Probabilistic Computation

Lecture 13

Understanding BPP

Recap

Recap

- Probabilistic computation

Recap

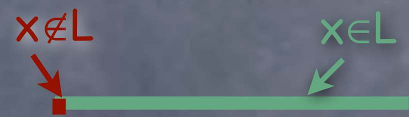
- Probabilistic computation
- NTM (on "random certificates") for L:

Recap

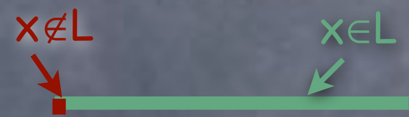
- Probabilistic computation
- NTM (on "random certificates") for L:
  - Pr[M(x)=yes]:

Recap

- Probabilistic computation
- NTM (on "random certificates") for L:
  - Pr[M(x)=yes]:

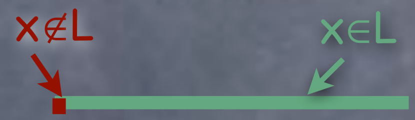


Recap

- Probabilistic computation
- NTM (on "random certificates") for L:
  - Pr[M(x)=yes]: 

- PTM for L: Pr[yes]: 

Recap

- Probabilistic computation
- NTM (on "random certificates") for L:
  - Pr[M(x)=yes]: 

- PTM for L: Pr[yes]: 

- BPTM for L: Pr[yes]: 

Recap

- Probabilistic computation
- NTM (on "random certificates") for L:
  - Pr[M(x)=yes]: 

- PTM for L: Pr[yes]: 

- BPTM for L: Pr[yes]: 

- RTM for L: Pr[yes]: 

Recap

Recap

- PP, RP, co-RP, BPP

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP $\subseteq$ PP

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP $\subseteq$ PP
  - RP, BPP, with bounded gap

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP ⊆ PP
  - RP, BPP, with bounded gap
    - Gap can be boosted from 1/poly to 1-1/exp

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP $\subseteq$ PP
  - RP, BPP, with bounded gap
    - Gap can be boosted from 1/poly to 1-1/exp
    - A realistic/useful computational model

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP ⊆ PP
  - RP, BPP, with bounded gap
    - Gap can be boosted from 1/poly to 1-1/exp
    - A realistic/useful computational model
- Today:

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP $\subseteq$ PP
  - RP, BPP, with bounded gap
    - Gap can be boosted from 1/poly to 1-1/exp
    - A realistic/useful computational model
- Today:
  - NP $\not\subseteq$ BPP, unless PH collapses

Recap

- PP, RP, co-RP, BPP
  - PP too powerful: NP ⊆ PP
  - RP, BPP, with bounded gap
    - Gap can be boosted from 1/poly to 1-1/exp
    - A realistic/useful computational model
- Today:
  - NP ⊄ BPP, unless PH collapses
  - BPP ⊆ $\Sigma_2^P \cap \Pi_2^P$

BPP vs. NP

BPP vs. NP

- Can randomized algorithms efficiently decide all NP problems?

BPP vs. NP

- Can randomized algorithms efficiently decide all NP problems?
  - Unlikely: NP $\subseteq$ BPP $\Rightarrow$ PH = Σ_2^P

BPP vs. NP

- Can randomized algorithms efficiently decide all NP problems?
  - Unlikely: NP $\subseteq$ BPP $\Rightarrow$ PH = Σ_2^P
  - Will show BPP $\subseteq$ P/poly

BPP vs. NP

- Can randomized algorithms efficiently decide all NP problems?
  - Unlikely: NP $\subseteq$ BPP $\Rightarrow$ PH = Σ_2^P
  - Will show BPP $\subseteq$ P/poly
    - Then NP $\subseteq$ BPP $\Rightarrow$ NP $\subseteq$ P/poly

