Computational Complexity Theory

Lecture 24: GNI is in BP.NP

Department of Computer Science,
Indian Institute of Science

Recap: Randomized reduction

- Definition. We say a L_1 reduces to a L_2 in *randomized polynomial-time*, denoted $L_1 \leq_r L_2$, if there's a poly-time PTM M s.t. for every $x \in \{0,1\}^*$

$$\Pr[L_1(x) = L_2(M(x))] \geq 2/3.$$

- For arbitrary L_1 and L_2, we may not be able to boost the success probability $2/3$, and so, the above kind of reductions **needn't be transitive**. However,
- Obs. If $L_1 \leq_r L_2$ and $L_2 \in$ BPP, then $L_1 \in$ BPP.

(Easy homework)

Recap: Randomized reduction

- Definition. We say a L_1 reduces to a L_2 in *randomized polynomial-time*, denoted $L_1 \leq_r L_2$, if there's a poly-time PTM M s.t. for every $x \in \{0,1\}^*$

$$\Pr[L_1(x) = L_2(M(x))] \geq 2/3.$$

- Obs. If L_2 = SAT, then we can boost the success probability from $\frac{1}{2} + |x|^{-c}$ to $1 - \exp(-|x|^d)$.
- *Proof idea.* BPP error reduction trick + Cook-Levin.

(homework)

Recap: Randomized reduction

- Definition. We say a L_1 reduces to a L_2 in <u>*randomized polynomial-time*</u>, denoted $L_1 \leq_r L_2$, if there's a poly-time PTM M s.t. for every $x \in \{0,1\}^*$

$$\Pr[L_1(x) = L_2(M(x))] \geq 2/3.$$

- Obs. If L_2 = SAT, then we can boost the success probability from $½ + |x|^{-c}$ to $1 - \exp(-|x|^d)$.

- Recall, NP = $\{L : L \leq_p \text{SAT}\}$. It makes sense to define a similar class using randomized poly-time reduction.

Recap: Class BP.NP

- Definition. We say a L_1 reduces to a L_2 in *randomized polynomial-time*, denoted $L_1 \leq_r L_2$, if there's a poly-time PTM M s.t. for every $x \in \{0,1\}^*$

$$\Pr[L_1(x) = L_2(M(x))] \geq 2/3.$$

- Obs. If L_2 = SAT, then we can boost the success probability from $\frac{1}{2} + |x|^{-c}$ to $1 - \exp(-|x|^d)$.
- Definition. $\mathrm{BP.NP} = \{L : L \leq_r \mathrm{SAT}\}$.
- Class BP.NP is also known as AM (*Arthur-Merlin protocol*) in the literature.

Recap: Class BP.NP

- Definition. BP.NP = $\{L : L \leq_r \text{SAT}\}$.
- Observe that NP $\subseteq$ BP.NP and BPP $\subseteq$ BP.NP. Is BP.NP = NP ? Many believe that the answer is "yes".

- Theorem. If certain reasonable circuit lower bounds hold, then BP.NP = NP.
- *Proof idea*. Similar to Nisan & Wigderson's conditional BPP = P result.

Recap: Class BP.NP

- Definition. $\text{BP.NP} = \{L : L \leq_r \text{SAT}\}$.
- Observe that $\text{NP} \subseteq \text{BP.NP}$ and $\text{BPP} \subseteq \text{BP.NP}$. Is $\text{BP.NP} = \text{NP}$? Many believe that the answer is "yes".

- We may further ask:

1. Is BP.NP in PH? Recall, BPP is in PH.

2. Is $\overline{\text{SAT}} \in \text{BP.NP}$? Recall, if $\text{SAT} \in \text{BPP}$ then PH collapses. ($\text{SAT} \in \text{BP.NP}$ as $\text{NP} \subseteq \text{BP.NP}$.)

Recap: Class BP.NP

- Definition. BP.NP = $\{L : L \leq_r \text{SAT}\}$.

- Theorem. BP.NP is in Σ_3. (In fact, BP.NP is in Π_2.)
- *Proof idea.* Similar to the Sipser-Gacs-Lautemann theorem. *(Assignment problem)*

- Wondering if BP.NP $\subseteq \Pi_2$ implies BP.NP $\subseteq \Sigma_2$? Is BP.NP = co-BP.NP ? (Recall, BPP = co-BPP).
- If BP.NP = co-BP.NP then co-NP $\subseteq$ BP.NP. The next theorem shows that this would lead to PH collapse.

