CSC 2426: Fundamentals of Cryptography **Fall 2023**

Lecture 4: Goldreich-Levin & Computational Indistinguishability

Instructor: Akshayaram Srinivasan *Scribe: Yug Shah*

Date: October 2 2023

Theorem 4.1 (Goldreich-Levin Theorem) *If injective one-way functions (OWFs) exist, then* $\exists\{g_n, h_n\}_{n\in\mathbb{N}}$ *such that,* h_n *is a hard-core predicate for* g_n

4.1 Proof of Goldreich-Levin Theorem (continued)

In sections 3.1.1 and 3.1.2 we discussed the proofs for the trivial case where $\mathcal{A}$ predicts h with probability 1 and the non-trivial case where this probability is weakened to be at-least $\frac{3}{4} + \epsilon(n)$ where $\epsilon(n) = \frac{1}{poly(n)}$.

4.1.1 General Case

Now, we move onto the general case, wherein we assume that $\mathcal{A}$, given random (x, r) computes $h(x, r)$ with probability $\frac{1}{2} + \epsilon(n)$. (where $\epsilon(n) = \frac{1}{poly(n)}$), i.e.

$$\Pr_{(x,r)\leftarrow\{0,1\}^{2k(n)}}\left[\mathcal{A}(1^{2k(n)}, f_n(x)||r) = h_n(x, r)\right] \geq \frac{1}{2} + \frac{1}{p(n)}$$

for infinitely many n.

Similar to the non-trivial case, we define the set Good_n as:

$$\text{Good}_n := \left\{x \in \{0,1\}_{k(n)}|_{r\leftarrow\{0,1\}_{k(n)}} \Pr\left[\mathcal{A}(1^{2k(n)}, f_n(x)||r) = h_n(x, r)\right] \geq \frac{1}{2} + \frac{1}{2p(n)}\right\}$$

Claim 4.2 $\underset{x\leftarrow\{0,1\}^{k(n)}}{Pr}[x \in \textbf{Good}_n] \geq \frac{1}{2p(n)}$

Proof:

$$\begin{aligned}
\frac{1}{2} + \frac{1}{p(n)} &\leq \Pr_{x,r}[\mathcal{A} \text{ predicts } h_n] \\
&= \Pr_x[x \in \text{Good}_n].\Pr_r[\mathcal{A} \text{ predicts } h_n|x \in \text{Good}_n] + \Pr_x[x \notin \text{Good}_n].\Pr_r[\mathcal{A} \text{ predicts } h_n|x \notin \text{Good}_n] \\
&\leq \Pr_x[x \in \text{Good}_n] + \Pr_r[\mathcal{A} \text{ predicts } h_n|x \notin \text{Good}_n] \\
&\leq \Pr_x[x \in \text{Good}_n] + \frac{1}{2} + \frac{1}{2p(n)}
\end{aligned}$$

This shows that $\Pr_x[x \in \text{Good}_n] \geq \frac{1}{2p(n)}$ ■

Given a $x \in \text{Good}_n$, and that we have $r_1, r_2, \ldots, r_m$, the probability that $\mathcal{A}$ correctly guesses each of $r_i \oplus e_i$ is $\geq \frac{1}{2} + \frac{1}{2p(n)}$.

Let's come back to the analysis of the previous case from last lecture. Here, we are going to prove that even if $r_1, \ldots, r_m$ are not completely independent, but only pairwise independent, then the same procedure of taking the majority of the values still work.

Similar to the proof for the non-trivial case, we can again model each attempt with a random variable, Z_j with $j = 1, \ldots, m$, as, $Z_j = \begin{cases} 1 & , \iff x_i \text{ is correctly computed in the } j^{th} \text{ run} \\ 0 & , \text{otherwise} \end{cases}$. This gives us $\Pr[Z_j = 1] \geq \frac{1}{2} + \frac{1}{p(n)}$, and let $Z = \sum_{i=1}^{m} Z_i$.

$$\mathbb{E}[Z] = \sum \mathbb{E}[Z_i] \qquad \text{(since linearity holds for any random variable)}$$
$$\mathbb{E}[Z] \geq m\left(\frac{1}{2} + \frac{1}{p(n)}\right)$$

