Complexity of Counting

Lecture 21

#P: Toda’s Theorem
Last Time
Last Time

#P: counting problems of the form \( \#R(x) = |\{w: R(x,w)=1\}| \)
where \( R \) is a polynomial time relation
Last Time

- **#P**: counting problems of the form $\#R(x) = |\{w: R(x,w)=1\}|$, where $R$ is a polynomial time relation.

  - Can be hard: even $\#\text{CYCLE}$ is not in FP (unless P = NP).
Last Time

- \#P: counting problems of the form \( \#R(x) = |\{w: R(x,w)=1\}| \), where \( R \) is a polynomial time relation
- Can be hard: even \#\text{CYCLE} is not in FP (unless P = NP)
- \( \#P \subseteq FP^{PP} \) (and \( PP \subseteq P^{#P} \))
Last Time

- **#P**: counting problems of the form \( #R(x) = |\{w: R(x,w)=1\}| \), where \( R \) is a polynomial time relation

- Can be hard: even \( \#\text{CYCLE} \) is not in \( \text{FP} \) (unless \( P = NP \))

- \( \#P \subseteq \text{FP}^{\text{PP}} \) (and \( \text{PP} \subseteq P^{\#P} \))

- \( \#P \) complete problems
Last Time

- **#P**: counting problems of the form \( \#R(x) = |\{w: R(x,w)=1\}| \), where \( R \) is a polynomial time relation

- Can be hard: even #CYCLE is not in FP (unless P = NP)

- \( #P \subseteq FP^{PP} \) (and \( PP \subseteq P^{#P} \))

- #P complete problems

- #SAT
Last Time

- **#P**: counting problems of the form $#R(x) = \left| \{w: R(x,w)=1\} \right|$, where $R$ is a polynomial time relation.
- Can be hard: even #$\text{CYCLE}$ is not in FP (unless $P = NP$).
- #$P \subseteq FP^{PP}$ (and $PP \subseteq P^{#P}$).
- #$P$ complete problems.
- #$\text{SAT}$
- Permanent
Last Time

- **#P**: counting problems of the form $\#R(x) = |\{w: R(x,w)=1\}|$, where $R$ is a polynomial time relation

  - Can be hard: even $\#\text{CYCLE}$ is not in FP (unless $P = NP$)

- $\#P \subseteq \text{FP}^{PP}$ (and $\text{PP} \subseteq P^{#P}$)

- $\#P$ complete problems

- $\#\text{SAT}$

- Permanent

Next: Toda’s Theorem: $\text{PH} \subseteq P^{#P} = P^{PP}$
⊕P

⊕P: parity of the number of witnesses
⊕P

⊕P: parity of the number of witnesses

e.g. ⊕SAT. Least significant bit of #SAT.
\( \oplus P \)

- \( \oplus P \): **parity** of the number of witnesses
  - e.g. \( \oplus SAT \). Least significant bit of \#SAT.
  - May not be as powerful as PP (or \#P)
\( \oplus P \)

- \( \oplus P \): **parity** of the number of witnesses
- e.g. \( \oplus \text{SAT} \). Least significant bit of \( \#\text{SAT} \).
- May not be as powerful as \( \text{PP} \) (or \( \#P \))
- \( \oplus P \subseteq P \) may not imply \( \text{NP} = P \)
\[\oplus P\]

\[\oplus P: \text{parity of the number of witnesses}\]

\[\text{e.g. } \oplus \text{SAT. Least significant bit of } \#\text{SAT.}\]

\[\text{May not be as powerful as PP (or } \#P)\]

\[\oplus P \subseteq P \text{ may not imply } NP = P\]

\[\text{But it does imply } NP \subseteq RP \text{ (even if only } \oplus P \subseteq RP)\]
⊕P

⊕P: parity of the number of witnesses

e.g. ⊕SAT. Least significant bit of #SAT.

May not be as powerful as PP (or #P)

⊕P ⊆ P may not imply NP = P

But it does imply \( NP \subseteq \text{RP} \) (even if only \( ⊕P \subseteq \text{RP} \))

Randomized reduction of NP to ⊕P
\[ \oplus P \]

\[ \oplus P: \text{parity of the number of witnesses} \]

e.g. \( \oplus \text{SAT} \). Least significant bit of \#SAT.

May not be as powerful as \( \text{PP} \) (or \#P)

\( \oplus P \subseteq P \) may not imply \( \text{NP} = P \)

But it does imply \( \text{NP} \subseteq \text{RP} \) (even if only \( \oplus P \subseteq \text{RP} \))

Randomized reduction of \( \text{NP} \) to \( \oplus P \)

i.e., \( \oplus P \) oracle is quite useful to randomized algorithms
⊕P

⊕P: parity of the number of witnesses

e.g. ⊕SAT. Least significant bit of #SAT.

May not be as powerful as PP (or #P)

⊕P ⊆ P may not imply NP = P

But it does imply NP ⊆ RP (even if only ⊕P ⊆ RP)

Randomized reduction of NP to ⊕P

i.e., ⊕P oracle is quite useful to randomized algorithms
ΘP ⊆ RP \implies NP=RP
\[ \oplus P \subseteq RP \Rightarrow NP=RP \]

Randomized reduction of NP to \( \oplus P \)
$\oplus P \subseteq \text{RP} \Rightarrow \text{NP=RP}$

Randomized reduction of NP to $\oplus P$

A probabilistic polynomial time algorithm $A$ such that
\[ \oplus P \subseteq \text{RP} \implies \text{NP} = \text{RP} \]

- Randomized reduction of \text{NP} to \( \oplus P \)

- A probabilistic polynomial time algorithm \( A \) such that
  \[ \varphi \notin \text{SAT} \implies \Pr[A(\varphi) \in \oplus \text{SAT}] = 0 \]
\[ \Theta P \subseteq RP \Rightarrow NP=RP \]

- Randomized reduction of NP to \( \Theta P \)

- A probabilistic polynomial time algorithm A such that
  - \( \varphi \notin SAT \Rightarrow Pr[A(\varphi) \in \oplus SAT] = 0 \)
  - In fact A(\varphi) will have no satisfying assignment
$\oplus P \subseteq RP \Rightarrow NP=RP$

- Randomized reduction of $NP$ to $\oplus P$

- A probabilistic polynomial time algorithm $A$ such that

  - $\varphi \notin SAT \Rightarrow Pr[A(\varphi) \in \oplus SAT] = 0$
  
  - In fact $A(\varphi)$ will have no satisfying assignment

  - $\varphi \in SAT \Rightarrow Pr[A(\varphi) \in \oplus SAT] \geq \epsilon(n)$
\[ \oplus P \subseteq \text{RP} \Rightarrow \text{NP=RP} \]

