CS 7880 Graduate Cryptography October 08, 2015

Lecture 8: Goldreich-Levin Theorem (continued)

Lecturer: Daniel Wichs *Scribe: Tanay Mehta*

1 Topics Covered

- Finish proof of Goldreich-Levin Theorem
- Constructions of PRGs from OWFs

2 Goldreich-Levin Theorem

Last time, we began the proof of the Goldreich-Levin Theorem, which we will state again.

Theorem 1 (Goldreich-Levin) *Let f be a one-way function and define*

$$g(x, r) = (f(x), r)$$

where $|x| = |r|$. Define

$$hc(x, r) = \langle x, r \rangle$$

where $\langle x, r \rangle$ is the usual inner product on $(\mathbb{Z}/2\mathbb{Z})$. Then, g is also a one-way function, and hc is a hardcore predicate of g.

We also stated the following two claims and proved the first.

Claim 1 *If $\Pr_{x,r}[P(f(x), r) = \langle x, r \rangle] \geq \frac{1}{2} + \varepsilon(n)$ then for all $n \in \mathbb{N}$ there exists a set $G_n \subseteq \{0,1\}^n$ of size $|G_n| \geq \frac{\varepsilon(n)}{2} 2^n$ such that for all $x \in G_n$*

$$\Pr_r[P(f(x), r) = \langle x, r \rangle] \geq \frac{1}{2} + \frac{\varepsilon(n)}{2}$$

Claim 2 (Decoder) *For any $\delta(n) = \frac{1}{O(n^c)}$ and constant c, there exists a probabilistic polynomial time algorithm Dec^O (decoder with oracle O) and there exists $p(n) = poly(n)$ such that for all $n \in \mathbb{N}$ and for all $x \in \{0,1\}^n$ we have*

$$\Pr_r[O(r) = \langle x, r \rangle] \geq \frac{1}{2} + \delta(n) \Rightarrow \Pr[Dec^O(1^n) = x] \geq \frac{1}{p(n)}$$

Note that the second probability function is over the randomness of Dec^O.

Let us interpret this claim from the context of coding theory. We can think of $x \in \{0,1\}^n$ as a message and define an exponentially long codeword $c \in \{0,1\}^{2^n}$ defined by $c[r] = \langle x, r \rangle$

for all $r \in \{0,1\}^n$, where we can think of r as denoting a position between $0, \ldots, 2^n - 1$ in binary. We can think of the oracle O as defining a "noisy codewrod" c' defined via $c'[r] = O(r)$ with $(1/2 - \delta(n))n$ errors where $c'[r] \neq c[r]$. From coding theory, we know that c' cannot be uniquely decoded to recover the correct message x with complete certainty unless the number of errors is $\leq (1/4)n$, meaning that $\delta(n) \geq 1/4$. This holds even if we didn't care about the efficiency of the decoder and it could read all of c'.
The implication of the above claim tells us two things:

1. Although c' cannot be uniquely decoded to recover the correct message x with complete certainty, it can be decoded to recover x with probability $1/p(n)$ for some polynomial $p(n)$. By running such a decoder many times, we can recover a polynomial size *list* of candidate values that contains x with extremely high probability. This is called *list decoding* and, in general, it allows us to recover from much higher error rates than unique decoding.

2. Not only can we decode c' to recover x with probability $1/p(n)$, we can do so in polynomial time by only querying the oracle $O(r) = c'[r]$ at polynomially many positions r. This is called *local decoding*, meaning that we do not need to read the entire codeword to recover the message. This is essential in our setting since the codeword c' is of size 2^n and therefore reading it in its entirety (i.e., querying O at all values r) would take exponential time.

Let us return to the proof of Goldreich-Levin. We will complete the proof of the theorem assuming that the decoder claim is true, and complete it's proof afterwards.

Proof: We proceed by contradiction. Assume hc is not a hardcore predicate of g. Using the unpredictability definition of hardcore predicates, there exists a probabilistic polynomial-time algorithm P and $\varepsilon(n) \neq negl(n)$ such that

$$\Pr_{x,r}[P(f(x), r) = \langle x, r \rangle] \geq \frac{1}{2} + \varepsilon(n)$$

where the probability is over $x, r \leftarrow \{0,1\}^n$. We show that this means we can invert the one-way function f.