Recap: Class BP.NP

- Definition. $\mathrm{BP.NP} = \{L : L \leq_r \mathrm{SAT}\}$.

- Theorem. If $\overline{\mathrm{SAT}} \in \mathrm{BP.NP}$ then $\mathrm{PH} = \Sigma_3$ (in fact, $\mathrm{PH} = \Sigma_2$).
- *Proof idea.* Similar to Adleman's theorem + Karp-Lipton theorem. *(Assignment problem)*

Recap: Class BP.NP

- Definition. BP.NP = $\{L : L \leq_r SAT\}$.

- Theorem. If $\overline{SAT} \in$ BP.NP then PH = Σ_2.

- We would use the above theorem to show that if GI is NP-complete then PH collapses.
- Thus, even without designing an efficient algorithm for GI, we know GI is unlikely to be NP-complete!

Recap: Class BP.NP

- Definition. BP.NP = $\{L : L \leq_r \text{SAT}\}$.
- Theorem. If $\overline{\text{SAT}} \in$ BP.NP then PH = Σ_2.
- We would use the above theorem to show that if GI is NP-complete then PH collapses.
- Theorem. *(Goldwasser-Sipser '87, Boppana, Hastad, Zachos '87)* GNI $\in$ BP.NP.
- *Proof.* We'll prove it today.

Recap: Class BP.NP

- **Definition.** BP.NP = $\{L : L \leq_r SAT\}$.

- **Theorem.** If $\overline{SAT} \in$ BP.NP then PH = Σ_2.

- We would use the above theorem to show that if GI is NP-complete then PH collapses.
- **Theorem.** *(Goldwasser-Sipser '87, Boppana, Hastad, Zachos '87)* GNI $\in$ BP.NP.
- If GI is NP-complete then GNI is co-NP-complete. If so, then the above two theorems imply PH = Σ_2.

Recap: GI in Quasi-P

- Theorem. *(Babai 2015)* There's a deterministic $\exp(O((\log n)^3))$ time algorithm to solve the graph isomorphism problem.

GNI is in BP.NP

Graph Non-isomorphism

- Definition. Let G_1 and G_2 be two undirected graphs on n vertices. Identify the vertices with $[n]$. We say G_1 is *isomorphic* to G_2, denoted $G_1 \cong G_2$, if there's a bijection/permutation $\pi:[n] \rightarrow [n]$ s.t. for all $u, v \in [n]$, (u,v) is an edge in G_1 if and only if $(\pi(u),\pi(v))$ is an edge in G_2.

- Definition. $GNI = \{(G_1, G_2) : G_1 \ncong G_2\}$.

- Clearly, $GNI \in$ co-NP, it is not known if $GNI \in NP$.

GNI is in BP.NP

- The idea.

1. **Step 1**: Let $x = (G_1, G_2)$. Associate a set S_x with (G_1, G_2) s.t. $|S_x|$ is "large" ($2n!$) if $G_1 \ncong G_2$, and $|S_x|$ is "small" ($n!$) if $G_1 \cong G_2$. Elements of S_x can be represented using $m = n^{O(1)}$ bits. *Furthermore, membership in S_x can be <u>certified</u> in* $m^{O(1)} = n^{O(1)}$ time.

GNI is in BP.NP

- The idea.

1. **Step 1**: Let $x = (G_1, G_2)$. Associate a set S_x with (G_1, G_2) s.t. $|S_x|$ is "large" ($2n!$) if $G_1 \ncong G_2$, and $|S_x|$ is "small" ($n!$) if $G_1 \cong G_2$. Elements of S_x can be represented using $m = n^{O(1)}$ bits. *Furthermore, membership in S_x can be* *<u>certified</u>* *in $m^{O(1)} = n^{O(1)}$* time.

There's a poly-time TM V and a polynomial function $q(.)$ s.t.