- Randomized reduction of NP to \( \oplus P \)

- A probabilistic polynomial time algorithm \( A \) such that:
  - \( \varphi \notin \text{SAT} \Rightarrow \Pr[A(\varphi) \in \oplus \text{SAT}] = 0 \)
  - In fact \( A(\varphi) \) will have no satisfying assignment
  - \( \varphi \in \text{SAT} \Rightarrow \Pr[A(\varphi) \in \oplus \text{SAT}] \geq \varepsilon(n) \)
  - With prob. \( \geq \varepsilon(n) \), \( A(\varphi) \) will have exactly one satisfying assignment
⊕P ⊆ RP \Rightarrow \text{NP=RP}

Randomized reduction of NP to ⊕P

A probabilistic polynomial time algorithm A such that

φ \notin \text{SAT} \Rightarrow \Pr[A(φ) \in ⊕\text{SAT}] = 0

In fact A(φ) will have no satisfying assignment

φ \in \text{SAT} \Rightarrow \Pr[A(φ) \in ⊕\text{SAT}] \geq \varepsilon(n)

With prob. \geq \varepsilon(n), A(φ) will have exactly one satisfying assignment

If an RP algorithm for ⊕\text{SAT}, then an RP algorithm for \text{SAT}
$\Theta P \subseteq RP \Rightarrow NP=RP$
$\Theta P \subseteq RP \Rightarrow NP=RP$

Randomized reduction of SAT to Unique-SAT: A probabilistic polynomial time algorithm $A$ such that
\[ \Theta P \subseteq RP \Rightarrow NP=RP \]

Randomized reduction of SAT to Unique-SAT: A probabilistic polynomial time algorithm \( A \) such that

- If \( \varphi \in SAT \), with prob. \( \geq \varepsilon(n) \), \( A_\varphi \) will have exactly one satisfying assignment. Else \( A_\varphi \) will have none.
Randomized reduction of SAT to Unique-SAT: A probabilistic polynomial time algorithm $A$ such that

- If $\varphi \in \text{SAT}$, with prob. $\geq \varepsilon(n)$, $A_\varphi$ will have exactly one satisfying assignment. Else $A_\varphi$ will have none.

- Add a filter which will pass exactly one witness (if any): $A_\varphi(w) = \varphi(w)$ and $\text{filter}(w)$
Hashing for unique preimage
Hashing for unique preimage

Let \( S \subseteq X \) be a set of size \( m \). Consider a pair-wise independent hash-function family \( H \), from \( X \) to \( \mathbb{R} \), such that \( |S|/|R| \in [1/4, 1/2] \).
Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $\mathbb{R}$, such that $|S|/|R| \in [1/4,1/2]$.

$\Pr[h(x) = 0] = 1/|R| =: p$, and $\Pr[h(x) = h(y) = 0] = p^2$. $|S|p \in [1/4,1/2]$. 

Hashing for unique preimage
Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4,1/2]$.

- $\Pr_h[h(x)=0] = 1/|R| =: p$, and $\Pr_h[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4,1/2]$.

- Let $N := |\{x \in S| h(x)=0\}|$. $\Pr_h[N=1] = \Pr_h[N \geq 1] - \Pr_h[N \geq 2]$.

Hashing for unique preimage
Hashing for unique preimage

Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4,1/2]$.

- $\Pr_h[h(x)=0] = 1/|R| =: p$, and $\Pr_h[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4,1/2]$.

- Let $N := |\{x \in S| h(x)=0\}|$. $\Pr_h[N=1] = \Pr_h[N \geq 1] - \Pr_h[N \geq 2]$

- By inclusion-exclusion: $\Pr_h[N \geq 1] \geq \Sigma_x \Pr_h[h(x)=0] - \Sigma_{x>y} \Pr_h[h(x)=h(y)=0]$
Hashing for unique preimage

Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4, 1/2]$.

- $Pr_{h}[h(x)=0] = 1/|R| =: p$, and $Pr_{h}[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4, 1/2]$.

- Let $N := |\{x \in S : h(x)=0\}|$. $Pr_{h}[N=1] = Pr_{h}[N\geq 1] - Pr_{h}[N\geq 2]$

- By inclusion-exclusion: $Pr_{h}[N\geq 1] \geq \Sigma_x Pr_{h}[h(x)=0] - \Sigma_{x>y} Pr_{h}[h(x)=h(y)=0] - |S|p^2$
Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4,1/2]$.

$\Pr_{h}[h(x)=0] = 1/|R| =: p$, and $\Pr_{h}[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4,1/2]$.

Let $N := |\{x \in S| h(x)=0\}|$. $\Pr_{h}[N=1] = \Pr_{h}[N\geq1] - \Pr_{h}[N\geq2]$

By inclusion-exclusion: $\Pr_{h}[N\geq1] \geq \Sigma_x \Pr_{h}[h(x)=0] - \Sigma_{x>y} \Pr_{h}[h(x)=h(y)=0]$
Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4,1/2]$.

$Pr_h[h(x)=0] = 1/|R| =: p$, and $Pr_h[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4,1/2]$.

Let $N := |\{x \in S| h(x)=0\}|$. $Pr_h[N=1] = Pr_h[N\geq 1] - Pr_h[N\geq 2]$

By inclusion-exclusion: $Pr_h[N\geq 1] \geq \sum_x Pr_h[h(x)=0] - \sum_{x>y} Pr_h[h(x)=h(y)=0]$

By Union-bound: $Pr_h[N\geq 2] \leq \sum_{x>y} Pr_h[h(x)=h(y)=0]$
Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4,1/2]$.

- $Pr_h[h(x)=0] = 1/|R| =: p$, and $Pr_h[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4,1/2]$.

- Let $N := |\{x \in S| h(x)=0\}|$. $Pr_h[N=1] = Pr_h[N\geq 1] - Pr_h[N\geq 2]$

By inclusion-exclusion:

- $Pr_h[N\geq 1] \geq \sum_x Pr_h[h(x)=0] - \sum_{x>y} Pr_h[h(x)=h(y)=0]$

By Union-bound:

- $Pr_h[N\geq 2] \leq \sum_{x>y} Pr_h[h(x)=h(y)=0]$
Hashing for unique preimage

Let $S \subseteq X$ be a set of size $m$. Consider a pair-wise independent hash-function family $H$, from $X$ to $R$, such that $|S|/|R| \in [1/4, 1/2]$.