By the first claim, we have that for all $n \in \mathbb{N}$ there exists $G_n \subseteq \{0,1\}^n$ such that $|G_n| \geq \frac{\varepsilon(n)}{2} 2^n$ such that for all $x \in G_n$

$$\Pr_{r}[P(f(x), r) = \langle x, r \rangle] \geq \frac{1}{2} + \frac{\varepsilon(n)}{2}$$

where the probability is over $r \leftarrow \{0,1\}^n$.

We wish to invert $y = f(x)$ by applying the decoder from the second claim with the oracle $O(\cdot) = P(y, \cdot)$ so that $O(r) = P(y, r)$. Intuitively, we want to fix the parameter $\delta(n)$ of the decoder to $\delta(n) = \frac{\varepsilon(n)}{2}$. However, we only know that $\varepsilon(n)$ is *not* negligible which is not the same thing as $\varepsilon(n) = 1/\mathsf{poly}(n)$. Instead, $\varepsilon(n) \neq \mathsf{negl}(n)$ implies that for infinitely many $n \in \mathbb{N}$ (let's call these "good" n) and some constant c we have $\varepsilon(n) > \frac{1}{n^c}$. We will set

$\delta(n) = \frac{1}{2n^c}$ so that $\varepsilon(n) \geq \delta(n)$ for all "good" n. This ensures that for all "good" n and all $x \in G_n$ we have

$$\Pr[P(f(x), r) = \langle x, r \rangle] \geq \frac{1}{2} + \delta(n)$$

Now using the decoder claim, we will define the inverter for the one-way function f,

$$A(y) = \{ \text{ Output } Dec^{P(y,\cdot)}(1^n) \quad // \ Dec \text{ is tailored to } \delta(n)$$

and obtain that for all "good" n and all $x \in G_n$ we have:

$$\Pr[A(f(x)) = x] = \Pr[Dec^{P(f(x),\cdot)}(1^n) = x] \ \geq \frac{1}{p(n)}.$$

where the randomness is over the random choices of A and p is some polynomial. To finish the proof, we take the probability over a random $x \leftarrow \{0,1\}^n$. Then for all "good" $n \in \mathbb{N}$

$$\begin{aligned} \Pr_x[A(f(x)) = x] &\geq \Pr_x[A(f(x)) = x | x \in G_n] \cdot \Pr_x[x \in G_n] \\ &\geq \frac{1}{p(n)} \cdot \frac{\varepsilon(n)}{2} = \frac{1}{p(n)} \cdot \delta(n) \end{aligned}$$

Thus, we have that $\Pr[A(f(x)) = x] \neq \mathsf{negl}(n)$ and that f is not a one-way function. □

Now we need to prove that the decoder claim is true. We will do this by designing the decoder $Dec^O(1^n)$. Recall the simple cases we looked at last lecture to build intuition. We wanted to learn the i^{th} bit of x. Let $b_1 = O(r)$ and $b_i = O(r + e_i)$. Then, if the oracle answers correctly both times we have $b_1 = \langle x, r \rangle$ and $b_i = \langle x, r + e_i \rangle$ which means that we can recover

$$x_i = b_i - b_1 = \langle x, e_i \rangle$$

However, the above is only meaningful if the probability that the oracle answers correctly twice is bigger than a half, which is only meaningful if the oracle answers correctly with probability $> 3/4$ (by taking union bound). The main intuition behind the actual strategy is to query the oracle on many pairwise independent inputs. In that case, even if the oracle answers correctly with probability just slightly higher than $> 1/2$ we can argue that the majority of the answers are likely to be correct using the Chebyshev bound.

Proof: We start out by defining the decoder.