We fix $\mathbb{E}[Z] = m\left(\frac{1}{2} + \frac{1}{p(n)}\right)$ (as a larger expectation will make our case easier) and use Chebyshev's Inequality to find:

$$\Pr\left[Z \leq \frac{m}{2}\right] \leq \Pr\left[|Z - \mathbb{E}[Z]| \geq \frac{m}{p(n)}\right] \leq \frac{\text{Var}(Z)}{\left(\frac{m}{p(n)}\right)^2}$$
$$\text{P} = \Pr\left[\left|Z - \left(\frac{m}{2} + \frac{m}{p(n)}\right)\right| \geq \frac{m}{p(n)}\right] \leq \frac{m\text{Var}(Z_1)}{\left(\frac{m}{p(n)}\right)^2} \quad (\text{Var}(Z) = \sum \text{Var}(Z_i) \text{ when } Z_i\text{'s are pairwise independent})$$

Also,

$$\begin{aligned} \text{Var}(Z_1) &= \mathbb{E}\left[Z_1^2\right] - \left[\mathbb{E}[Z_1]\right]^2 \\ &= p - p^2 \leq \frac{1}{4} \\ &\implies m\text{Var}(Z_1) \leq \frac{m}{4} \end{aligned}$$

Substituting this value in the equation for P above, we get

$$\begin{aligned} \text{P} &\leq \frac{m}{4\left(\frac{m^2}{p^2(n)}\right)} \\ &\leq \frac{p^2(n)}{4m} = \frac{1}{4n} \qquad (\text{set } m = n.p^2(n)) \end{aligned}$$

$$\Pr_x[\text{at-least 1 } x_i \text{ is wrong}] = n.P = n\left(\frac{1}{4n}\right) = \frac{1}{4}$$
$$\therefore \Pr_x[\text{no } x_i \text{ is wrong}] = 1 - n.P = \frac{3}{4}$$

In order to construct $\mathcal{B}$, we first assume that it can call a black-box $\mathcal{C}$, which returns correct guesses for samples $(b_1 = \langle x, r_1\rangle, r_1), (b_2 = \langle x, r_2\rangle, r_2), \ldots, (b_m = \langle x, r_m\rangle, r_m)$ where $r_1, \ldots, r_m$ are random and independent, with probability $\frac{1}{q(n)}$ (The working of $\mathcal{C}$ is discussed in section 4.1.2)

Now, the probability of inverting x, i.e. probability that $\mathcal{B}$ inverts f is given by

$$\begin{aligned}\Pr[\text{inverting x}] &= \left(\Pr_x[x \in \text{Good}_n]\right)\left(\Pr_{x,r}[\text{black-box gives correct guesses}]\right)\left(\Pr_x[\text{no } x_i \text{ is wrong}]\right)\\ &= \left(\frac{1}{2p(n)}\right)\left(\frac{1}{q(n)}\right)\left(\frac{3}{4}\right)\\ &= \frac{3}{8p(n)q(n)}\end{aligned}$$

Here, $\frac{1}{p(n)q(n)}$ is non-negligible, hence, $\mathcal{B}$ inverts a OWF f with non-negligible probability, which contradicts the one-wayness of f

4.1.2 Working of black-box $\mathcal{C}$

One of the key observations from the non-trivial case was that the values of r can be fixed and reused for different x_i's. We also do not need these r's to be completely independent. We can use pairwise independent sampling of these r's to generate our guesses. As we have already seen, Chebyshev's Inequality can be used with pairwise independent sampling to bound the expected value. In this section, we show how to generate pairwise independent values for r

In order to generate pairwise independent samples with probability $\geq \frac{1}{m}$, we take $\log(m)$ samples $s_1, s_2, \ldots, s_{\log(m)}$ from $\{0,1\}^n$. We define $\tau_i = 1, 2, \ldots, \log(m)$ as the set of all bits that are 1 in the bit-representation of i (note that $i \in [m]$ can be represented with $\log m$ bits). Now, we set $r_i = \bigoplus_{j \in \tau_i} s_j$ and by linearity we get, $\langle x, r_i \rangle = \bigoplus_{j \in \tau_i} \langle x, s_j \rangle$

Now,

$$\Pr[\text{all } \langle x, r_i \rangle \text{ are correct}] \geq \frac{1}{m} = \frac{1}{q(n)} \qquad \text{for some polynomial } q(n)$$

4.1.3 Comments/Thoughts on $\mathcal{C}$

Let's take a step back and question what we did with the black-box $\mathcal{C}$. Given a short seed, of $\log(m)$ strings, we somehow managed to expand it to longer strings such that they are pairwise independent. A natural follow-up question would be whether it is possible to generate long, random and independent strings, given a short string/seed?