$u \in S_x \Rightarrow \exists c \in \{0,1\}^{q(|x|)}\ \ V(x, u, c) = 1$

$u \notin S_x \Rightarrow \forall c \in \{0,1\}^{q(|x|)}\ \ V(x, u, c) = 0.$

GNI is in BP.NP

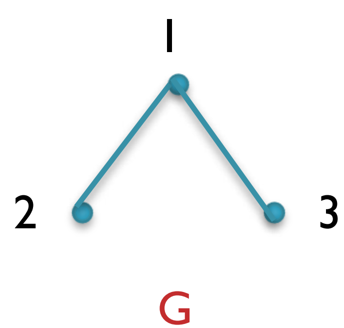
- The idea.

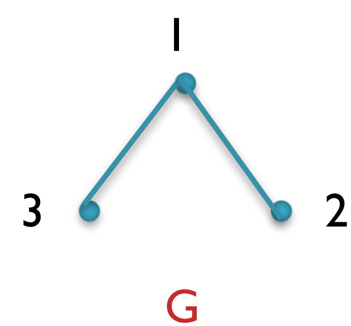
1. **Step 1**: Let $x = (G_1, G_2)$. Associate a set S_x with (G_1, G_2) s.t. $|S_x|$ is "large" ($2n!$) if $G_1 \ncong G_2$, and $|S_x|$ is "small" ($n!$) if $G_1 \cong G_2$. Elements of S_x can be represented using $m = n^{O(1)}$ bits. *Furthermore, membership in S_x can be <u>certified</u> in $m^{O(1)} = n^{O(1)}$* time.

2. **Step 2:** Devise a *<u>randomized</u>* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".

GNI is in BP.NP

- **Step 1**: Let $x = (G_1, G_2)$. Associate a set S_x with (G_1, G_2) s.t. $|S_x|$ is "large" ($2n!$) if $G_1 \ncong G_2$, and $|S_x|$ is "small" ($n!$) if $G_1 \cong G_2$. Elements of S_x can be represented using $m = n^{O(1)}$ bits. *Furthermore, membership in S_x can be <u>certified</u> in $m^{O(1)} = n^{O(1)}$ time.*
- Defn. $Aut(G) = \{$bijection $\pi:[n] \rightarrow [n] : \pi(G) = G\}$.





Permutation $\pi = (1,3,2)$ is in $Aut(G)$.

GNI is in BP.NP

- **Step 1**: Let $x = (G_1, G_2)$. Associate a set S_x with (G_1, G_2) s.t. $|S_x|$ is "large" ($2n!$) if $G_1 \ncong G_2$, and $|S_x|$ is "small" ($n!$) if $G_1 \cong G_2$. Elements of S_x can be represented using $m = n^{O(1)}$ bits. *Furthermore, membership in S_x can be <u>certified</u> in $m^{O(1)} = n^{O(1)}$* time.
- Defn. $Aut(G) = \{$bijection $\pi:[n] \rightarrow [n] : \pi(G) = G\}$.
- Let $S_x = \{(H, \pi): H \cong G_1$ or $H \cong G_2$ and $\pi \in Aut(H)\}$.
- Obs. S_x satisfies the properties stated in Step 1.

(Homework)

GNI is in BP.NP

- **Step 2:** Devise a *randomized* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".

GNI is in BP.NP

- **Step 2:** Devise a *randomized* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".
- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

$|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

$|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

$r \in \{0,1\}^{q(|x|)}$ $y \in \{0,1\}^{q(|x|)}$

GNI is in BP.NP

- **Step 2:** Devise a *<u>randomized</u>* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".
- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) $\Rightarrow$ $\Pr_r[\exists y$ s.t. $M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) $\Rightarrow$ $\Pr_r[\forall y$ s.t. $M(x, y, r) = 0] \geq 2/3$.
- *Proof.* Uses ***Goldwasser-Sipser set lower bound protocol***. We'll see the proof in a while.

GNI is in BP.NP

- **Step 2:** Devise a *randomized* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".
- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

$|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

$|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

We can think of M's computation as a Boolean circuit $\psi_{x,r}(y)$, which can be computed in randomized $|x|^{O(1)}$ time by fixing x and picking $r \in \{0,1\}^{q(n)}$ randomly. *Cook-Levin*

GNI is in BP.NP

- **Step 2:** Devise a *randomized* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".
- Corollary. There's randomized poly-time reduction that maps x to a Boolean circuit $\psi_{x,r}$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\psi_{x,r}(y)$ is satisfiable$] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\psi_{x,r}(y)$ is unsatisfiable$] \geq 2/3$.