$\Pr_h[h(x)=0] = 1/|R| = p$, and $\Pr_h[h(x)=h(y)=0] = p^2$. $|S|p \in [1/4, 1/2]$.

Let $N := |\{x \in S \mid h(x)=0\}|$. $\Pr_h[N=1] = \Pr_h[N \geq 1] - \Pr_h[N \geq 2]$

By inclusion-exclusion: $\Pr_h[N \geq 1] \geq \sum_x \Pr_h[h(x)=0] - \sum_{x>y} \Pr_h[h(x)=h(y)=0]$

By Union-bound: $\Pr_h[N \geq 2] \leq \sum_{x>y} \Pr_h[h(x)=h(y)=0]$

$\Pr_h[N=1] \geq |S|p - 2 \binom{|S|}{2} p^2 \geq |S|p - (|S|p)^2 \geq 3/16$
$\Theta P \subseteq \text{RP } \Rightarrow \text{NP=RP}$

Randomized reduction of SAT to Unique-SAT: A probabilistic polynomial time algorithm $A$ such that

- If $\varphi \in \text{SAT}$, with prob. $\geq \varepsilon(n)$, $A_\varphi$ will have exactly one satisfying assignment. Else $A_\varphi$ will have none.

- Add a filter which will pass exactly one witness (if any): $A_\varphi(w) = \varphi(w) \text{ and filter}(w)$
\( \Theta P \subseteq RP \Rightarrow NP=RP \)

Randomized reduction of SAT to Unique-SAT: A probabilistic polynomial time algorithm \( A \) such that

If \( \phi \in SAT \), with prob. \( \geq \varepsilon(n) \), \( A_{\phi} \) will have exactly one satisfying assignment. Else \( A_{\phi} \) will have none.

Add a filter which will pass exactly one witness (if any):
\[ A_{\phi}(w) = \phi(w) \text{ and } filter(w) \]

\( filter(w) \): a Boolean formula saying \( h(w)=0 \). (If using auxiliary variables, i.e., \( \exists z \ filter(w,z) \), use a parsimonious reduction.)
Randomized reduction of SAT to Unique-SAT: A probabilistic polynomial time algorithm $A$ such that

If $\phi \in \text{SAT}$, with prob. $\geq \epsilon(n)$, $A_{\phi}$ will have exactly one satisfying assignment. Else $A_{\phi}$ will have none.

Add a filter which will pass exactly one witness (if any): $A_{\phi}(w) = \phi(w) \text{ and } \text{filter}(w)$

filter(w): a Boolean formula saying $h(w)=0$. (If using auxiliary variables, i.e., $\exists z \text{ filter}(w,z)$, use a parsimonious reduction.)

If witness $n$-bit long ($|X| = \{0,1\}^n$), pick $R = \{0,1\}^k$, with $k$ random in the range $[1,n]$
Reducing PH to $\mathbb{P}^{\mathbb{NP}}$
Reducing PH to $P^{\#P}$

- Two steps
Reducing PH to $\text{P}^\#\text{P}$

Two steps

Randomized reduction of PH to $\text{P}^\oplus\text{P}$
Reducing PH to $\mathsf{P^{\#P}}$

- Two steps
  - Randomized reduction of PH to $\mathsf{P^{\#P}}$
  - Converting the probabilistic guarantee to a deterministic $\mathsf{\#P}$ statement
Quantifier Gallery!
Quantifier Gallery!

∃

For at least one
Quantifier Gallery!

∃
For at least one

∀
For all
Quantifier Gallery!

\[\exists\]
For at least one

\[\forall\]
For all

\[\exists_r\]
For at least \(r\) fraction
Quantifier Gallery!

∃
For at least one

∀
For all

∃_r
For at least r fraction

∃!
For exactly one
Quantifier Gallery!

- $\exists$ For at least one
- $\forall$ For all
- $\exists_r$ For at least $r$ fraction
- $\exists!$ For exactly one
- For an odd number of
QBF to $\Theta BF$
QBF to $\Theta_B F$

We have a randomized reduction: $\varphi$ to $A_{\varphi}$ such that
QBF to $\oplus BF$

We have a randomized reduction: $\varphi$ to $A_{\varphi}$ such that

$$\exists_w \varphi(w) \Rightarrow \oplus_w A_{\varphi}(w) \text{ with prob. } \geq \epsilon(n)$$
QBF to $\oplus BF$

We have a randomized reduction: $\varphi$ to $A_{\varphi}$ such that

- $\exists_w \varphi(w) \Rightarrow \bigoplus_w A_{\varphi}(w)$ with prob. $\geq \varepsilon(n)$
- $\forall_w \neg \varphi(w) \Rightarrow \neg \bigoplus_w A_{\varphi}(w)$ (with prob. = 1)
We have a randomized reduction: $\varphi$ to $A_{\varphi}$ such that:

- $\exists_w \varphi(w) \Rightarrow \bigoplus_w A_{\varphi}(w)$ with prob. $\geq \epsilon(n)$
- $\forall_w \text{not } \varphi(w) \Rightarrow \text{not } \bigoplus_w A_{\varphi}(w)$ (with prob. = 1)

i.e., with prob $\geq \epsilon(n)$, we have $\exists_w \varphi(w) \leftrightarrow \bigoplus_w A_{\varphi}(w)$ (and hence also $\forall_w \text{not } \varphi(w) \leftrightarrow \text{not } \bigoplus_w A_{\varphi}(w)$)
QBF to $\oplus BF$

We have a randomized reduction: $\varphi$ to $A_{\varphi}$ such that

- $\exists_w \varphi(w) \Rightarrow \oplus_w A_{\varphi}(w)$ with prob. $\geq \epsilon(n)$
- $\forall_w \neg \varphi(w) \Rightarrow \neg \oplus_w A_{\varphi}(w)$ (with prob. $= 1$)

i.e., with prob $\geq \epsilon(n)$, we have $\exists_w \varphi(w) \Leftrightarrow \oplus_w A_{\varphi}(w)$ (and hence also $\forall_w \neg \varphi(w) \Leftrightarrow \neg \oplus_w A_{\varphi}(w)$)

Reduction works even if $\varphi(w)$ is a partially quantified Boolean formula
QBF to $\oplus$BF

We have a randomized reduction: $\varphi$ to $A_{\varphi}$ such that

$\exists_w \varphi(w) \Rightarrow \oplus_w A_{\varphi}(w)$ with prob. $\geq \epsilon(n)$

