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

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

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

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

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

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

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

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

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

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

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

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

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

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 **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

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

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 $M(x)$.
  - $x \in L$:

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 $M(x)$.
    - $x \in L$: $M(x) = 1$

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

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

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 $M'(x) = 1$.
  - $x \notin L$:

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.

A Theorem on **BPP**

Theorem

$$\mathbf{BPP} \subseteq \Sigma_2$$

A Theorem on BPP

Theorem

$$\text{BPP} \subseteq \Sigma_2$$

Let $L \in$ BPP .

A Theorem on **BPP**

Theorem

$$\mathbf{BPP} \subseteq \Sigma_2$$

Let $L \in$ **BPP** . Then $\exists$ a randomized **TM** M such that:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-|x|}$

A Theorem on **BPP**

Theorem

$$\mathbf{BPP} \subseteq \Sigma_2$$

Let $L \in$ **BPP** . Then $\exists$ a randomized **TM** M such that:

- $x \in L$: Prob $[M(x,y) = 1] \geq 1 - 2^{-|x|}$
- $x \notin L$: Prob $[M(x,y) = 1] \leq 2^{-|x|}$

where y is a m-bit random string.

A Theorem on BPP

Theorem

$$\mathbf{BPP} \subseteq \Sigma_2$$

Let $L \in$ **BPP** . Then $\exists$ a randomized **TM** M such that:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-|x|}$
- $x \notin L$: Prob $[M(x, y) = 1] \leq 2^{-|x|}$

where y is a m-bit random string. To prove $L \in \Sigma_2$, show there exists **TM** N:

$$x \in L \Longleftrightarrow \exists\ u \quad \forall\ v \quad N(x, u, v) = 1$$

A Theorem on BPP

Theorem

$$\textbf{BPP} \subseteq \Sigma_2$$

Let $L \in$ **BPP** . Then $\exists$ a randomized **TM** M such that:

- $x \in L$: Prob $[M(x,y) = 1] \geq 1 - 2^{-|x|}$
- $x \notin L$: Prob $[M(x,y) = 1] \leq 2^{-|x|}$

where y is a m-bit random string. To prove $L \in \Sigma_2$, show there exists **TM** N:

$$x \in L \Longleftrightarrow \exists\ u \quad \forall\ v \quad N(x,u,v) = 1$$

- m is the number of random bits M uses on $|x| = n$. Note that m is polynomial in n.

A Theorem on BPP

Theorem

$$\textbf{BPP} \subseteq \Sigma_2$$

Let $L \in$ **BPP** . Then $\exists$ a randomized **TM** M such that:

- $x \in L$: Prob $[M(x,y) = 1] \geq 1 - 2^{-|x|}$
- $x \notin L$: Prob $[M(x,y) = 1] \leq 2^{-|x|}$

where y is a m-bit random string. To prove $L \in \Sigma_2$, show there exists **TM** N:

$$x \in L \Longleftrightarrow \exists\ u \quad \forall\ v \quad N(x,u,v) = 1$$

- m is the number of random bits M uses on $|x| = n$. Note that m is polynomial in n.
- $y_1, \ldots, y_{2^m}$ denote the 2^m possible random strings

A Theorem on **BPP**

Theorem

$$\mathbf{BPP} \subseteq \Sigma_2$$

Let $L \in$ **BPP** . Then $\exists$ a randomized **TM** M such that:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-|x|}$
- $x \notin L$: Prob $[M(x, y) = 1] \leq 2^{-|x|}$

where y is a m-bit random string. To prove $L \in \Sigma_2$, show there exists **TM** N:

$$x \in L \Longleftrightarrow \exists\ u \quad \forall\ v \quad N(x, u, v) = 1$$

- m is the number of random bits M uses on $|x| = n$. Note that m is polynomial in n.
- $y_1, \ldots, y_{2^m}$ denote the 2^m possible random strings
- Input to M is a m-bit string picked uniformly from all the y_i's

The Construction

From now on, lets fix the input x where $|x| = n$. Then:

- $x \in L$: $\text{Prob}\left[M(x, y) = 1\right] \geq 1 - 2^{-n}$
- $x \notin L$: $\text{Prob}\left[M(x, y) = 1\right] \leq 2^{-n}$

The Construction

From now on, lets fix the input x where $|x| = n$. Then:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-n}$
- $x \notin L$: Prob $[M(x, y) = 1] \leq 2^{-n}$

This is equivalent to the following:

- $x \in L$: $\geq (1 - 2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$

The Construction

From now on, lets fix the input x where $|x| = n$. Then:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-n}$
- $x \notin L$: Prob $[M(x, y) = 1] \leq 2^{-n}$

This is equivalent to the following:

- $x \in L$: $\geq (1 - 2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$
- $x \notin L$: $\leq (2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$

The Construction

From now on, lets fix the input x where $|x| = n$. Then:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-n}$
- $x \notin L$: Prob $[M(x, y) = 1] \leq 2^{-n}$

This is equivalent to the following:

- $x \in L$: $\geq (1 - 2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$
- $x \notin L$: $\leq (2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$

To prove $L \in \Sigma_2$, have to find a way to distinguish the two cases using a '$\exists$' and a '$\forall$' quantifier.

The Construction

From now on, lets fix the input x where $|x| = n$. Then:

- $x \in L$: Prob $[M(x, y) = 1] \geq 1 - 2^{-n}$
- $x \notin L$: Prob $[M(x, y) = 1] \leq 2^{-n}$

This is equivalent to the following:

- $x \in L$: $\geq (1 - 2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$
- $x \notin L$: $\leq (2^{-n}) \cdot 2^m$ strings y_i s.t. $M(x, y_i) = 1$

To prove $L \in \Sigma_2$, have to find a way to distinguish the two cases using a '$\exists$' and a '$\forall$' quantifier.

Basic idea: Find a way to formulate, with polynomial-sized certificates u and v, the fact that $x \in L$ if and only if **there exist** lots ($\geq (1 - 2^{-n}) \cdot 2^m$) of m-bit strings y_i such that **for all** of them, N returns 1.

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- p_1, ..., p_{2^m} are 2^m permutations of the y_i's

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- p_1, ..., p_{2^m} are 2^m permutations of the y_i's
- Claim: Each permutation p_i can be coded by a m-bit string

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- $p_1, \ldots, p_{2^m}$ are 2^m permutations of the y_i's
- Claim: Each permutation p_i can be coded by a m-bit string
  - For a fixed p, and any y_i, consider m-bit string $y_i \oplus p$

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- p_1, ..., p_{2^m} are 2^m permutations of the y_i's
- Claim: Each permutation p_i can be coded by a m-bit string
  - For a fixed p, and any y_i, consider m-bit string $y_i \oplus p$
  - For contradiction, assume $T = y_i \oplus p = y_j \oplus p$

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- $p_1, \ldots, p_{2^m}$ are 2^m permutations of the y_i's
- Claim: Each permutation p_i can be coded by a m-bit string
  - For a fixed p, and any y_i, consider m-bit string $y_i \oplus p$
  - For contradiction, assume $T = y_i \oplus p = y_j \oplus p$
  - $A \oplus B = C \implies A = B \oplus C$. So $\oplus$ is a 1-to-1 function

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- $p_1, \ldots, p_{2^m}$ are 2^m permutations of the y_i's
- Claim: Each permutation p_i can be coded by a m-bit string
  - For a fixed p, and any y_i, consider m-bit string $y_i \oplus p$
  - For contradiction, assume $T = y_i \oplus p = y_j \oplus p$
  - $A \oplus B = C \implies A = B \oplus C$. So $\oplus$ is a 1-to-1 function
  - Hence $y_i = y_j = T \oplus p$, contradiction since each y_i distinct