GNI is in BP.NP

- **Step 2:** Devise a *randomized* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".
- Corollary. There's randomized poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\phi_{x,r}(z) \text{ is satisfiable}] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\phi_{x,r}(z) \text{ is unsatisfiable}] \geq 2/3$.

$\phi_{x,r}$ is a CNF and $z = y$ + auxiliary variables.
Cook-Levin

GNI is in BP.NP

- **Step 2:** Devise a *randomized* poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t. over the randomness of r, $\phi_{x,r}$ is satisfiable w.h.p if S_x is "large" and unsatisfiable w.h.p if S_x is "small".
- Corollary. There's randomized poly-time reduction that maps x to a CNF $\phi_{x,r}$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\phi_{x,r}(z) \text{ is satisfiable}] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\phi_{x,r}(z) \text{ is unsatisfiable}] \geq 2/3$.

- Hence, GNI is in BP.NP. It remains to prove Lemma *.

■

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

  $r \in \{0,1\}^{q(|x|)}$ $\quad$ $y \in \{0,1\}^{q(|x|)}$

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" family of hash functions that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.

The value of k will be fixed in the analysis.

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $Pr_r [\exists y$ s.t. $M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $Pr_r [\forall y$ s.t. $M(x, y, r) = 0] \geq 2/3$.

- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" family of hash functions that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.

- Let $t = n^{O(1)}$ be sufficiently large. M interprets r as $(i_1, i_2, \ldots, i_t)$, where $i_1, \ldots, i_t$ are indices of hash functions in H.

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" family of hash functions that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.
- Let $t = n^{O(1)}$ be sufficiently large. M interprets r as $(i_1, i_2, \ldots, i_t)$, where $i_1, \ldots, i_t$ are indices of hash functions in H.

$$|r| = n^{O(1)}.$$

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $Pr_r [\exists y$ s.t. $M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $Pr_r [\forall y$ s.t. $M(x, y, r) = 0] \geq 2/3$.

- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" family of hash functions that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.

- M interprets y as $((u_1,c_1), (u_2,c_2),\ldots, (u_t,c_t))$, where $u_1,\ldots, u_t$ are m-bit strings, and c_p is an alleged certificate of u_p's membership in S_x for every $p \in [t]$.

  $|y| = n^{O(1)}$.

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.
  $|S_x| = 2n!$ (large) ➡ $Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$
  $|S_x| = n!$ (small) ➡ $Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.
- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" <u>family of hash functions</u> that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.
- For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

Recall, membership in S_x can be efficiently certified.

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" <u>family of hash functions</u> that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.
- For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$. If sufficiently many (say, t^*) of these checks pass, M outputs 1, else it o/ps 0.

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y$ s.t. $M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y$ s.t. $M(x, y, r) = 0] \geq 2/3$.

- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" <u>family of hash functions</u> that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.
- For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$. If sufficiently many (say, t^*) of these checks pass, M outputs 1, else it o/ps 0. <u>Intuitively, $\exists y$ s.t. t^* of the checks pass iff S_x is large.</u>

Set lower bound protocol

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.
  ??
- *Proof idea.* Let $H = \{h_i\}$ be a "suitable" family of hash functions that map m-bit strings to k-bit strings for an appropriate k. Recall, m = size of an element in S_x.
- For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$. If sufficiently many (say, t^*) of these checks pass, M outputs 1, else it o/ps 0. Intuitively, $\exists y$ s.t. t^* of the checks pass iff S_x is large.

Pairwise independent hash functions

- Definition. A family $H_{m,k}$ of (hash) functions from $\{0,1\}^m$ to $\{0,1\}^k$ is *pairwise independent* if for every distinct $x, x' \in \{0,1\}^m$ and for every $y, y' \in \{0,1\}^k$,

$$\Pr_{h \in_r H_{m,k}} [h(x) = y \text{ and } h(x') = y'] = 2^{-2k}.$$

Pairwise independent hash functions

- Definition. A family $H_{m,k}$ of (hash) functions from $\{0,1\}^m$ to $\{0,1\}^k$ is *pairwise independent* if for every distinct $x, x' \in \{0,1\}^m$ and for every $y, y' \in \{0,1\}^k$,

$$\Pr_{h \in_r H_{m,k}} [h(x) = y \text{ and } h(x') = y'] = 2^{-2k}.$$

- Obs. Let $H_{m,k}$ be a pairwise independent hash function family. For every $x \in \{0,1\}^m$ and $y \in \{0,1\}^k$,