$Dec^O(1^n)$:

1. Set $\ell = \lceil \log(\frac{n}{\delta^2(n)} + 1) \rceil = O(\log n)$.
   //This is for technical reasons that will become clear later.
2. Choose $s_1, \ldots, s_\ell \leftarrow \{0,1\}^n$
   $\sigma_1, \ldots, \sigma_\ell \leftarrow \{0,1\}$
3. For all $I \subseteq [\ell] = 1, \ldots, l$, where $I \neq \emptyset$
   $r^I := \sum_{i \in I} s_i \pmod 2$
   $\sigma^I := \sum_{i \in I} \sigma_i \pmod 2$

4. For all $j \in [n]$

   For all $I \subseteq [\ell]$, $I \neq \emptyset$

   $$\tilde{x}_j^I := O(r^I + e_j) - \sigma^I$$

   Set $\tilde{x}_j := \text{majority}\{\tilde{x}_j^I\}$

5. Output $(\tilde{x}_1, \ldots, \tilde{x}_n)$

Note that for any $I_1 \neq I_2 \subseteq [\ell]$ with $I_1, I_2 \neq \emptyset$ the values r^{I_2}, r^{I_1} are uniformly random and independent. This is because if $I_1 \neq I_2$ then there is some $i \in I_1$ but $i \notin I_2$ (or vice versa), and therefore even conditioned on the value of r^{I_2}, the random variable r^{I_1} is random and independent over the choice of s_i. This means that the random variables r^I are pairwise independent.

We will now proceed with a series of claims.

Define the event LG ("lucky guess") to be $\sigma_i = \langle x, s_i \rangle$ for all $i \in [\ell]$.

Define the events E_j^I to be the event that $O(r^I + e_j) = \langle x, r^I + e_j \rangle$.

Claim 3 *If LG occurs and E_j^I occurs then $\tilde{x}_j^I = x_j$.*
If LG occurs and $\sum_j E_j^I > \frac{1}{2}(2^\ell - 1)$ then $\tilde{x}_j = x_j$.

Proof: If LG occurs, then for all $I \subseteq [\ell]$:

$$\sigma^I = \sum_{i \in I} \sigma_i = \sum_{i \in I} \langle x, s_i \rangle = \langle x, r^I \rangle$$

by linearity of inner product. If LG and E_j^I occur then:

$$\tilde{x}_j^I = O(r^I + e_j) - \sigma^I = \langle x, r^I + e_j \rangle - \langle x, r^I \rangle = \langle x, e_j \rangle = x_j.$$

The second part follows since if $\sum_j E_j^I > \frac{1}{2}(2^\ell - 1)$ (majority of the events occur) then $x_j = \text{majority}\{\tilde{x}_j^I\}$. □

Claim 4 *For any $j \in [n], I \subseteq [\ell], I \neq \emptyset$: $\Pr[E_j^I = 1] \geq \frac{1}{2} + \delta(n)$. Furthermore for any j, the events $\{E_j^I\}_I$ are pairwise indepdendent, and also independent of LG.*

Proof: This follows because the values r^I are uniformly random are pairwise independent. Furthermore the values r^I only depend on s_i while the event LG happens with probability $2^{-\ell}$ over the choice of σ_i no matter what values s_i take on and therefore is independent of the values s_i. □

Claim 5 *For all $j \in [n]$:*

$$\Pr[\sum_j E_j^I \leq \frac{1}{2}(2^\ell - 1) \mid LG] = \Pr[\sum_j E_j^I \leq \frac{1}{2}(2^\ell - 1)] \leq \frac{1}{4\delta^2(n) \cdot (2^\ell - 1)} \leq \frac{1}{4n}$$

Proof: The first part follows since the events $\{E_j^I\}$ independent of LG. The second part follows directly from Chebyshev inequality and the fact that the events $\{E_j^I\}$ are pairwise independent (proof omitted). □

The middle term of the Chebyshev Inequality is the reason we defined ℓ to be the way we did. We now finish off the proof of the decoder claim. For all $j \in [n]$

$$\Pr[\tilde{x}_j \neq x_j | LG] \leq \Pr[\sum_j E_j^I \leq \frac{1}{2}(2^\ell - 1) \mid LG] \leq \frac{1}{4n}$$

By the union bound, we have

$$\Pr[\exists j \ \tilde{x}_j \neq x_j | LG] \leq \frac{1}{4n} \cdot n = \frac{1}{4}$$