$\forall_w \neg \varphi(w) \Rightarrow \neg \oplus_w A_{\varphi}(w)$ (with prob. $= 1$)

i.e., with prob $\geq \epsilon(n)$, we have $\exists_w \varphi(w) \Leftrightarrow \oplus_w A_{\varphi}(w)$ (and hence also $\forall_w \neg \varphi(w) \Leftrightarrow \neg \oplus_w A_{\varphi}(w)$)

Reduction works even if $\varphi(w)$ is a partially quantified Boolean formula

Can all $\exists/\forall$ be removed, by repeating, so that only $\oplus$ remain?
Some # arithmetic
Some # arithmetic

Given two boolean formulas $\varphi(x)$ and $\psi(y)$, define
Given two boolean formulas $\varphi(x)$ and $\psi(y)$, define $F_{\varphi,\psi}(x,y)$: $\varphi(x)$ and $\psi(y)$
Some # arithmetic

Given two boolean formulas $\varphi(x)$ and $\psi(y)$, define

$F_{\varphi,\psi}(x,y)$: $\varphi(x)$ and $\psi(y)$

$\#F_{\varphi,\psi} = \#\varphi \cdot \#\psi$
Some # arithmetic

Given two boolean formulas \( \varphi(x) \) and \( \psi(y) \), define

- \( F_{\varphi \cdot \psi}(x,y) \): \( \varphi(x) \) and \( \psi(y) \)
- \( \# F_{\varphi \cdot \psi} = \# \varphi \cdot \# \psi \)
- \( F_{\varphi + \psi}(x,y,z) \): (\( z=0, y=0 \) and \( \varphi(x) \)) or (\( z=1, x=0 \) and \( \psi(y) \))
Some # arithmetic

Given two boolean formulas $\varphi(x)$ and $\psi(y)$, define

- $F_{\varphi \cdot \psi}(x,y): \varphi(x)$ and $\psi(y)$
- $\#F_{\varphi \cdot \psi} = \#\varphi \cdot \#\psi$
- $F_{\varphi + \psi}(x,y,z): (z=0, y=0 \text{ and } \varphi(x)) \text{ or } (z=1, x=0 \text{ and } \psi(y))$
- $\#F_{\varphi + \psi} = \#\varphi + \#\psi$
Some \# arithmetic

Given two boolean formulas $\varphi(x)$ and $\psi(y)$, define

- $F_{\varphi,\psi}(x,y): \varphi(x) \text{ and } \psi(y)$
- $\#F_{\varphi,\psi} = \#\varphi \cdot \#\psi$
- $F_{\varphi+\psi}(x,y,z): (z=0, y=0 \text{ and } \varphi(x)) \text{ or } (z=1, x=0 \text{ and } \psi(y))$
- $\#F_{\varphi+\psi} = \#\varphi + \#\psi$
- $F_{\varphi+1} := (z=0 \text{ and } \varphi(x)) \text{ or } (z=1 \text{ and } x=0). \#F_{\varphi+1} = \#\varphi + 1$
Some $\#$ arithmetic

Given two boolean formulas $\varphi(x)$ and $\psi(y)$, define

$F_{\varphi \psi}(x,y)$: $\varphi(x)$ and $\psi(y)$

$\#F_{\varphi \psi} = \#\varphi \cdot \#\psi$

$F_{\varphi + \psi}(x,y,z)$: $(z=0,y=0$ and $\varphi(x))$ or $(z=1,x=0$ and $\psi(y))$

$\#F_{\varphi + \psi} = \#\varphi + \#\psi$

$F_{\varphi + 1} := (z=0$ and $\varphi(x))$ or $(z=1$ and $x=0)$. $\#F_{\varphi + 1} = \#\varphi + 1$

Works even if $\varphi$, $\psi$ are partially quantified boolean formulas
Some $\oplus$ arithmetic
Some $\oplus$ arithmetic

- Boolean combinations of QBFs with $\oplus$ quantifiers
Some $\oplus$ arithmetic

Boolean combinations of QBFs with $\oplus$ quantifiers

$\oplus_x \varphi(x) \text{ and } \oplus_y \psi(y) \iff \oplus_{x,y} F_{\varphi,\psi}(x,y)$, i.e. $\oplus_{x,y} \varphi(x) \text{ and } \psi(y)$
Some $\oplus$ arithmetic

Boolean combinations of QBFs with $\oplus$ quantifiers

1. $\oplus_x \varphi(x)$ and $\oplus_y \psi(y) \iff \oplus_{x,y} F_{\varphi,\psi}(x,y)$, i.e. $\oplus_{x,y} \varphi(x)$ and $\psi(y)$

2. $\text{not } \oplus_x \varphi(x) \iff \oplus_{x,z} F_{\varphi+1}(x,z)$, i.e. $\oplus_{x,z} (z=1, x=0) \text{ or } (z=0, \varphi(x))$
Some $\oplus$ arithmetic

Boolean combinations of QBFs with $\oplus$ quantifiers

- $\oplus_x \varphi(x) \quad \text{and} \quad \oplus_y \psi(y) \iff \oplus_{x,y} F_{\varphi,\psi}(x,y)$, i.e. $\oplus_{x,y} \varphi(x) \quad \text{and} \quad \psi(y)$

- not $\oplus_x \varphi(x) \iff \oplus_{x,z} F_{\varphi,1}(x,z)$, i.e. $\oplus_{x,z} (z=1,x=0) \quad \text{or} \quad (z=0,\varphi(x))$

- $\oplus_x (\oplus_y \varphi(x,y)) \iff \oplus_{x,y} \varphi(x,y)$
Some $\oplus$ arithmetic

- Boolean combinations of QBFs with $\oplus$ quantifiers
  - $\oplus_x \varphi(x)$ and $\oplus_y \psi(y) \iff \oplus_{x,y} F_{\varphi,\psi}(x,y)$, i.e. $\oplus_{x,y} \varphi(x)$ and $\psi(y)$
  - not $\oplus_x \varphi(x) \iff \oplus_{x,z} F_{\varphi+1}(x,z)$. i.e. $\oplus_{x,z} (z=1, x=0)$ or $(z=0, \varphi(x))$
  - $\oplus_x (\oplus_y \varphi(x,y)) \iff \oplus_{x,y} \varphi(x,y)$

- $(\oplus, \exists, \forall)$-QBF can be converted to the form $\oplus_z F(z)$, where $F$ is a $(\exists, \forall)$-QBF, increasing the size by at most a constant factor, and not changing number of $\exists, \forall$
QBF to $\Theta BF$
QBF to $\oplus BF$