$$\Pr_{h \in_r H_{m,k}} [h(x) = y] = 2^{-k}.$$

Pairwise independent hash functions

- Definition. A family $H_{m,k}$ of (hash) functions from $\{0,1\}^m$ to $\{0,1\}^k$ is *pairwise independent* if for every <u>distinct</u> $x, x' \in \{0,1\}^m$ and for every $y, y' \in \{0,1\}^k$,

$$\Pr_{h \in_r H_{m,k}} [h(x) = y \text{ and } h(x') = y'] = 2^{-2k}.$$

$$= \Pr_{h \in_r H_{m,k}} [h(x) = y] \,.\, \Pr_{h \in_r H_{m,k}} [h(x') = y'] \,.$$

Pairwise independent hash functions

- Definition. A family $H_{m,k}$ of (hash) functions from $\{0,1\}^m$ to $\{0,1\}^k$ is *pairwise independent* if for every distinct $x, x' \in \{0,1\}^m$ and for every $y, y' \in \{0,1\}^k$,

$$\Pr_{h \in_r H_{m,k}} [h(x) = y \text{ and } h(x') = y'] = 2^{-2k}.$$

$$= \Pr_{h \in_r H_{m,k}} [h(x) = y] \,.\, \Pr_{h \in_r H_{m,k}} [h(x') = y'] \,.$$

- Example. Let $\ell > 0$ and F be the *finite field* of size 2^ℓ. We can identify F with $\{0,1\}^\ell$ as elements of F are ℓ-bit strings. For $a, b \in F$, define the function $h_{a,b}$ as $h_{a,b}(x) = ax + b$ for every $x \in F$. Then, $H_{\ell,\ell} = \{h_{a,b} : a,b \in F\}$ is a pairwise independent hash family.

Pairwise independent hash functions

- Example. Let $\ell > 0$ and F be the field *finite field* of size 2^{ℓ}. We can identify F with $\{0,1\}^{\ell}$ as elements of F are ℓ-bit strings. For $a, b \in F$, define the function $h_{a,b}$ as $h_{a,b}(x) = ax + b$ for every $x \in F$. Then, $H_{\ell,\ell} = \{h_{a,b} : a,b \in F\}$ is a pairwise independent hash family.
- *Proof*. Let $x, x' \in F$ be distinct and $y, y' \in F$. Then, $h_{a,b}(x) = y$ & $h_{a,b}(x') = y'$ if and only if $a = (y-y')/(x-x')$ and $b = (xy' - x'y)/(x-x')$.

Pairwise independent hash functions

- Example. Let $\ell > 0$ and F be the *finite field* of size 2^{ℓ}. We can identify F with $\{0,1\}^{\ell}$ as elements of F are ℓ-bit strings. For $a, b \in F$, define the function $h_{a,b}$ as $h_{a,b}(x) = ax + b$ for every $x \in F$. Then, $H_{\ell,\ell} = \{h_{a,b} : a,b \in F\}$ is a pairwise independent hash family.
- *Proof.* Let $x, x' \in F$ be distinct and $y, y' \in F$. Then, $h_{a,b}(x) = y$ & $h_{a,b}(x') = y'$ if and only if $a = (y-y')/(x-x')$ and $b = (xy' - x'y)/(x-x')$. Therefore,

$$\Pr_{a,b \in_r F} [h_{a,b}(x) = y \ \& \ h_{a,b}(x') = y']$$

$$= \Pr_{a,b \in_r F} [a = (y-y')/(x-x') \ \& \ b = (xy' - x'y)/(x-x')]$$

$$= 2^{-2\ell}$$ (as a and b are independently chosen).

∎

Pairwise independent hash functions

- Example. Let $\ell > 0$ and F be the *finite field* of size 2^ℓ. We can identify F with $\{0,1\}^\ell$ as elements of F are ℓ-bit strings. For $a, b \in F$, define the function $h_{a,b}$ as $h_{a,b}(x) = ax + b$ for every $x \in F$. Then, $H_{\ell,\ell} = \{h_{a,b} : a,b \in F\}$ is a pairwise independent hash family.

- Obs. If $m \geq k$, then we can construct a pairwise independent $H_{m,k}$ by considering $H_{m,m}$ as above. Truncate the output of a function to the first k bits.