BPP vs. NP

- Can randomized algorithms efficiently decide all NP problems?
  - Unlikely: NP $\subseteq$ BPP $\Rightarrow$ PH = Σ_2^P
  - Will show BPP $\subseteq$ P/poly
    - Then NP $\subseteq$ BPP $\Rightarrow$ NP $\subseteq$ P/poly
      - $\Rightarrow$ PH = Σ_2^P

BPP $\subseteq$ P/poly

BPP ⊆ P/poly

- If error probability is sufficiently small, will show there should be at least one random tape which works for all 2^n inputs of length n

BPP ⊆ P/poly

- If error probability is sufficiently small, will show there should be at least one random tape which works for all 2^n inputs of length n

r \ x						
	✓	✗	✗	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓
	✓	✓	✗	✓	✓	✓
	✓	✓	✓	✓	✗	✓
	✗	✗	✓	✓	✓	✓
	✗	✓	✓	✓	✓	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✗	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓

BPP ⊆ P/poly

- If error probability is sufficiently small, will show there should be at least one random tape which works for all 2^n inputs of length n
  - Then, can give that random tape as advice

r \ x						
	✓	✗	✗	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓
	✓	✓	✗	✓	✓	✓
	✓	✓	✓	✓	✗	✓
	✗	✗	✓	✓	✓	✓
	✗	✓	✓	✓	✓	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✗	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓

BPP $\subseteq$ P/poly

- If error probability is sufficiently small, will show there should be at least one random tape which works for all 2^n inputs of length n
  - Then, can give that random tape as advice
- One such random tape if average (over x) error probability is less than 2^{-n}

r \ x						
	✓	✗	✗	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓
	✓	✓	✗	✓	✓	✓
	✓	✓	✓	✓	✗	✓
	✗	✗	✓	✓	✓	✓
	✗	✓	✓	✓	✓	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✗	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓

BPP ⊆ P/poly

- If error probability is sufficiently small, will show there should be at least one random tape which works for all 2^n inputs of length n
  - Then, can give that random tape as advice
- One such random tape if average (over x) error probability is less than 2^{-n}
  - BPP: can make worst error probability $< 2^{-n}$

r \ x						
	✓	✗	✗	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓
	✓	✓	✗	✓	✓	✓
	✓	✓	✓	✓	✗	✓
	✗	✗	✓	✓	✓	✓
	✗	✓	✓	✓	✓	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✗	✓
	✓	✓	✓	✗	✓	✓
	✓	✓	✓	✓	✓	✓
	✓	✓	✓	✓	✓	✗
	✓	✓	✓	✓	✓	✓

BPP vs. PH

BPP vs. PH

- BPP $\subseteq \Sigma_2^P$

BPP vs. PH

- BPP $\subseteq \Sigma_2^P$
  - So BPP $\subseteq \Sigma_2^P \cap \Pi_2^P$

$BPP \subseteq \Sigma_2^P$

$BPP \subseteq \Sigma_2^P$

- $x \in L$: "for almost all" r, M(x,r)=yes

BPP $\subseteq \Sigma_2^P$

- $x \in L$: "for almost all" r, M(x,r)=yes
- $x \notin L$: M(x,r)=yes for very few r

BPP $\subseteq \Sigma_2^P$

- $x \in L$: "for almost all" r, M(x,r)=yes
- $x \notin L$: M(x,r)=yes for very few r
- L = { x| for almost all r, M(x,r)=yes }

BPP $\subseteq \Sigma_2^P$

- x∈L: "for almost all" r, M(x,r)=yes
- x∉L: M(x,r)=yes for very few r
- L = { x| for almost all r, M(x,r)=yes }
  - If it were "for all", in coNP