- Recall: with prob $\geq \varepsilon(n)$, we have $\exists_w \varphi(w) \Leftrightarrow \bigoplus_w A_{\varphi}(w)$ (and $\forall_w \neg \varphi(w) \Leftrightarrow \neg \bigoplus_w A_{\varphi}(w)$)
QBF to $\oplus$ BF

- Recall: with prob $\geq \epsilon(n)$, we have $\exists_w \varphi(w) \iff \oplus_w A_{\varphi}(w)$ (and $\forall_w \neg \varphi(w) \iff \neg \oplus_w A_{\varphi}(w)$)

- Boosting the probability: $\epsilon(n)$ to $1-\delta(n)$
QBF to $\oplus$ BF

- Recall: with prob $\geq \epsilon(n)$, we have $\exists_w \varphi(w) \iff \oplus_w A_{\varphi}(w)$ (and $\forall_w \neg \varphi(w) \iff \neg \oplus_w A_{\varphi}(w)$)

- Boosting the probability: $\epsilon(n)$ to $1-\delta(n)$

  $\oplus_w A^1_{\varphi}(w)$ or $\oplus_w A^2_{\varphi}(w)$ or ... or $\oplus_w A^+_{\varphi}(w)$
QBF to $\bigoplus\mathrm{BF}$

- Recall: with prob $\geq \varepsilon(n)$, we have $\exists_w \varphi(w) \iff \bigoplus_w A_{\varphi}(w)$ (and $\forall_w \neg \varphi(w) \iff \neg \bigoplus_w A_{\varphi}(w)$)

- Boosting the probability: $\varepsilon(n)$ to $1-\delta(n)$

  - $\bigoplus_w A_{\varphi}^1(w)$ or $\bigoplus_w A_{\varphi}^2(w)$ or ... or $\bigoplus_w A_{\varphi}^t(w)$

  - Can rewrite in the form $\bigoplus_z B_{\varphi}(z)$ where $B_{\varphi}$ has no $\bigoplus$
QBF to \( \oplus \text{BF} \)

Recall: with prob \( \geq \varepsilon(n) \), we have \( \exists_w \varphi(w) \Leftrightarrow \bigoplus_w A_{\varphi}(w) \) (and \( \forall_w \neg \varphi(w) \Leftrightarrow \neg \bigoplus_w A_{\varphi}(w) \))

Boosting the probability: \( \varepsilon(n) \) to \( 1-\delta(n) \)

\( \bigoplus_w A^1_{\varphi}(w) \) or \( \bigoplus_w A^2_{\varphi}(w) \) or ... or \( \bigoplus_w A^t_{\varphi}(w) \)

Can rewrite in the form \( \bigoplus_z B_{\varphi}(z) \) where \( B_{\varphi} \) has no \( \bigoplus \)

In prenex form \( \bigoplus_z B_{\varphi}(z) \) has one less \( \exists/\forall \) than \( \exists_w \varphi(w) \)
QBF to $\oplus$BF

- Recall: with prob $\geq \varepsilon(n)$, we have $\exists_w \varphi(w) \Leftrightarrow \oplus_w A_{\varphi}(w)$ (and $\forall_w \neg \varphi(w) \Leftrightarrow \neg \oplus_w A_{\varphi}(w)$)

- Boosting the probability: $\varepsilon(n)$ to $1-\delta(n)$

  - $\oplus_w A^1_{\varphi}(w)$ or $\oplus_w A^2_{\varphi}(w)$ or ... or $\oplus_w A^t_{\varphi}(w)$

  - Can rewrite in the form $\oplus_z B_{\varphi}(z)$ where $B_{\varphi}$ has no $\oplus$

  - In prenex form $\oplus_z B_{\varphi}(z)$ has one less $\exists/\forall$ than $\exists_w \varphi(w)$

  - If we start from $\oplus_x \exists_w \varphi(w,x)$ we get equivalent (with probability $1-\delta(n)$) $\oplus_x \oplus_z B_{\varphi}(z,x)$
QBF to ⊕BF

- Recall: with prob ≥ ε(n), we have \( \exists_w \varphi(w) \Leftrightarrow \bigoplus_w A_{\varphi}(w) \) (and \( \forall_w \neg \varphi(w) \Leftrightarrow \neg \bigoplus_w A_{\varphi}(w) \))

- Boosting the probability: ε(n) to 1−δ(n)

- \( \bigoplus_w A^1_{\varphi}(w) \) or \( \bigoplus_w A^2_{\varphi}(w) \) or ... or \( \bigoplus_w A^t_{\varphi}(w) \)
  - Can rewrite in the form \( \bigoplus_z B_{\varphi}(z) \) where \( B_{\varphi} \) has no \( \bigoplus \)
  - In prenex form \( \bigoplus_z B_{\varphi}(z) \) has one less \( \exists / \forall \) than \( \exists_w \varphi(w) \)
  - If we start from \( \bigoplus_x \exists_w \varphi(w, x) \) we get equivalent (with probability 1−δ(n)) \( \bigoplus_x \bigoplus_z B_{\varphi}(z, x) \)
  - By repeating, QBF can be converted to the form \( \bigoplus_z F(z) \) where \( F \) is unquantified, equivalent with prob. close to 1
Reducing PH to P^{#P}
Reducing PH to $\mathsf{P}^{\#\mathsf{p}}$

- Two steps
Reducing $\text{PH}$ to $P^{\#P}$

- Two steps
  - Randomized reduction of $\text{PH}$ to $P^{\oplus P}$
Reducing PH to $\texttt{P}^{\#\texttt{P}}$

- Two steps
- Randomized reduction of PH to $\texttt{P}^{\oplus\texttt{P}}$
  - TQBF instance $\psi$ to $\oplus\texttt{SAT}$ instance $\varphi_\psi$
Reducing \( \text{PH} \) to \( \text{P}^{\#\text{P}} \)

- Two steps
  - Randomized reduction of \( \text{PH} \) to \( \text{P}^{\oplus \text{P}} \)
    - TQBF instance \( \psi \) to \( \oplus \text{SAT} \) instance \( \varphi_\psi \)
    - \( \psi \Rightarrow \oplus \varphi_\psi \) w.p. > 2/3; \( \neg \psi \Rightarrow \neg \oplus \varphi_\psi \) (w.p. 1)
Reducing PH to $P^{\#P}$