(Homework)

Pairwise independent hash functions

- Example. Let $\ell > 0$ and F be the *finite field* of size 2^ℓ. We can identify F with $\{0,1\}^\ell$ as elements of F are ℓ-bit strings. For $a, b \in F$, define the function $h_{a,b}$ as $h_{a,b}(x) = ax + b$ for every $x \in F$. Then, $H_{\ell,\ell} = \{h_{a,b} : a,b \in F\}$ is a pairwise independent hash family.

- Obs. If $m \leq k$, then we can construct a pairwise independent $H_{m,k}$ by considering $H_{k,k}$ as above. Generate k-bit i/p for a function by padding with 0.

(Homework)

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* Let $H_{m,k}$ be a family of pairwise independent hash functions.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $Pr_r\,[\exists y$ s.t. $M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $Pr_r\,[\forall y$ s.t. $M(x, y, r) = 0] \geq 2/3$.

- *Proof.* Let $H_{m,k}$ be a family of pairwise independent hash functions. Recall, $r = (i_1, i_2, \ldots, i_t)$, where $i_1, \ldots, i_t$ are indices of functions in $H_{m,k}$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* Let $H_{m,k}$ be a family of pairwise independent hash functions. Recall, $r = (i_1, i_2, \ldots, i_t)$, where $i_1, \ldots, i_t$ are indices of functions in $H_{m,k}$. Also, $y = ((u_1, c_1), (u_2, c_2), \ldots, (u_t, c_t))$, where $u_1, \ldots, u_t \in \{0,1\}^m$, and c_p is an alleged certificate of u_p's membership in S_x for every $p \in [t]$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- For a fixed p, what is the probability (over the randomness of i_p) there's a $u_p \in S_x$ s.t. $h_{i_p}(u_p)=0^k$? We'll upper & lower bound this probability.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.
  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$
  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.
- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- Simplifying notations. As p is fixed, let $h_{i_p} = h$ and $u_p = u$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- Upper bound. $\Pr_h [\exists u \in S_x \text{ s.t. } h(u) = 0^k] \leq |S_x|/2^k$.
- As $H_{m,k}$ is pairwise independent, for every $u \in \{0,1\}^m$, $\Pr_h [h(u) = 0^k] = 2^{-k}$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y$ s.t. $M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y$ s.t. $M(x, y, r) = 0] \geq 2/3$.

- *Proof*. For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- Lower bound.

$$\Pr_h[\exists u \in S_x \text{ s.t. } h(u) = 0^k]$$

$$\geq \sum_{u \in S_x} \Pr_h[h(u) = 0^k] - \sum_{\substack{u,u' \in S_x \\ u \neq u'}} \Pr_h[h(u) = 0^k \,\&\, h(u') = 0^k]$$

(by inclusion-exclusion principle)

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- Lower bound.

$$\Pr_h[\exists u \in S_x \text{ s.t. } h(u) = 0^k]$$

$$\geq |S_x|/2^k - |S_x|^2 / 2^{2k+1}.$$

(as $H_{m,k}$ is pairwise independent)

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $Pr_r\ [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $Pr_r\ [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- Lower bound.

  $$Pr_h\ [\exists u \in S_x \text{ s.t. } h(u) = 0^k]$$

  $$\geq \ |S_x|/2^k \cdot (1 - |S_x|/2^{k+1}).$$

  (as $H_{m,k}$ is pairwise independent)

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ Pr_r [$\exists y$ s.t. $M(x, y, r) = 1$] $\geq 2/3$

  $|S_x| = n!$ (small) ➡ Pr_r [$\forall y$ s.t. $M(x, y, r) = 0$] $\geq 2/3$.
- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- If $|S_x| = n!$ then (by the upper bound)

  Pr_h [$\exists u \in S_x$ s.t. $h(u) = 0^k$] $\leq n!/2^k$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.
  $|S_x| = 2n!$ (large) ➡ $Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$
  $|S_x| = n!$ (small) ➡ $Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.
- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- If $|S_x| = n!$ then (by the upper bound)
  $Pr_h [\exists u \in S_x \text{ s.t. } h(u) = 0^k] \leq n!/2^k$. Hence,
- $Exp_r [\ |\{p\in[t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}|\ ] \leq t.\ n!/2^k$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- Choosing k. Fix k s.t. $2^{k-2} < 2n! \leq 2^{k-1}$.