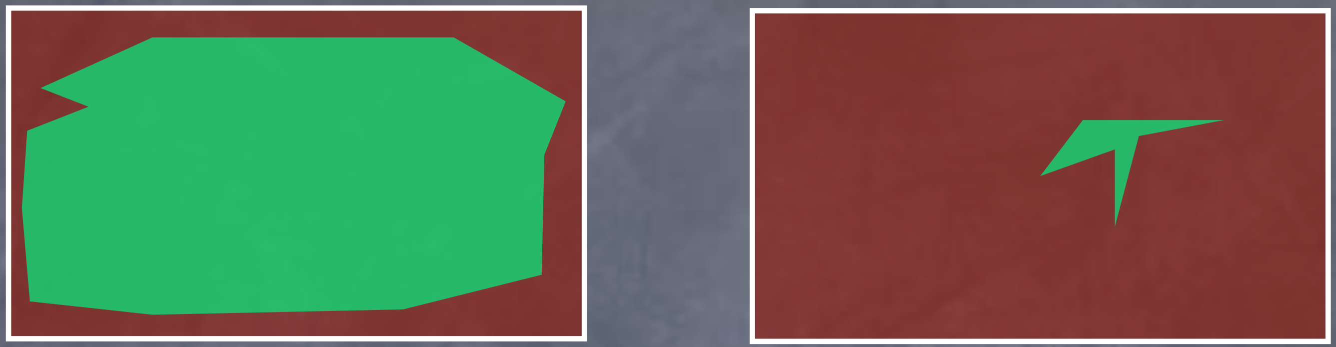
BPP $\subseteq \Sigma_2^P$

- x∈L: "for almost all" r, M(x,r)=yes
- x∉L: M(x,r)=yes for very few r
- L = { x| for almost all r, M(x,r)=yes }
  - If it were "for all", in coNP
  - L = { x| ∃a small "neighborhood", ∀z, for some r "near" z, M(x,r)=yes }

$BPP \subseteq \Sigma_2^P$

- $x \in L$: "for almost all" r, M(x,r)=yes
- $x \notin L$: M(x,r)=yes for very few r
- L = { x| for almost all r, M(x,r)=yes }
  - If it were "for all", in coNP
  - L = { x| $\exists$a small "neighborhood", $\forall$z, for some r "near" z, M(x,r)=yes }
    - Note: Neighborhood of z is small (polynomially large), so can go through all of them in polynomial time