As a direct result of Shannon's Source Coding Theorem, a classic result in information theory, there is no *deterministic* way to generate such long, random and independent strings from a short seed, for computationally unbounded adversaries. Surprisingly enough, it is possible for computationally bounded adversaries!

4.2 Computational Indistinguishability

Definition 4.3 (Distribution) *X is a distribution over a sample space S if it assigns a probability p_s to the element $s \in S$ such that $\sum_s p_s = 1$*

Definition 4.4 (Support of a discrete random variable) *The support of a discrete random variable X is the set $Supp(X) := \{x | Pr[X = x] > 0\}$*

From the perspective of a computationally bounded tests, when can we sat that two distributions are identically distributed?

Let X, Y be random variables defined over the same support, $Supp(X)$ and let $x \in Supp(X)$. We say that X and Y are identically distributed when $\Pr[X = x] = \Pr[Y = x]$

Let's carry out a thought experiment where we have a deterministic algorithm $\mathcal{A}$ that distinguishes between two distributions. When we have two identically distributed random variables, $X \equiv Y$, we can see that the value

$$\Pr_{x \leftarrow X}[\mathcal{A}(x) = 1] - \Pr_{y \leftarrow Y}[\mathcal{A}(y) = 1] = 0$$

This implies that no deterministic algorithm, $\mathcal{A}$ can distinguish between two identically distributed random variables X and Y. This can be further formalised into the notion of computational indistinguishability

Definition 4.5 (Ensemble) *A sequence $\{X_n\}_{n \in \mathbb{N}}$ is called an ensemble if for each $n \in \mathbb{N}, X_n$ is a probability distribution over $\{0,1\}^*$*

Definition 4.6 (Computational Indistinguishability) *Two ensembles, $\{X_n\}_n$ and $\{Y_n\}_n$ are said to be computationally indistinguishable i.e. $\{X_n\}_n \approx_c \{Y_n\}_n$ if $\forall$ non-uniform p.p.t. $\mathcal{A}$ (also called the "distinguisher"),*

$$\left| \Pr_{t \leftarrow X_n}[\mathcal{A}(1^n, t) = 1] - \Pr_{t \leftarrow Y_n}[\mathcal{A}(1^n, t) = 1] \right| \leq \epsilon(n)$$

where $\epsilon(n)$ is negligible. The absolute difference between the two probabilities in the LHS is known as the advantage.

We can also think of it as, two ensembles are computationally indistinguishable if there is no efficient distinguisher $\mathcal{A}$ that can tell them apart with a non-negligible(i.e. better than negligible) advantage.

4.2.1 Properties

This section highlights some important properties of computational indistinguishability

- Closure Under Efficient Operations:

  If two distributions are indistinguishable, then the outputs of a p.p.t. algorithm run on both, also remains indistinguishable.

  Lemma 4.7 (Closure Under Efficient Operations) *If $\{X_n\}_n \approx_c \{Y_n\}_n$ and P is some p.p.t algorithm, then $\{P(X_n)\}_n \approx_c \{P(Y_n)\}_n$*

  Proof:

  Suppose we have some non-uniform p.p.t, $\mathcal{A}$ and polynomial $p(n)$ such that for infinitely many n, we have:

  $$\left| \Pr_{x \leftarrow P(X_n)}[\mathcal{A}(1^n, x) = 1] - \Pr_{x \leftarrow P(Y_n)}[\mathcal{A}(1^n, x) = 1] \right| > \frac{1}{p(n)}$$

  This allows us to construct another non-uniform p.p.t., $\mathcal{A}'(.) = \mathcal{A}(P(.))$ such that

  $$\left| \Pr_{x \leftarrow X_n}[\mathcal{A}'(1^n, x) = 1] - \Pr_{x \leftarrow Y_n}[\mathcal{A}'(1^n, x) = 1] \right| > \frac{1}{p(n)}$$

for infinitely many n.

Hence, $\mathcal{A}'$ distinguishes between $\{X_n\}_n$ and $\{Y_n\}_n$ which contradicts our assumption that $\{X_n\}_n \approx_c \{Y_n\}_n$

■

- Transitivity - The Hybrid Lemma:

  Computational indistinguishability is transitive over $poly(n)$ distributions i.e.