The Construction

Consider the 2^m strings y_i's and take 2^m *permutations*

	y_1	y_2	y_3	y_4	y_5	y_6
p_1	y_2	y_3	y_1	y_6	y_4	y_5
p_2	y_6	y_1	y_5	y_2	y_3	y_4
p_3	y_4	y_3	y_5	y_1	y_6	y_2
p_4	y_3	y_4	y_6	y_5	y_2	y_1

	y_1	$\bullet$	$\bullet$	y_{2^m}
p_1	$p_1 \oplus y_1$			$p_1 \oplus y_{2^m}$
$\bullet$				$\bullet$
$\bullet$				$\bullet$
p_{2^m}	$p_{2^m} \oplus y_1$	$\bullet$	$\bullet$	$p_{2^m} \oplus y_{2^m}$

- $p_1, \ldots, p_{2^m}$ are 2^m permutations of the y_i's
- Claim: Each permutation p_i can be coded by a m-bit string
  - For a fixed p, and any y_i, consider m-bit string $y_i \oplus p$
  - For contradiction, assume $T = y_i \oplus p = y_j \oplus p$
  - $A \oplus B = C \implies A = B \oplus C$. So $\oplus$ is a 1-to-1 function
  - Hence $y_i = y_j = T \oplus p$, contradiction since each y_i distinct
- The value in each cell: $A_{ij} = M(x, P_i \oplus y_j)$

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \to y_j$ and p_2 also maps $y_i \to y_j$

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \rightarrow y_j$ and p_2 also maps $y_i \rightarrow y_j$
  - Then $y_j = y_i \oplus p_1$ and $y_j = y_i \oplus p_2$

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \rightarrow y_j$ and p_2 also maps $y_i \rightarrow y_j$
  - Then $y_j = y_i \oplus p_1$ and $y_j = y_i \oplus p_2$
  - But then, $p_1 = y_j \oplus y_i$ and $p_2 = y_j \oplus y_i$

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \to y_j$ and p_2 also maps $y_i \to y_j$
  - Then $y_j = y_i \oplus p_1$ and $y_j = y_i \oplus p_2$
  - But then, $p_1 = y_j \oplus y_i$ and $p_2 = y_j \oplus y_i$
  - Hence $p_1 = p_2$, contradiction since all p_i distinct strings

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \rightarrow y_j$ and p_2 also maps $y_i \rightarrow y_j$
  - Then $y_j = y_i \oplus p_1$ and $y_j = y_i \oplus p_2$
  - But then, $p_1 = y_j \oplus y_i$ and $p_2 = y_j \oplus y_i$
  - Hence $p_1 = p_2$, contradiction since all p_i distinct strings
  - Each column is just a permutation of the y_i's

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \rightarrow y_j$ and p_2 also maps $y_i \rightarrow y_j$
  - Then $y_j = y_i \oplus p_1$ and $y_j = y_i \oplus p_2$
  - But then, $p_1 = y_j \oplus y_i$ and $p_2 = y_j \oplus y_i$
  - Hence $p_1 = p_2$, contradiction since all p_i distinct strings
  - Each column is just a permutation of the y_i's
- If $x \in L$, every column has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's

Some Properties of these Permutations

Because each row is just a permutation of y_i's, we have

- If $x \in L$, every row has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every row has $\leq (2^{-n}) \cdot 2^m$ 1's
- No string y_i is mapped to the same string y_j twice
  - Suppose p_1 maps $y_i \rightarrow y_j$ and p_2 also maps $y_i \rightarrow y_j$
  - Then $y_j = y_i \oplus p_1$ and $y_j = y_i \oplus p_2$
  - But then, $p_1 = y_j \oplus y_i$ and $p_2 = y_j \oplus y_i$
  - Hence $p_1 = p_2$, contradiction since all p_i distinct strings
  - Each column is just a permutation of the y_i's
- If $x \in L$, every column has $\geq (1 - 2^{-n}) \cdot 2^m$ 1's
- If $x \notin L$, every column has $\leq (2^{-n}) \cdot 2^m$ 1's

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Proof

- We know each row has at most 2^{m-n} 0's.