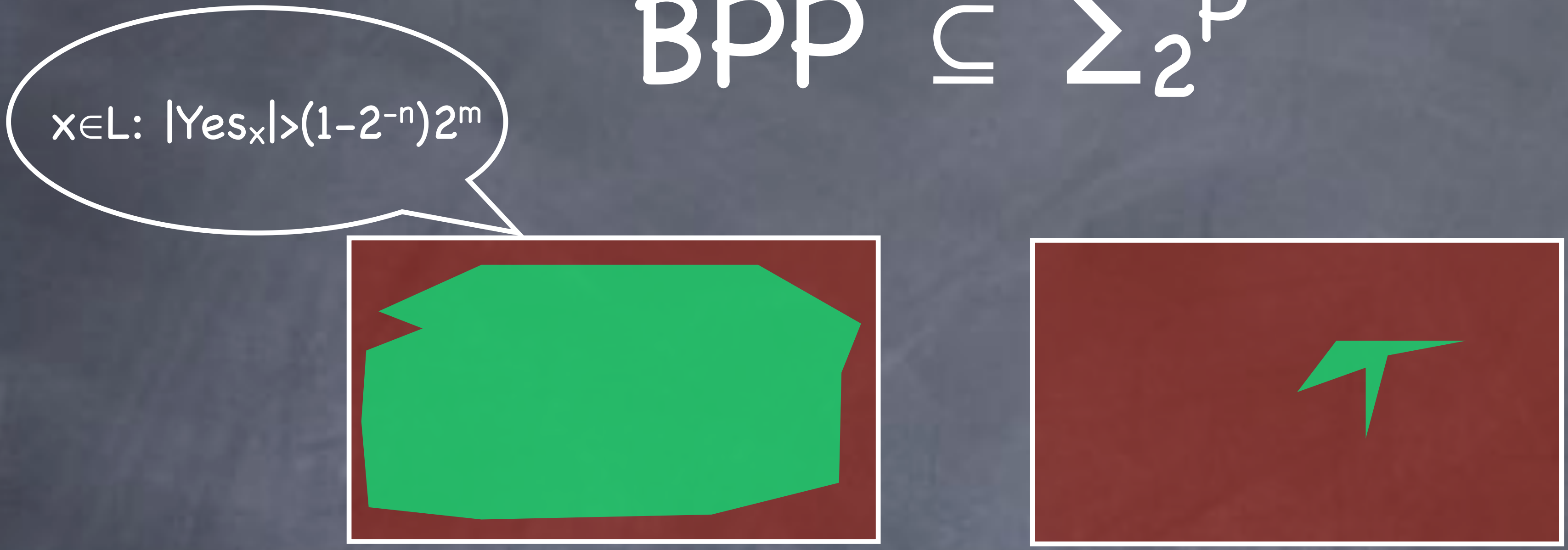
$BPP \subseteq \Sigma_2^P$



Space of random tapes = $\{0,1\}^m$

Yes_x = {r| M(x,r)=yes }

$BPP \subseteq \Sigma_2^P$



Space of random tapes = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r)=yes\}$

BPP $\subseteq \Sigma_2^P$

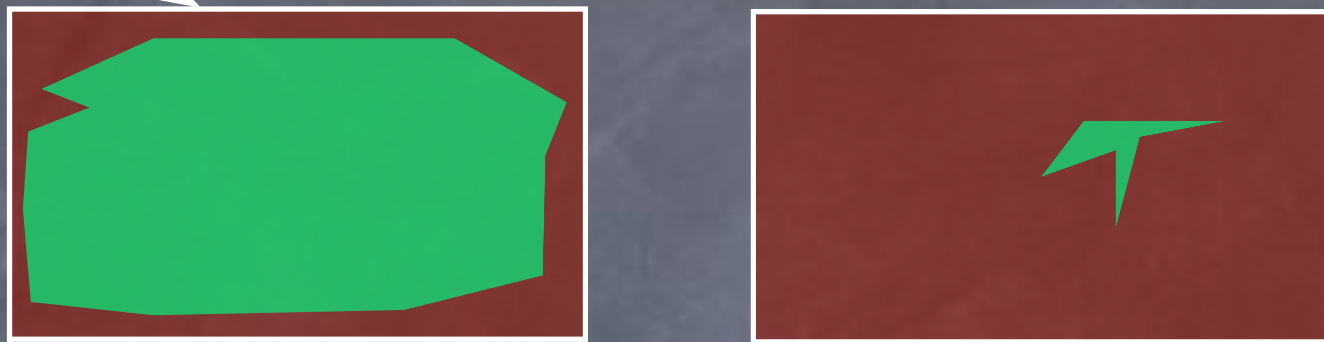
Space of random tapes = $\{0,1\}^m$

$Yes_x = \{r | M(x,r)=yes\}$

BPP $\subseteq \Sigma_2^P$

$x \in L$: $|Yes_x| > (1-2^{-n})2^m$

$x \notin L$: $|Yes_x| < 2^{-n}2^m$



Space of random tapes = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r)=yes\}$

- $x \in L$: Will show that there exist a small set of shifts of Yes_x that cover all z

$BPP \subseteq \Sigma_2^P$

Space of random tapes = $\{0,1\}^m$

$Yes_x = \{r | M(x,r)=yes\}$

- $x \in L$: Will show that there exist a small set of shifts of Yes_x that cover all z
  - If z is a shift of $r \in Yes_x$, r is in the neighborhood of z

$BPP \subseteq \Sigma_2^P$

Space of random tapes = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r)=yes\}$

- $x \in L$: Will show that there exist a small set of shifts of Yes_x that cover all z
  - If z is a shift of $r \in Yes_x$, r is in the neighborhood of z
- $x \notin L$: Yes_x very small, so its few shifts cover only a small region