  If $\{X_n^1\}_n \approx_c \{X_n^2\}_n, \{X_n^2\}_n \approx_c \{X_n^3\}_n, \ldots \{X_n^{k-1}\}_n \approx_c \{X_n^k\}_n$ for some $k \leq \mathsf{poly}(n)$, then, $\{X_n^1\}_n \approx_c \{X_n^k\}_n$

  Lemma 4.8 (Hybrid Lemma) *Let $X^1, X^2, \ldots X^k$ be a sequence of distribution ensembles such that for for each $i \in [1, k-1]$, we have $X^i \approx_c X^{i+1}$. We then have $X^1 \approx_c X^k$.*

  Proof:

  Let us fix an nuPPT adversary $\mathcal{A}$. Since $X^i \approx_c X^{i+1}$, we have:

  $$|\Pr_{x \leftarrow X_n^i}[\mathcal{A}(1^n, x) = 1] - \Pr_{x \leftarrow X_n^{i+1}}[\mathcal{A}(1^n, x) = 1]| \leq \mu_i(n)$$

  where μ_i is a negligible function.

  $$\begin{aligned} |\Pr_{x \leftarrow X_n^1}[\mathcal{A}(1^n, x) = 1] - \Pr_{x \leftarrow X_n^k}[\mathcal{A}(1^n, x) = 1]| &\leq \sum_{i=1}^{k-1} |\Pr_{x \leftarrow X_n^i}[\mathcal{A}(1^n, x) = 1] - \Pr_{x \leftarrow X_n^{i+1}}[\mathcal{A}(1^n, x) = 1]| \\ &\leq \sum_{i=1}^{k-1} \mu_i(n) \\ &\leq \mathsf{negl}(n) \end{aligned}$$

  In the last inequality, we are using the fact that sum of a polynomial number of neglgible functions is negligible.

  ■

4.3 Pseudorandom Generator(PRG)

A function G,

$$G \coloneqq \{G_n : \{0,1\}^{k(n)} \to \{0,1\}^{m(n)}\} \qquad\qquad m(n) > k(n)$$

is a Pseudorandom Generator if it satisfies the following conditions:

(Note- The difference, $m(n) - k(n)$ is also defined as the stretch of the PRG)

1. **Efficiently Computable:**

   $\exists$ some polynomial time algorithm M such that $M\left(x \in \{0,1\}^{k(n)}\right) = G_n(x)$

2. **Pseudorandom:**

$\forall$ non-uniform p.p.t. $\mathcal{A}$,

$$\left| \underbrace{\Pr_{x \leftarrow \{0,1\}^{k(n)}} [\mathcal{A}(1^n, G_n(x)) = 1]}_{D_1 := (x \leftarrow \{0,1\}^{k(n)}, G_n(x))} - \underbrace{\Pr_{y \leftarrow \{0,1\}^{m(n)}} [\mathcal{A}(1^n, y) = 1]}_{D_2 := (y \leftarrow \{0,1\}^{m(n)}, y)} \right| \leq \epsilon(n)$$

where $\epsilon(n)$ is negligible.

The second distribution can also be seen as the uniform distribution. This property also means that a distribution is pseudorandom if it is indistinguishable from the uniform distribution. $m(n) - k(n)$ is called the stretch of the PRG.

4.3.1 Expanding the Stretch

We now show that if there is a PRG with a stretch of one bit, then we can construct a PRG with arbitrary polynomial stretch.

Theorem 4.9 *Let $k(n) = n$ and suppose there exists $G_n : \{0,1\}^n \to \{0,1\}^{n+1}$, then we can construct $G' : \{0,1\}^n \to \{0,1\}^{p(n)}$ for arbitrary polynomial $p(n)$.*

Proof: The idea is to use G_n recursively for $p(n)$ times as follows:

$$\begin{aligned} G_n(x) &\to (\overbrace{x_1}^{\text{n bits}}, \overbrace{\sigma_1}^{\text{1 bit}}) \\ G_n(x_1) &\to (x_2, \sigma_2) \\ G_n(x_2) &\to (x_3, \sigma_3) \\ &\vdots \\ G_n(x_{p(n)-1}) &\to (x_{p(n)}, \sigma_{p(n)}) \end{aligned}$$

We take all the σ_i's and show that $\left(\sigma_1, \sigma_2, \ldots, \sigma_{p(n)}\right)$ is pseudorandom. We do this by showing that all of these σ_i's are computationally indistinguishable. The inductive nature of how these σ_i's are generated creates 3 categories as follows:

1. **Hyb$_{origin}$-**

   $x \leftarrow \{0,1\}^n$ and run the run the PRG $G_n(x)$ recursively to get values for $(\sigma_1, \ldots, \sigma_p(n))$. From the 2^{nd} category, we notice that $\text{Hyb}_0 \equiv \text{Hyb}_{origin}$

2. **Hyb$_i$-**

   $x \leftarrow \{0,1\}^n$, $x_i \leftarrow \{0,1\}^n$ and sample i values for $(\sigma_1, \ldots, \sigma_i) \leftarrow \{0,1\}^i$.