Two steps

Randomized reduction of PH to $P^{\oplus P}$

- TQBF instance $\psi$ to $\oplus$SAT instance $\varphi_\psi$
- $\psi \Rightarrow \oplus \varphi_\psi$ w.p. $> 2/3$; $\neg \psi \Rightarrow \neg \oplus \varphi_\psi$ (w.p. 1)

Converting the probabilistic guarantee to a deterministic $\#P$ calculation
Reducing PH to $P^{\#P}$

Two steps

- Randomized reduction of PH to $P^{\oplus P}$
  - TQBF instance $\psi$ to $\oplus$SAT instance $\varphi_\psi$
  - $\psi \Rightarrow \ominus \varphi_\psi$ w.p. $> 2/3$; $\neg \psi \Rightarrow \neg \ominus \varphi_\psi$ (w.p. 1)

- Converting the probabilistic guarantee to a deterministic $\#P$ calculation
  - $\psi$ s.t. $\neg \ominus \varphi_\psi \Rightarrow \#\theta_\psi = 0 \pmod N$
Reducing PH to $P^{\#P}$

Two steps

Randomized reduction of PH to $P^{\oplus P}$

- TQBF instance $\psi$ to $\oplus$SAT instance $\varphi_\psi$
- $\psi \Rightarrow \oplus \varphi_\psi$ w.p. > $2/3$; $\neg \psi \Rightarrow \neg \oplus \varphi_\psi$ (w.p. 1)

Converting the probabilistic guarantee to a deterministic $\#P$ calculation

- $\psi$ s.t. $\neg \oplus \varphi_\psi \Rightarrow \#\theta_\psi = 0 \pmod{N}$
- $\psi$ s.t. $\oplus \varphi_\psi$ w.p. > $2/3 \Rightarrow \#\theta_\psi \neq 0 \pmod{N}$
Reduction to $\mathbb{#P}$
Reduction to \#P

- Converting the probabilistic guarantee to a deterministic \#P calculation
Reduction to \#P

- Converting the probabilistic guarantee to a deterministic \#P calculation

\[ \psi \text{ s.t. } \neg \oplus \varphi_\psi \Rightarrow \#\theta_\psi = 0 \pmod{N} \]
Reduction to \#P

- Converting the probabilistic guarantee to a deterministic \#P calculation
  - \( \psi \) s.t. \( \neg \oplus \varphi_\psi \Rightarrow \#\theta_\psi = 0 \pmod{N} \)
  - \( \psi \) s.t. \( \oplus \varphi_\psi \) w.p. > 2/3 \( \Rightarrow \#\theta_\psi \neq 0 \pmod{N} \)
Reduction to \#P

- Converting the probabilistic guarantee to a deterministic \#P calculation
  - \( \psi \text{ s.t. } \neg \oplus \phi_\psi \Rightarrow \#\theta_\psi = 0 \pmod{N} \)

- \( \psi \text{ s.t. } \oplus \phi_\psi \text{ w.p. } > 2/3 \Rightarrow \#\theta_\psi \neq 0 \pmod{N} \)

- Attempt 1: let \( \phi_\psi^r \) be the formula generated using random tape \( r \). To determine if \( \psi \) is such that number of random tapes \( r \) for which \( \oplus \phi_\psi^r \) holds is 0 or \( > (2/3)2^m \)
Reduction to \#P

- Converting the probabilistic guarantee to a deterministic \#P calculation
  - \( \psi \) s.t. \( \neg \bigoplus \varphi_{\psi} \Rightarrow \#\theta_{\psi} = 0 \pmod{N} \)
  - \( \psi \) s.t. \( \bigoplus \varphi_{\psi} \) w.p. > 2/3 \( \Rightarrow \#\theta_{\psi} \neq 0 \pmod{N} \)

- Attempt 1: let \( \varphi_{\psi}^r \) be the formula generated using random tape \( r \). To determine if \( \psi \) is such that number of random tapes \( r \) for which \( \bigoplus \varphi_{\psi}^r \) holds is 0 or > \((2/3)^2 m\)
  - Enough to compute \( \#_r \bigoplus \varphi_{\psi}^r \)
Reduction to #P

- Converting the probabilistic guarantee to a deterministic #P calculation
  - $\psi$ s.t. $\neg \bigoplus \varphi_\psi \Rightarrow \#\theta_\psi = 0 \pmod{N}$
  - $\psi$ s.t. $\bigoplus \varphi_\psi$ w.p. > 2/3 $\Rightarrow \#\theta_\psi \neq 0 \pmod{N}$

- Attempt 1: let $\varphi_{\psi^r}$ be the formula generated using random tape $r$. To determine if $\psi$ is such that number of random tapes $r$ for which $\bigoplus \varphi_{\psi^r}$ holds is 0 or > $(2/3)2^m$
  - Enough to compute $\#_r \bigoplus \varphi_{\psi^r}$
  - But $\bigoplus \varphi_{\psi^r}$ is not in P (though $\varphi_{\psi^r}(x)$ is in P)
Reduction to #P
Reduction to \( \#P \)

Attempt 2: If \( \bigoplus_{x} \varphi_{\psi}^r = \#_{x} \varphi_{\psi}^r \) then enough to compute the number of \((x,r)\) such that \( \varphi_{\psi}^r(x) \)
Reduction to \#P

Attempt 2: If $\bigoplus_x \varphi_\psi^r = \#_x \varphi_\psi^r$ then enough to compute the number of $(x,r)$ such that $\varphi_\psi^r(x)$

But $\bigoplus \varphi_\psi$ is $\#\varphi_\psi \mod 2$
Reduction to $\#P$

- Attempt 2: If $\bigoplus_x \varphi_\psi^r = \#_x \varphi_\psi^r$ then enough to compute the number of $(x,r)$ such that $\varphi_\psi^r(x)$

  - But $\bigoplus \varphi_\psi$ is $\# \varphi_\psi \mod 2$

- Plan: Create $\varphi' = T(\varphi)$, such that
Reduction to $\#P$

Attempt 2: If $\bigoplus_x \varphi_\psi^r = \#_x \varphi_\psi^r$ then enough to compute the number of $(x,r)$ such that $\varphi_\psi^r(x)$

But $\bigoplus \varphi_\psi$ is $\#\varphi_\psi \mod 2$

Plan: Create $\varphi' = T(\varphi)$, such that

$\neg \bigoplus \varphi \Rightarrow \#\varphi' = 0 \mod N$
Reduction to \#P

Attempt 2: If $\bigoplus_x \varphi_\psi^r = \#_x \varphi_\psi^r$ then enough to compute the number of $(x,r)$ such that $\varphi_\psi^r(x)$