BPP $\subseteq \Sigma_2^P$

BPP ⊆ Σ_2^P

- "A small set of shifts": $P = \{u_1, u_2, \ldots, u_k\}$

$BPP \subseteq \Sigma_2^P$

- "A small set of shifts": $P = \{u_1, u_2, \ldots, u_k\}$
  - $P(r) = \{ r \oplus u_1, r \oplus u_2, \ldots, r \oplus u_k\}$ where r, u_i are m-bit strings, and k is "small" (poly(n))

$BPP \subseteq \Sigma_2^P$

- "A small set of shifts": $P = \{u_1,u_2,\ldots,u_k\}$
  - $P(r)=\{ r\oplus u_1,r\oplus u_2,\ldots,r\oplus u_k\}$ where r, u_i are m-bit strings, and k is "small" (poly(n))
- For each $x \in L$, does there exist a P s.t. $P(Yes_x) := \cup_{r\in Yes(x)} P(r) = \{0,1\}^m$?

BPP $\subseteq \Sigma_2^P$

- "A small set of shifts": $P = \{u_1,u_2,\ldots,u_k\}$
  - $P(r)=\{ r\oplus u_1,r\oplus u_2,\ldots,r\oplus u_k\}$ where r, u_i are m-bit strings, and k is "small" (poly(n))
- For each $x\in L$, does there exist a P s.t. $P(Yes_x) := \cup_{r\in Yes(x)} P(r) = \{0,1\}^m$?
  - Yes! For all large S (like Yes_x) can indeed find a P s.t. $P(S) = \{0,1\}^m$

BPP $\subseteq \Sigma_2^P$

- "A small set of shifts": $P = \{u_1, u_2, \ldots, u_k\}$
  - $P(r) = \{ r \oplus u_1, r \oplus u_2, \ldots, r \oplus u_k\}$ where r, u_i are m-bit strings, and k is "small" (poly(n))
- For each $x \in L$, does there exist a P s.t. $P(Yes_x) := \cup_{r \in Yes(x)} P(r) = \{0,1\}^m$?
  - Yes! For all large S (like Yes_x) can indeed find a P s.t. $P(S) = \{0,1\}^m$
    - In fact, most P work (if k big enough)!

BPP $\subseteq \Sigma_2^P$

BPP $\subseteq \Sigma_2^P$

- Probabilistic Method (finding hay in haystack)

BPP $\subseteq \Sigma_2^P$

- Probabilistic Method (finding hay in haystack)
  - To prove $\exists P$ with some property

BPP ⊆ Σ_2^P

- Probabilistic Method (finding hay in haystack)
  - To prove ∃P with some property
  - Define a probability distribution over all candidate P's and prove that the property holds with positive probability (often even close to one)

BPP $\subseteq \Sigma_2^P$

- Probabilistic Method (finding hay in haystack)
  - To prove ∃P with some property
  - Define a probability distribution over all candidate P's and prove that the property holds with positive probability (often even close to one)
    - Distribution s.t. easy to prove positive probability of property holding

BPP $\subseteq \Sigma_2^P$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1, u_2, \ldots, u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$

BPP $\subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(over\ P)}[P(S) \neq \{0,1\}^m] = Pr_{(over\ P)}[\exists z\ z \notin P(S)]$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(over\ P)}[P(S) \neq \{0,1\}^m] = Pr_{(over\ P)}[\exists z\ z \notin P(S)]$
    $\leq \Sigma_z\ Pr_{(over\ P)}[z \notin P(S)]$

BPP $\subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(\text{over } P)}[P(S) \neq \{0,1\}^m] = Pr_{(\text{over } P)}[\exists z\ z \notin P(S)]$
    $\leq \Sigma_z\ Pr_{(\text{over } P)}[z \notin P(S)] \quad = \Sigma_z\ Pr_{(\text{over } u1..uk)}[\forall i \quad z \oplus u_i \notin S]$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1, u_2, \ldots, u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(over\ P)}[P(S) \neq \{0,1\}^m] = Pr_{(over\ P)}[\exists z\ z \notin P(S)]$