By taking the complement of the above probability,

$$\Pr[\tilde{x} = x | LG] \geq \frac{3}{4}$$

Finally, we have

$$\begin{aligned} \Pr[\tilde{x} = x] &\geq \Pr[\tilde{x} = x | LG] \cdot \Pr[LG] \\ &\geq \frac{3}{4} \cdot \Pr[LG] = \frac{3}{4} \cdot 2^{-\ell} = \frac{1}{\mathsf{poly}(n)}. \end{aligned}$$

□

Intuitively, we have picked $\ell = O(\log n)$ random vectors s_i and guessed their inner products with x to be $\langle x, s_i \rangle = \sigma_i$. Since ℓ is small, we guess correctly with probability $2^{-\ell} = 1/\mathsf{poly}(n)$. Then we try to recover every bit x_j of x by calling the oracle on many values $r^I + e_j$ which are pairwise independent but allow us to recover x_j if the majority of the answers is correct. Since the oracle is expected to answer correctly with probability $\frac{1}{2} + \delta(n)$ on each query, we can use the Chebyshev bound to argue that the majority of the answers is correct with high probability.

PRG from OWP. The Goldreich-Levin theorem allows us to construct a PRG from any one-way permutation (OWP).

Corollary 1 *If f is a one-way permutation (OWP), then g is a one-way permutation with hardcore predicate hc (as defined in Goldreich-Levin theorem) and $G(x) = (g(x), hc(x))$ is a pseudorandom generator.*

Proof:

$$G(U_n) \equiv (g(x), hc(x)) \approx (g(x), b) \equiv U_{n+1}$$

where $x \leftarrow \{0,1\}^n$, $b \leftarrow \{0,1\}$ and U_m denotes the uniform distribution over $\{0,1\}^m$. □

PRG from injective OWF. The above shows how to construct PRGs from OWPs. Using a slightly more complicated construction, we can construct a PRG from any injective one-way function. Let's say f is an injective one-way function but not necessarily a permutation. This implies that g is an injective one-way function with hardcore predicate $hc(x)$ (as defined in Goldreich-Levin theorem) but the above construction isn't necesserally a PRG since $g(x)$ is not necessarily uniformly random. Consider the function

$$g^m(\bar{x}) = (g(x_1), hc(x_1), \ldots, g(x_m), hc(x_m))$$

where $\bar{x} = (x_1, \ldots, x_m) \in \{0,1\}^{nm}$. Then

$$g^m(\bar{x}) = (g(x_1), hc(x_1), \ldots, g(x_m), hc(x_m)) \approx (g(x_1), b_1, \ldots, g(x_m), b_m)$$

where $b_1, \ldots, b_m \leftarrow \{0,1\}$. The left hand side has only mn bits of entropy but the right hand side has $mn + m$ bits of entropy. So we get m bits of computational entropy for free. We can then construct a PRG by relying on randomness extractors to convert entropy into a uniformly random output.

Claim 6 *If $m = \omega(\log n)$ and Ext is a universal hash function based extractor with a w-bit seed and $\ell = nm + 1$ bit output, then $G(\bar{x}, s) = (s, Ext(g^m(\bar{x}), s))$ is a pseudorandom generator with 1 bit stretch.*

Note: the input to the PRG consists of $n' = nm + w$ bits and the output is $n' + 1$ bits. We think of n rather then n' as the security parameter.

Proof:

$$\begin{aligned} G(\bar{x}, s) &\equiv (s, Ext((g(x_1), hc(x_1), \ldots, g(x_m), hc(x_m)), s)) \\ &\approx (s, Ext((g(x_1), b_1, \ldots, g(x_m), b_m), s)) \\ &\approx (U_w, U_{nm+1}) \end{aligned}$$

where $\bar{x} \leftarrow \{0,1\}^{nm}$, $s \leftarrow \{0,1\}^w$ and $b_i \leftarrow \{0,1\}$. The first $\approx$ comes from the Goldreich-Levin theorem. The second $\approx$ comes from the leftover hash lemma and relying on the fact that we have $nm+m$ bits of entropy and $\ell = nm+1$ bit output and therefore the statistical distance from uniform is $\varepsilon = 2^{-(m-1)/2} = \mathsf{negl}(n)$. □

It turns out that it's possible to construct a PRG from *any* one-way function. This requires more work and we will not prove this in class.