseteq \textbf{coRP}$, the other proof is similar
- Let $L \in \textbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.
  - $x \notin L$:

A Theorem on ZPP

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.
  - $x \notin L$: $M(x) = 0$

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
    - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.
    - $x \notin L$: $M(x) = 0$, which is correct

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.
  - $x \notin L$: $M(x) = 0$, which is correct, or $M(x) =$**'Don't know'**

A Theorem on **ZPP**

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.
  - $x \notin L$: $M(x) = 0$, which is correct, or $M(x) =$**'Don't know'** where $M'(x)$ returns incorrectly with probability $\leq 1/2$.

A Theorem on ZPP

Theorem

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$$

Claim: $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$

- Show that $\mathbf{ZPP} \subseteq \mathbf{RP}$ and $\mathbf{ZPP} \subseteq \mathbf{coRP}$
- We prove $\mathbf{ZPP} \subseteq \mathbf{coRP}$, the other proof is similar
- Let $L \in \mathbf{ZPP}$. Then $\exists$ **PTM** M that, for all x, either correctly decides $x \in L$ or outputs **'Don't know'**.
- Let M' be the Turing Machine that on input x, returns 1 if $M(x) =$**'Don't know'** and otherwise returns $M(x)$.
  - $x \in L$: $M(x) = 1$ or $M(x) =$**'Don't know'**, so $M'(x) = 1$.
  - $x \notin L$: $M(x) = 0$, which is correct, or $M(x) =$**'Don't know'** where $M'(x)$ returns incorrectly with probability $\leq 1/2$.
- So $L \in \mathbf{coRP}$, and $\mathbf{ZPP} \subseteq \mathbf{coRP}$

A Theorem on ZPP

Claim: $\mathbf{RP} \cap \mathbf{coRP} \subseteq \mathbf{ZPP}$

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.

A Theorem on ZPP

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.
  - Else return **'Don't know'**.

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.
  - Else return **'Don't know'**.
- Claim: If $M'(x)$ returns a 0 or a 1, it is correct.

A Theorem on ZPP

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.
  - Else return **'Don't know'**.
- Claim: If $M'(x)$ returns a 0 or a 1, it is correct.
- Claim: $M'(x)$ returns **'Don't know'** with prob. $\leq 1/2$

A Theorem on ZPP

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.
  - Else return **'Don't know'**.
- Claim: If $M'(x)$ returns a 0 or a 1, it is correct.
- Claim: $M'(x)$ returns **'Don't know'** with prob. $\leq 1/2$
  - Assume $x \in L$. Then we don't return a 1 iff $M_2(x) = 0$.

A Theorem on ZPP

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.
  - Else return **'Don't know'**.
- Claim: If $M'(x)$ returns a 0 or a 1, it is correct.
- Claim: $M'(x)$ returns **'Don't know'** with prob. $\leq 1/2$
  - Assume $x \in L$. Then we don't return a 1 iff $M_2(x) = 0$.
  - By definition, $M_2(x) = 0$ for $x \in L$ with prob. $\leq 1/2$.

A Theorem on **ZPP**

Claim: **RP** $\cap$ **coRP** $\subseteq$ **ZPP**

- Let $L \in$ **RP** $\cap$ **coRP** . Then there exist two **TM** 's s.t.:
  - $M_1(x) = 1$ if $x \in L$. Incorrect for $x \notin L$ with prob. $\leq 1/2$
  - $M_2(x) = 0$ if $x \notin L$. Incorrect for $x \in L$ with prob. $\leq 1/2$
- Construct a **PTM** M' which works as follows on $M'(x)$:
  - Run $M_1(x)$, and if $M_1(x) = 0$, return $x \notin L$.
  - Run $M_2(x)$, and if $M_2(x) = 1$, return $x \in L$.
  - Else return **'Don't know'**.
- Claim: If $M'(x)$ returns a 0 or a 1, it is correct.
- Claim: $M'(x)$ returns **'Don't know'** with prob. $\leq 1/2$
  - Assume $x \in L$. Then we don't return a 1 iff $M_2(x) = 0$.
  - By definition, $M_2(x) = 0$ for $x \in L$ with prob. $\leq 1/2$.
  - The case for $x \notin L$ similar.