    $\leq \Sigma_z\ Pr_{(over\ P)}[z \notin P(S)] \quad = \Sigma_z\ Pr_{(over\ u1..uk)}[\forall i\quad z \oplus u_i \notin S]$

    $= \Sigma_z\ \Pi_i\ \ Pr_{\ (over\ ui)}[z \oplus u_i\ \notin S]$

$\mathrm{BPP} \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $\Pr_{(\text{over } P)}[P(S) \neq \{0,1\}^m] = \Pr_{(\text{over } P)}[\exists z\ z \notin P(S)]$

    $\leq \Sigma_z \Pr_{(\text{over } P)}[z \notin P(S)] \quad = \Sigma_z \Pr_{(\text{over } u1..uk)}[\forall i \quad z \oplus u_i \notin S]$

    $= \Sigma_z \Pi_i \Pr_{(\text{over } ui)}[z \oplus u_i \notin S] = \Sigma_z \Pi_i \Pr_{(\text{over } ui)}[u_i \notin z \oplus S]$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(\text{over }P)}[P(S) \neq \{0,1\}^m] = Pr_{(\text{over }P)}[\exists z\ z \notin P(S)]$

    $\leq \Sigma_z\ Pr_{(\text{over }P)}[z \notin P(S)] \quad = \Sigma_z\ Pr_{(\text{over }u1..uk)}[\forall i \quad z \oplus u_i \notin S]$

    $= \Sigma_z\ \Pi_i\ Pr_{(\text{over }ui)}[z \oplus u_i \notin S] = \Sigma_z\ \Pi_i\ Pr_{(\text{over }ui)}[u_i \notin z \oplus S]$

    $= \Sigma_z\ \Pi_i\ (|S^c|/2^m)$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(\text{over } P)}[P(S) \neq \{0,1\}^m] = Pr_{(\text{over } P)}[\exists z\ z \notin P(S)]$

    $\leq \Sigma_z\ Pr_{(\text{over } P)}[z \notin P(S)] \quad = \Sigma_z\ Pr_{(\text{over } u1..uk)}[\forall i\ \ z \oplus u_i \notin S]$

    $= \Sigma_z\ \Pi_i\ \ Pr_{(\text{over } ui)}[z \oplus u_i \notin S] = \Sigma_z\ \Pi_i\ \ Pr_{(\text{over } ui)}[u_i \notin z \oplus S]$

    $= \Sigma_z\ \Pi_i\ (|S^c|/2^m) \quad < \Sigma_z\ \Pi_i\ 2^{-n}$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1, u_2, \ldots, u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(over\ P)}[P(S) \neq \{0,1\}^m] = Pr_{(over\ P)}[\exists z\ z \notin P(S)]$

    $\leq \Sigma_z\ Pr_{(over\ P)}[z \notin P(S)] \quad = \Sigma_z\ Pr_{(over\ u1..uk)}[\forall i \quad z \oplus u_i \notin S]$

    $= \Sigma_z\ \Pi_i\ Pr_{(over\ ui)}[z \oplus u_i \notin S] = \Sigma_z\ \Pi_i\ Pr_{(over\ ui)}[u_i \notin z \oplus S]$

    $= \Sigma_z\ \Pi_i\ (|S^c|/2^m) \quad < \Sigma_z\ \Pi_i\ 2^{-n} \quad = 2^m.(2^{-n})^k = 1$

$BPP \subseteq \Sigma_2^P$

- Probabilistic method to find $P = \{u_1,u_2,\ldots,u_k\}$, s.t. for all large S, $P(S) = \{0,1\}^m$
  - Distribution over P's: randomized experiment to generate P
    - Pick each u_i independently, and uniformly at random from $\{0,1\}^m$
  - $Pr_{(over\ P)}[P(S) \neq \{0,1\}^m] = Pr_{(over\ P)}[\exists z\ z \notin P(S)]$