But $\bigoplus \varphi_\psi$ is $\#\varphi_\psi$ mod 2

Plan: Create $\varphi' = T(\varphi)$, such that

- $\neg \bigoplus \varphi \Rightarrow \#\varphi' = 0$ mod N
- $\bigoplus \varphi \Rightarrow \#\varphi' = -1$ mod N
Reduction to \#P

- Attempt 2: If \( \bigoplus_x \varphi \psi^r = \#_x \varphi \psi^r \) then enough to compute the number of \((x,r)\) such that \( \varphi \psi^r(x) \)

- But \( \bigoplus \varphi \psi \) is \#\varphi \psi \mod 2

- Plan: Create \( \varphi' = T(\varphi) \), such that
  - \( \neg \bigoplus \varphi \Rightarrow \#\varphi' = 0 \mod N \)
  - \( \bigoplus \varphi \Rightarrow \#\varphi' = -1 \mod N \)

- \( N > 2^m \) so that for \((2/3).2^m < R \leq 2^m\) we have \( R.(-1) \neq 0 \mod N \)
Reduction to \( \#P \)

- **Attempt 2:** If \( \bigoplus_x \varphi_\psi^r = \#_x \varphi_\psi^r \) then enough to compute the number of \((x,r)\) such that \( \varphi_\psi^r(x) \)

- But \( \bigoplus \varphi_\psi \) is \( \#\varphi_\psi \) mod 2

- **Plan:** Create \( \varphi' = T(\varphi) \), such that
  - \( \neg \bigoplus \varphi \Rightarrow \#\varphi' = 0 \) mod \( N \)
  - \( \bigoplus \varphi \Rightarrow \#\varphi' = -1 \) mod \( N \)

- \( N > 2^m \) so that for \( (2/3).2^m < R \leq 2^m \) we have \( R.(-1) \neq 0 \) mod \( N \)

- Let \( \theta_\psi(x,r) = T(\varphi_\psi^r)(x) \). Use \( \#\theta_\psi \) mod \( N \) to check if w.h.p. \( \bigoplus \varphi \)
Reduction to #P
Reduction to \#P

Remains to do: Given \( \varphi \), create \( \varphi' \) such that for \( N=2^{2^k} \), where \( k = O(\log m) \)
Reduction to \( \#P \)

- Remains to do: Given \( \varphi \), create \( \varphi' \) such that for \( N = 2^{2^k} \), where \( k = O(\log m) \)

- \( \neg \oplus \varphi \Rightarrow \#\varphi' = 0 \mod N \)
Reduction to \#P

Remains to do: Given \( \varphi \), create \( \varphi' \) such that for \( N=2^{2^k} \), where \( k = O(\log m) \)

\[ \neg \oplus \varphi \Rightarrow \#\varphi' = 0 \mod N \]

\[ \oplus \varphi \Rightarrow \#\varphi' = -1 \mod N \]
Reduction to \#P

Remains to do: Given $\varphi$, create $\varphi'$ such that for $N=2^{2^k}$, where $k = O(\log m)$

- $-\top \varphi \Rightarrow \#\varphi' = 0 \mod N$
- $\top \varphi \Rightarrow \#\varphi' = -1 \mod N$

Initially true for $N = 2$ ($2^{2^i}, i=0$)
Reduction to #$P$

- Remains to do: Given $\phi$, create $\phi'$ such that for $N=2^{2^k}$, where $k = O(\log m)$

- $\neg \oplus \phi \Rightarrow \#\phi' = 0 \mod N$

- $\oplus \phi \Rightarrow \#\phi' = -1 \mod N$

- Initially true for $N = 2 \left(2^{2^i}, i=0\right)$

- $\phi_{i+1} = F_4(\phi_i)^3 + 3(\phi_i)^4$ so that $\#\phi_{i+1} = 4(\#\phi_i)^3 + 3(\#\phi_i)^4$
Reduction to \#P

- Remains to do: Given $\varphi$, create $\varphi'$ such that for $N=2^{2^k}$, where $k = O(\log m)$
  
  - $\varphi \oplus \oplus \Rightarrow \#\varphi' = 0 \mod N$
  - $\oplus \varphi \Rightarrow \#\varphi' = -1 \mod N$

- Initially true for $N = 2 (2^{2^i}, i=0)$

  - $\varphi_{i+1} = F_4(\varphi_i)^3 + 3(\varphi_i)^4$ so that $\#\varphi_{i+1} = 4(\#\varphi_i)^3 + 3(\#\varphi_i)^4$

  - $\#\varphi_i = -1 \mod 2^{2^i}$ implies $\varphi_{i+1} = -1 \mod 2^{2^{i+1}}$ (for $i \geq 0$)
Reduction to \#P

Remains to do: Given \( \varphi \), create \( \varphi' \) such that for \( N=2^{2^k} \), where \( k = O(\log m) \)

\[ \neg \bigoplus \varphi \Rightarrow \#\varphi' = 0 \mod N \]

\[ \bigoplus \varphi \Rightarrow \#\varphi' = -1 \mod N \]

Initially true for \( N = 2 \ (2^{2^i}, i=0) \)

\[ \varphi_{i+1} = F_4(\varphi_i)^3 + 3(\varphi_i)^4 \] so that \( \#\varphi_{i+1} = 4(\#\varphi_i)^3 + 3(\#\varphi_i)^4 \)

\[ \#\varphi_i = -1 \mod 2^{2^i} \] implies \( \varphi_{i+1} = -1 \mod 2^{2^{i+1}} \) (for \( i \geq 0 \))

Clearly \( \#\varphi_i = 0 \mod 2^{2^i} \) implies \( \varphi_{i+1} = 0 \mod 2^{2^{i+1}} \)
\[ \text{PH} \subseteq \text{P}^{\#P} \]
PH \subseteq P^{\#P}

Summary:
\[ \mathsf{PH} \subseteq \mathsf{P}^{\# \mathsf{P}} \]

Summary:

First, randomized reduction of \( \mathsf{PH} \) to \( \mathsf{P}^{\oplus \mathsf{P}} \).
\[ \text{PH} \subseteq \text{P}^{\#P} \]

Summary:

- First, randomized reduction of PH to \( \text{P}^{\oplus\text{P}} \)
- TQBF instance \( \psi \) to \( \oplus\text{SAT} \) instance \( \varphi_\psi \)
**Summary:**