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Proof

- We know each row has at most 2^{m-n} 0's.
- For a fixed row r, there are $\binom{2^{m-n}}{m}$ ways to choose m columns such that each one of them contains a 0 for that row

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Proof

- We know each row has at most 2^{m-n} 0's.
- For a fixed row r, there are $\binom{2^{m-n}}{m}$ ways to choose m columns such that each one of them contains a 0 for that row
- Therefore, for each row r, there are at most $\binom{2^{m-n}}{m}$ ways to choose m columns that are 'bad' for that row

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Proof

- We know each row has at most 2^{m-n} 0's.
- For a fixed row r, there are $\binom{2^{m-n}}{m}$ ways to choose m columns such that each one of them contains a 0 for that row
- Therefore, for each row r, there are at most $\binom{2^{m-n}}{m}$ ways to choose m columns that are 'bad' for that row
- Therefore, there are at most $\binom{2^{m-n}}{m} \cdot 2^m$ ways to choose m columns that are 'bad' for *some* row

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Proof

- We know each row has at most 2^{m-n} 0's.
- For a fixed row r, there are $\binom{2^{m-n}}{m}$ ways to choose m columns such that each one of them contains a 0 for that row
- Therefore, for each row r, there are at most $\binom{2^{m-n}}{m}$ ways to choose m columns that are 'bad' for that row
- Therefore, there are at most $\binom{2^{m-n}}{m} \cdot 2^m$ ways to choose m columns that are 'bad' for *some* row
- This is an over-count since same set of m columns can be 'bad' for more than one row

Case: $x \in L$

Claim 1

If $x \in L$, there exist m columns such that none of the rows have all 0's in these m columns

Proof

- We know each row has at most 2^{m-n} 0's.
- For a fixed row r, there are $\binom{2^{m-n}}{m}$ ways to choose m columns such that each one of them contains a 0 for that row
- Therefore, for each row r, there are at most $\binom{2^{m-n}}{m}$ ways to choose m columns that are 'bad' for that row
- Therefore, there are at most $\binom{2^{m-n}}{m} \cdot 2^m$ ways to choose m columns that are 'bad' for *some* row
- This is an over-count since same set of m columns can be 'bad' for more than one row
- The number of ways to choose m columns: $\binom{2^m}{m}$

Case: $x \in L$

Pigeon-hole Principle: If the number of ways to choose m columns that are 'bad' for some row is less than the total number of ways to choose m columns, then there must exist m columns that are 'good' for each row

Case: $x \in L$

Pigeon-hole Principle: If the number of ways to choose m columns that are 'bad' for some row is less than the total number of ways to choose m columns, then there must exist m columns that are 'good' for each row

First note that $(\frac{e2^{m-n}}{m})^m \cdot 2^m \leq (\frac{2^m}{m})^m$ since $2e < 2^n$

Case: $x \in L$

Pigeon-hole Principle: If the number of ways to choose m columns that are 'bad' for some row is less than the total number of ways to choose m columns, then there must exist m columns that are 'good' for each row

First note that $(\frac{e2^{m-n}}{m})^m \cdot 2^m \leq (\frac{2^m}{m})^m$ since $2e < 2^n$
Using the following well-known inequality,

$$(\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{e \cdot n}{k})^k$$

we get that $(\frac{2^m}{m})^m \leq \binom{2^m}{m}$ and $\binom{2^{m-n}}{m} \leq (\frac{e2^{m-n}}{m})^m$

Case: $x \in L$

Pigeon-hole Principle: If the number of ways to choose m columns that are 'bad' for some row is less than the total number of ways to choose m columns, then there must exist m columns that are 'good' for each row

First note that $(\frac{e2^{m-n}}{m})^m \cdot 2^m \leq (\frac{2^m}{m})^m$ since $2e < 2^n$
Using the following well-known inequality,

$$(\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{e \cdot n}{k})^k$$

we get that $(\frac{2^m}{m})^m \leq \binom{2^m}{m}$ and $\binom{2^{m-n}}{m} \leq (\frac{e2^{m-n}}{m})^m$