    $\leq \Sigma_z\ Pr_{(over\ P)}[z \notin P(S)] \quad = \Sigma_z\ Pr_{(over\ u1..uk)}[\forall i \quad z \oplus u_i \notin S]$

    $= \Sigma_z\ \Pi_i\ Pr_{(over\ ui)}[z \oplus u_i \notin S] = \Sigma_z\ \Pi_i\ Pr_{(over\ ui)}[u_i \notin z \oplus S]$

    $= \Sigma_z\ \Pi_i\ (|S^c|/2^m) \quad < \Sigma_z\ \Pi_i\ 2^{-n} \quad = 2^m.(2^{-n})^k = 1$
  - So (with $|S|>(1-2^{-n})2^m$ and $k=m/n$), $\exists P$, $P(S) = \{0,1\}^m$

$BPP \subseteq \Sigma_2^P$

Space of random strings = $\{0,1\}^m$

Yes_x = {r| M(x,r)=yes }

$BPP \subseteq \Sigma_2^P$

Space of random strings = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r)=yes\}$

- For each $x \in L$, $\exists P$ (of size $k=m/n$) s.t. $P(Yes_x)=\{0,1\}^m$

$BPP \subseteq \Sigma_2^P$

$x \in L$: $|Yes_x| > (1-2^{-n})2^m$

$x \notin L$: $|Yes_x| < 2^{-n}2^m$

Space of random strings = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r)=yes\}$

- For each $x \in L$, $\exists P$ (of size $k=m/n$) s.t. $P(Yes_x)=\{0,1\}^m$
- For each $x \notin L$, $P(Yes_x) \subsetneq \{0,1\}^m$

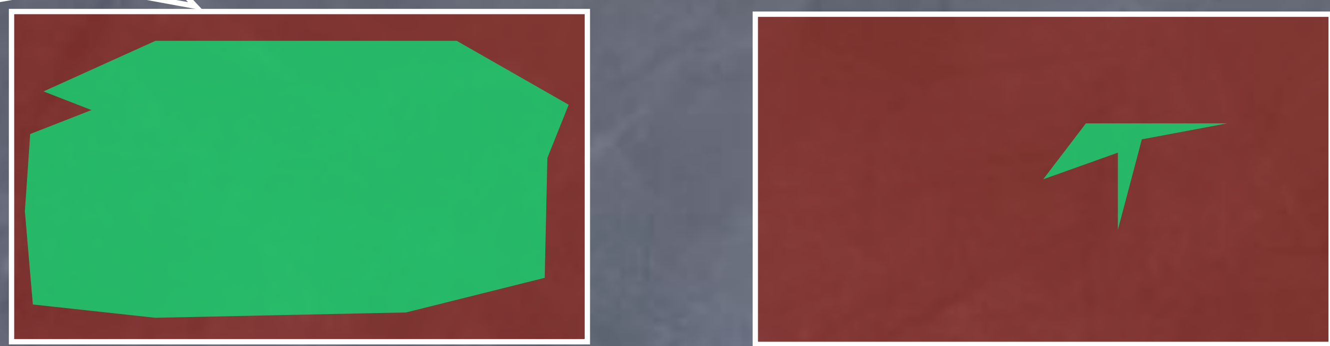
$BPP \subseteq \Sigma_2^P$

Space of random strings = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r) = yes\}$

- For each $x \in L$, $\exists P$ (of size $k=m/n$) s.t. $P(Yes_x) = \{0,1\}^m$
- For each $x \notin L$, $P(Yes_x) \subsetneq \{0,1\}^m$
  - $|P(Yes_x)| \leq k|Yes_x| = (m/n)\ 2^{-n}2^m < 2^m$

$BPP \subseteq \Sigma_2^P$



Space of random strings = $\{0,1\}^m$

$Yes_x = \{r \mid M(x,r)=yes\}$

- For each $x \in L$, $\exists P$ (of size $k=m/n$) s.t. $P(Yes_x)=\{0,1\}^m$
- For each $x \notin L$, $P(Yes_x) \subsetneq \{0,1\}^m$
  - $|\ P(Yes_x)\ | \leq k|\ Yes_x\ | = (m/n)\ 2^{-n}2^m < 2^m$
- $L = \{\ x \mid \exists P\ \forall z$ for some $r \in P^{-1}(z)\ M(x,r)=yes\ \}$

BPP-Complete Problem?