First, randomized reduction of PH to $P^{\#P}$

TQBF instance $\psi$ to $\oplus SAT$ instance $\varphi_{\psi}$

$\psi \Rightarrow \oplus \varphi_{\psi}$ w.p. $> 2/3$; $\neg \psi \Rightarrow \neg \oplus \varphi_{\psi}$ (w.p. 1)
\[ \text{PH} \subseteq \text{P}^{\#P} \]

**Summary:**

- First, randomized reduction of PH to P^{\text{⊕P}}
- TQBF instance \( \psi \) to \( \oplus \text{SAT} \) instance \( \varphi_\psi \)
- \( \psi \Rightarrow \oplus \varphi_\psi \) w.p. > 2/3; \( \neg \psi \Rightarrow \neg \oplus \varphi_\psi \) (w.p. 1)
- Converting the probabilistic guarantee to a deterministic \#P calculation


\[
\mathbf{PH} \subseteq \mathbf{P}^{\#P}
\]

**Summary:**

- First, randomized reduction of PH to \(\mathbf{P}^{\oplus P}\)
- TQBF instance \(\psi\) to \(\oplus\mathbf{SAT}\) instance \(\varphi_\psi\)
- \(\psi \Rightarrow \oplus \varphi_\psi\) w.p. > 2/3; \(\neg \psi \Rightarrow \neg \oplus \varphi_\psi\) (w.p. 1)
- Converting the probabilistic guarantee to a deterministic \#P calculation
  - \(\psi\) s.t. \(\neg \oplus \varphi_\psi\) \(\Rightarrow \#\theta_\psi = 0\) (mod N)
Summary:

- First, randomized reduction of $PH$ to $P^{\oplus P}$
  - TQBF instance $\psi$ to $\oplus SAT$ instance $\varphi_\psi$
  - $\psi \Rightarrow \oplus \varphi_\psi$ w.p. > $2/3$; $\neg \psi \Rightarrow \neg \oplus \varphi_\psi$ (w.p. 1)

- Converting the probabilistic guarantee to a deterministic $\#P$ calculation
  - $\psi$ s.t. $\neg \oplus \varphi_\psi \Rightarrow \#\theta_\psi = 0$ (mod $N$)
  - $\psi$ s.t. $\oplus \varphi_\psi$ w.p. > $2/3$ $\Rightarrow$ $\#\theta_\psi \neq 0$ (mod $N$)
Approximation for \#P
Approximation for $\#P$

$\alpha$-approximation of $f$: estimate $f(x)$ within a factor $\alpha$
Approximation for \#P

- $\alpha$-approximation of $f$: estimate $f(x)$ within a factor $\alpha$

- Randomized approximation ("PAC"): answer is within a factor $\alpha$ with probability at least $1-\delta$
Approximation for \#P

- \(\alpha\)-approximation of \(f\): estimate \(f(x)\) within a factor \(\alpha\)

- Randomized approximation ("PAC"): answer is within a factor \(\alpha\) with probability at least \(1-\delta\)

- \#CYCLE is hard to even approximate unless \(P=NP\)
Approximation for #P

- $\alpha$-approximation of $f$: estimate $f(x)$ within a factor $\alpha$

- Randomized approximation ("PAC"): answer is within a factor $\alpha$ with probability at least $1-\delta$

- #CYCLE is hard to even approximate unless $P=NP$

- If $P=NP$, every problem in #P can be "well approximated"
Approximation for \#P

- \(\alpha\)-approximation of \(f\): estimate \(f(x)\) within a factor \(\alpha\)

- Randomized approximation ("PAC"): answer is within a factor \(\alpha\) with probability at least \(1-\delta\)

- \#CYCLE is hard to even approximate unless \(P=NP\)
  
  - If \(P=NP\), every problem in \#P can be "well approximated"

- Permanent has an FPRAS
Approximation for \#P

- \(\alpha\)-approximation of \(f\): estimate \(f(x)\) within a factor \(\alpha\)

- Randomized approximation ("PAC"): answer is within a factor \(\alpha\) with probability at least \(1-\delta\)

- \#CYCLE is hard to even approximate unless \(P=NP\)

  - If \(P=NP\), every problem in \#P can be "well approximated"

- Permanent has an FPRAS

  - For any \(\varepsilon, \delta > 0\), \(\alpha\)-approximation for \(\alpha = 1-\varepsilon\) in time \(\text{poly}(n, \log 1/\varepsilon, \log 1/\delta)\)
Approximation for \#P

- \(\alpha\)-approximation of \(f\): estimate \(f(x)\) within a factor \(\alpha\)
- Randomized approximation ("PAC"): answer is within a factor \(\alpha\) with probability at least 1\(-\delta\)
- \#CYCLE is hard to even approximate unless \(P=NP\)
  - If \(P=NP\), every problem in \#P can be “well approximated”
- Permanent has an FPRAS
  - For any \(\epsilon, \delta > 0\), \(\alpha\)-approximation for \(\alpha = 1-\epsilon\) in time \(\text{poly}(n, \log 1/\epsilon, \log 1/\delta)\)
  - Technique: Monte Carlo Markov Chain (MCMC)
Approximation for #P

- α-approximation of f: estimate $f(x)$ within a factor $\alpha$
- Randomized approximation (“PAC”): answer is within a factor $\alpha$ with probability at least $1-\delta$
- #CYCLE is hard to even approximate unless $P=NP$
  - If $P=NP$, every problem in #P can be “well approximated”
- Permanent has an FPRAS
  - For any $\epsilon, \delta > 0$, α-approximation for $\alpha = 1-\epsilon$ in time $\text{poly}(n, \log 1/\epsilon, \log 1/\delta)$
- Technique: Monte Carlo Markov Chain (MCMC)
  - Very useful for sampling. Turns out counting $\approx$ sampling!
Approximation for \#P

- \(\alpha\)-approximation of \(f\): estimate \(f(x)\) within a factor \(\alpha\)

- Randomized approximation ("PAC"): answer is within a factor \(\alpha\) with probability at least \(1 - \delta\)

- \#CYCLE is hard to even approximate unless \(P=NP\)

- If \(P=NP\), every problem in \#P can be "well approximated"

- Permanent has an FPRAS

- For any \(\epsilon, \delta > 0\), \(\alpha\)-approximation for \(\alpha = 1 - \epsilon\) in time \(\text{poly}(n, \log 1/\epsilon, \log 1/\delta)\)

- Technique: Monte Carlo Markov Chain (MCMC)
  - Very useful for sampling. Turns out counting \(\approx\) sampling!