- If $|S_x| = 2n!$ then (by the lower bound)

  $\Pr_h[\exists u \in S_x \text{ s.t. } h(u) = 0^k] \geq |S_x|/2^k . (1 - |S_x|/2^{k+1})$

  $\geq |S_x|/2^k . ¾ = 3/2. n!/2^k$

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3.$

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.
- Choosing k. Fix k s.t. $2^{k-2} < 2n! \leq 2^{k-1}$.
- If $|S_x| = 2n!$ then (by the lower bound)

  $Pr_h [\exists u \in S_x \text{ s.t. } h(u) = 0^k] \geq 3/2 \,.\, n!/2^k$. Hence,

- $Exp_r [\,|\{p\in[t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}|\,] \geq 3/2 \,.\, t \,.\, n!/2^k$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.
- If $|S_x| = 2n!$ then

  $\text{Exp}_r[\,|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}|\,] \geq 3/2 \,.\, t \,.\, n!/2^k$.

- If $|S_x| = n!$ then

  $\text{Exp}_r[\,|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}|\,] \leq t.\, n!/2^k$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function q(.) s.t.

  $|S_x| = 2n!$ (large) ➡ Pr_r [∃y s.t. M(x, y, r) = 1] ≥ 2/3

  $|S_x| = n!$ (small) ➡ Pr_r [∀y s.t. M(x, y, r) = 0] ≥ 2/3.

- *Proof.* For every p ∈ [t]: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- If $|S_x| = 2n!$ then

  Exp_r [|{p∈[t] : ∃$u_p \in S_x$ s.t. $h_{i_p}(u_p) = 0^k$}|] ≥ 3/2 . t . $n!/2^k$.

- If $|S_x| = n!$ then

  ↕ gap

  Exp_r [|{p∈[t] : ∃$u_p \in S_x$ s.t. $h_{i_p}(u_p) = 0^k$}|] ≤ t. $n!/2^k$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

- If $|S_x| = 2n!$, by Chernoff bd. & $n!/2^k \in [1/8, 1/4]$,

  $\Pr_r[|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}| \geq 1.4.\, t.\, n!/2^k] \geq 2/3$.

- If $|S_x| = n!$, by Chernoff/Markov bd. & $n!/2^k \in [1/8, 1/4]$

  $\Pr_r[|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}| < 1.4.\, t.\, n!/2^k] \geq 2/3$.

(Easy homework)

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$. $t^* = 1.4.\ t.\ n!/2^k$
- If $|S_x| = 2n!$, by Chernoff bd. & $n!/2^k \in [1/8, 1/4]$,

  $\Pr_r[|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}| \geq 1.4.\ t.\ n!/2^k] \geq 2/3$.

- If $|S_x| = n!$, by Chernoff/Markov bd. & $n!/2^k \in [1/8, 1/4]$

  $\Pr_r[|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}| < 1.4.\ t.\ n!/2^k] \geq 2/3$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r [\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r [\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$.

  $t^* = 1.4.\, t.\, n!/2^k$

- If $|S_x| = 2n!$ then

  $\Pr_r [|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}| \geq t^*] \geq 2/3$.

- If $|S_x| = n!$ then

  $\Pr_r [|\{p \in [t] : \exists u_p \in S_x \text{ s.t. } h_{i_p}(u_p) = 0^k\}| < t^*] \geq 2/3$.

Set lower bound protocol (contd.)

- Lemma *. There's a poly-time TM M that takes input $x = (G_1, G_2)$, y & r, and a polynomial function $q(.)$ s.t.

  $|S_x| = 2n!$ (large) ➡ $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$

  $|S_x| = n!$ (small) ➡ $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

- *Proof.* For every $p \in [t]$: M uses c_p & x to check if $u_p \in S_x$. If yes, M checks if $h_{i_p}(u_p) = 0^k$. $t^* = 1.4.\ t.\ n!/2^k$
- If $|S_x| = 2n!$ then

  $\Pr_r[\exists y \text{ s.t. } M(x, y, r) = 1] \geq 2/3$.

- If $|S_x| = n!$ then

  $\Pr_r[\forall y \text{ s.t. } M(x, y, r) = 0] \geq 2/3$.

∎