BPP-Complete Problem?

- Not known!

BPP-Complete Problem?

- Not known!
  - L = { $(M,x,1^t)$ | M(x)=yes in time t with probability > 2/3} ?

BPP-Complete Problem?

- Not known!
  - L = { $(M,x,1^t)$ | M(x)=yes in time t with probability > 2/3} ?
  - Is indeed BPP-Hard

BPP-Complete Problem?

- Not known!
  - L = { $(M,x,1^t)$ | M(x)=yes in time t with probability > 2/3} ?
  - Is indeed BPP-Hard
  - But in BPP?

BPP-Complete Problem?

- Not known!
  - L = { $(M,x,1^t)$ | M(x)=yes in time t with probability > 2/3} ?
  - Is indeed BPP-Hard
  - But in BPP?
    - Just run M(x) for t steps and accept if it accepts?

BPP-Complete Problem?

- Not known!
  - $L = \{ (M,x,1^t) \mid M(x)=\text{yes in time } t \text{ with probability} > 2/3\}$?
  - Is indeed BPP-Hard
  - But in BPP?
    - Just run M(x) for t steps and accept if it accepts?
      - If $(M,x,1^t)$ in L, we will indeed accept with prob. > 2/3

BPP-Complete Problem?

- Not known!
  - $L = \{ (M,x,1^t) \mid M(x)=\text{yes in time } t \text{ with probability} > 2/3\}$?
  - Is indeed BPP-Hard
  - But in BPP?
    - Just run M(x) for t steps and accept if it accepts?
      - If $(M,x,1^t)$ in L, we will indeed accept with prob. > 2/3
      - But M may not have a bounded gap. Then, if $(M,x,1^t)$ not in L, we may accept with prob. very close to 2/3.

BPTIME-Hierarchy Theorem?

BPTIME-Hierarchy Theorem?

- $\mathrm{BPTIME}(n) \subsetneq \mathrm{BPTIME}(n^{100})$?

BPTIME-Hierarchy Theorem?

- BPTIME(n) $\subsetneq$ BPTIME(n^{100})?
- Not known!

BPTIME-Hierarchy Theorem?

- BPTIME(n) $\subsetneq$ BPTIME(n^{100})?
- Not known!
  - But is true for BPTIME(T)/1

Today

Today

- Probabilistic computation

Today

- Probabilistic computation
- BPP ⊆ P/poly (so if NP ⊆ BPP, then PH=Σ_2^P)

Today

- Probabilistic computation
- BPP $\subseteq$ P/poly (so if NP $\subseteq$ BPP, then PH=Σ_2^P)
- BPP $\subseteq$ $\Sigma_2^P \cap \Pi_2^P$

Today

- Probabilistic computation
- BPP $\subseteq$ P/poly (so if NP $\subseteq$ BPP, then PH=Σ_2^P)
- BPP $\subseteq$ $\Sigma_2^P \cap \Pi_2^P$
- Coming up

Today

- Probabilistic computation
- BPP $\subseteq$ P/poly (so if NP $\subseteq$ BPP, then PH=Σ_2^P)
- BPP $\subseteq$ $\Sigma_2^P \cap \Pi_2^P$
- Coming up
  - Basic randomized algorithmic techniques

Today

- Probabilistic computation
- BPP $\subseteq$ P/poly (so if NP $\subseteq$ BPP, then PH=Σ_2^P)
- BPP $\subseteq$ $\Sigma_2^P \cap \Pi_2^P$
- Coming up
  - Basic randomized algorithmic techniques
  - Saving on randomness