Therefore, we have $\binom{2^{m-n}}{m} \cdot 2^m < \binom{2^m}{m}$

Case: $x \in L$

Pigeon-hole Principle: If the number of ways to choose m columns that are 'bad' for some row is less than the total number of ways to choose m columns, then there must exist m columns that are 'good' for each row

First note that $(\frac{e2^{m-n}}{m})^m \cdot 2^m \leq (\frac{2^m}{m})^m$ since $2e < 2^n$
Using the following well-known inequality,

$$(\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{e \cdot n}{k})^k$$

we get that $(\frac{2^m}{m})^m \leq \binom{2^m}{m}$ and $\binom{2^{m-n}}{m} \leq (\frac{e2^{m-n}}{m})^m$

Therefore, we have $\binom{2^{m-n}}{m} \cdot 2^m < \binom{2^m}{m}$

Hence there exists a way to select m columns such that none of the rows have all 0's in those columns.

Case: $x \notin L$

Claim 2

If $x \notin L$, no matter how we choose our m columns, at least one of the rows will have all 0's.

Case: $x \notin L$

Claim 2

If $x \notin L$, no matter how we choose our m columns, at least one of the rows will have all 0's.

Proof

- We know each column has at most 2^{m-n} 1's.

Case: $x \notin L$

Claim 2

If $x \notin L$, no matter how we choose our m columns, at least one of the rows will have all 0's.

Proof

- We know each column has at most 2^{m-n} 1's.
- So each column picked can only 'cover' at most 2^{m-n} rows

Case: $x \notin L$

Claim 2

If $x \notin L$, no matter how we choose our m columns, at least one of the rows will have all 0's.

Proof

- We know each column has at most 2^{m-n} 1's.
- So each column picked can only 'cover' at most 2^{m-n} rows
- The total number of rows covered with m columns: $2^{m-n} \cdot m$

Case: $x \notin L$

Claim 2

If $x \notin L$, no matter how we choose our m columns, at least one of the rows will have all 0's.

Proof

- We know each column has at most 2^{m-n} 1's.
- So each column picked can only 'cover' at most 2^{m-n} rows
- The total number of rows covered with m columns: $2^{m-n} \cdot m$
- The total number of rows is 2^m. As $m < 2^n$,

$$2^{m-n} \cdot m < 2^m$$

- Therefore, there will always be some rows having all 0's, for *any* choice of m columns

Putting Everything Together

Claim 1: If $x \in L$, we will be able to choose m strings, $y_1, \ldots, y_m$, such that for all m-bit permutation strings p, the machine M will output 1 for at least one of $M(x, y_i \oplus p)$, where $1 \leq i \leq m$

Putting Everything Together

Claim 1: If $x \in L$, we will be able to choose m strings, $y_1, \ldots, y_m$, such that for all m-bit permutation strings p, the machine M will output 1 for at least one of $M(x, y_i \oplus p)$, where $1 \leq i \leq m$

Claim 2: If $x \notin L$, whichever $y_1, \ldots, y_m$ we choose, for at least one particular p, the machine M will output 0 for all of $M(x, y_i \oplus p)$, where $1 \leq i \leq m$

Putting Everything Together

Claim 1: If $x \in L$, we will be able to choose m strings, $y_1, \ldots, y_m$, such that for all m-bit permutation strings p, the machine M will output 1 for at least one of $M(x, y_i \oplus p)$, where $1 \leq i \leq m$

Claim 2: If $x \notin L$, whichever $y_1, \ldots, y_m$ we choose, for at least one particular p, the machine M will output 0 for all of $M(x, y_i \oplus p)$, where $1 \leq i \leq m$

From Claims 1 and 2 we conclude

- $x \in L$ if and only if

$$\exists y_1, \ldots, y_m \;\; \forall p \; (M(x, y_1 \oplus z) \vee \ldots \vee M(x, y_m \oplus p)) = 1$$

- Therefore, $L \in \Sigma_2$, and so **BPP** $\subseteq \Sigma_2$.