   Now, run the PRG $G_n(x_i) = (x_{i+1}, \sigma_{i+1})$ recursively, until we generate values for $\left(\sigma_{i+1}, \ldots, \sigma_{p(n)}\right)$ to get all the values for $\left(\sigma_1, \ldots, \sigma_{p(n)}\right)$

3. **Hyb$_{final}$-**

   Sample all the values for $(\sigma_1, \ldots, \sigma_p(n)) \leftarrow \{0,1\}^p(n)$. From the 2^{nd} category, we see that $\text{Hyb}_{p(n)} \equiv \text{Hyb}_{final}$

Note that $\text{Hyb}_0 \approx_c \text{Hyb}_{origin}$ and $\text{Hyb}_{p(n)} \approx_c \text{Hyb}_{final}$. To show that $\text{Hyb}_0 \approx_c \text{Hyb}_{p(n)}$, all we need to show is $\text{Hyb}_{i-1} \approx_c \text{Hyb}_i$. Following is an informal outline of the proof.

For $\text{Hyb}_{i-1} : (\sigma_1, \ldots, \sigma_{i-1}) \leftarrow \{0,1\}^{i-1}$ and $x_{i-1} \leftarrow \{0,1\}^n$. Run the PRG $G_n(x_{i-1}) \rightarrow (x_i, \sigma_i)$ recursively until we get the outputs $(\sigma_1, \ldots, \sigma_p(n))$

For $\text{Hyb}_i : (\sigma_1, \ldots, \sigma_i) \leftarrow \{0,1\}^i$ and $x_i \leftarrow \{0,1\}^n$. Run the PRG $G_n(x_i) \rightarrow (x_{i+1}, \sigma_{i+1})$ recursively until we get the outputs $(\sigma_1, \ldots, \sigma_p(n))$

Between Hyb_{i-1} and Hyb_i, the only difference is whether the i^{th} bit is randomly sampled or generated by the PRG. This difference is the same as the one between pseudorandom G_n and the uniform distribution. We will provide a formal reduction below.

Assume for the sake of contradiction that Hyb_i and Hyb_{i-1} are distinguishable. This means that there exists an nuPPT adversary $\mathcal{A}$ that can distinguish between Hyb_i and Hyb_{i-1} with advantage $\frac{1}{p(n)}$ for some polynoomial $p(\cdot)$ (for infinitely many n).

We will now use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that can distinguish the output of the PRG from random with non-negligible advantage. $\mathcal{B}$ on input 1^n and a string $y \in \{0,1\}^{n+1}$ which is either uniformly random or is the output of the PRG does the following. It parses y as (x_i, σ_i). It samples $\sigma_1, \ldots, \sigma_{i-1}$ uniformly at random from $\{0,1\}^{i-1}$. It generates $\sigma_{i+1}, \ldots, \sigma_{p(n)}$ exactly as described in Hyb_{i-1} (which is the same as the procedure described in Hyb_i). It runs $\mathcal{A}$ on $(\sigma_1, \ldots, \sigma_{p(n)})$ and outputs whatever $\mathcal{A}$ outputs. Note that if (x_i, σ_i) is generated as the output of a PRG on a uniformly chosen seed, then the input to $\mathcal{A}$ is distributed identically to Hyb_{i-1}. Else, it's output is distributed identically to Hyb_i. Since we assumed that $\mathcal{A}$ can distinguish Hyb_i and Hyb_{i-1} with advantage $\frac{1}{p(n)}$, we have that $\mathcal{B}$ breaks the pseudorandomness of PRG which is a contradiction.

Thus, from the definition of computational indistinguishability,

$$\implies \text{Hyb}_{i-1} \approx_c \text{Hyb}_i$$
$$\implies \text{Hyb}_0 \approx_c \text{Hyb}_{p(n)} \text{ i.e. } \text{Hyb}_{origin} \approx_c \text{Hyb}_{final} \text{ (By Hybrid Lemma)}$$

■