Lecture 8: Pseudorandomness - II

Instructor: Omkant Pandey

Spring 2017 (CSE 594)

Last Time

- Computational Indistinguishability & Prediction Advantage
- Pseudorandom Distributions & Next-bit Test
- Definition of a PRG

Today

- Pseudorandom Generators (PRG)
  - 1-bit stretch
  - Polynomial stretch
- Pseudorandom Functions (PRF)
  - Definition
  - PRF from any PRG
- Volunteers for scribe notes?

Recall

Definition (Pseudorandom Ensembles)

An ensemble $\{X_n\}$, where X_n is a distribution over $\{0,1\}^{\ell(n)}$, is said to be pseudorandom if:

$$\{X_n\} \approx \{U_{\ell(n)}\}$$

Definition (Next-bit Unpredictability)

An ensemble of distributions $\{X_n\}$ over $\{0,1\}^{\ell(n)}$ is next-bit unpredictable if, for all $0 \leqslant i < \ell(n)$ and n.u. PPT $\mathcal{A}$, $\exists$ negligible function $\nu(\cdot)$ s.t.:

$$\Pr[t = t_1 \ldots t_{\ell(n)} \sim X_n \colon \mathcal{A}(t_1 \ldots t_i) = t_{i+1}] \leqslant \frac{1}{2} + \nu(n)$$

Theorem (Completeness of Next-bit Test)

If $\{X_n\}$ is next-bit unpredictable then $\{X_n\}$ is pseudorandom.

Pseudorandom Generators (PRG)

Definition (Pseudorandom Generator)

A deterministic algorithm G is called a *pseudorandom generator* (PRG) if:

- G can be computed in polynomial time
- $|G(x)| > |x|$
- $\left\{x \leftarrow \{0,1\}^n : G(x)\right\} \approx_c \left\{U_{\ell(n)}\right\}$ where $\ell(n) = |G(0^n)|$

The **stretch** of G is defined as: $|G(x)| - |x|$

A PRG with 1-bit stretch

- Remember the hardcore predicate?
- It is hard to guess $h(s)$ even given $f(s)$
- Let $G(s) = f(s) \| h(s)$ where f is a OWF
- Some small issues:
  - $|f(s)|$ might be less than $|s|$
  - $f(s)$ may always start with prefix 101 (not random)
- **Solution:** let f be a one-way *permutation* (OWP) over $\{0,1\}^n$
  - Domain and Range are of same size, i.e., $|f(s)| = |s| = n$
  - $f(s)$ is uniformly random over $\{0,1\}^n$ if s is

    $\forall y : \Pr[f(s) = y] = \Pr[s = f^{-1}(y)] = 2^{-n}$

    $\Rightarrow f(s)$ is uniform and cannot start with a fix value!

A PRG with 1-bit stretch

- Let $f : \{0,1\}^* \to \{0,1\}^*$ be a **OWP**
- Let $h : \{0,1\}^* \to \{0,1\}$ be a hardcore predicate for f
- **Construction:** G is defined as:

$$G(s) = f(s) \parallel h(s)$$

Theorem (PRG based on OWP)

G is a pseudorandom generator with 1-bit stretch.

- If you did the exercise proof (from last class) that "next bit test" implies pseudorandomness, then this proof is trivial: if G is not a PRG, an attacker D must succeed in next bit test. But first n bits of $G(s)$ are uniform (since f is a permutation), so D must predict the $(n+1)$-th bit – which is the hardcore bit – with $1/2+$non-negligible. (contradiction)
- For completeness, we do a proof from scratch that relies on hardcore bits.

Proof that G is a 1-bit stretch PRG

Observe that G is deterministic and efficient because f, h are; also stretch $= |G(s)| = |s| + 1$ because f is a permutation which preserves length.

Next, we show: $\left\{s \leftarrow \{0,1\}^n : G(s)\right\} \approx_c \left\{U_{n+1}\right\}$

- By contradiction, suppose that it is not true. Then, $\exists$ efficient distinguisher D, a polynomial $q(\cdot)$ s.t.:

  $$\left|\Pr_{s \leftarrow \{0,1\}^n}\left[D(G(s)) = 1\right] - \Pr_{u \leftarrow U_{n+1}}\left[D(u) = 1\right]\right| \geqslant \frac{1}{q(n)}$$

  for infinitely many values of n.

- We show how to use D to break the OWP f. $\Rightarrow$ contradiction

Proof that G is a 1-bit stretch PRG (contd.)

Given: $\left|\Pr_{s\leftarrow\{0,1\}^n}\left[D(G(s))=1\right]-\Pr_{u\leftarrow U_{n+1}}\left[D(u)=1\right]\right|\geqslant\frac{1}{q(n)}$

- Write $u=u_1\ldots\|u_{n+1}=y\|u_{n+1}$ where $y\in\{0,1\}^n$.
- Since f is a permutation, $\exists$ a unique s s.t. $f(s)=y$
- y is uniform over $\{0,1\}^n$, therefore so is s.
- We have:

$$\begin{aligned}
&\Pr_{u\leftarrow U_{n+1}}\left[D(u)=1\right]\\
&=\Pr_{y\leftarrow\{0,1\}^n,u_{n+1}\leftarrow\{0,1\}}\left[D(y\|u_{n+1})=1\right]\\
&=\Pr_{s\leftarrow\{0,1\}^n,u_{n+1}\leftarrow\{0,1\}}\left[D(f(s)\|u_{n+1})=1\right]\\
&=\sum_{r\in\{0,1\}}\Big(\Pr_{u_{n+1}\leftarrow\{0,1\}}[u_{n+1}=r]\quad\times\\
&\qquad\qquad\Pr_{s\leftarrow\{0,1\}^n}\left[D(f(s)\|u_{n+1})=1|u_{n+1}=r\right]\Big)
\end{aligned}$$

Proof that G is a 1-bit stretch PRG (contd.)

$$\Pr_{u \leftarrow U_{n+1}}\left[D(u) = 1\right]$$

$$= \sum_{r \in \{0,1\}} \frac{1}{2} \cdot \Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| u_{n+1}) = 1 | u_{n+1} = r\right]$$

$$= \frac{1}{2} \cdot \sum_{r \in \{0,1\}} \Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| r) = 1\right]$$

$$= \frac{1}{2}\Big(\Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| 0) = 1\right] + \Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| 1) = 1\right] \Big)$$

$$= \frac{1}{2}\Big(\Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| h(s)) = 1\right] + \Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| \overline{h(s)}) = 1\right] \Big)$$

where $\overline{h(s)} = 1 - h(s)$

By definition of $G(s)$:
$\Pr_{s \leftarrow \{0,1\}^n}\left[D(G(s)) = 1\right] = \Pr_{s \leftarrow \{0,1\}^n}\left[D(f(s) \| h(s)) = 1\right]$

Subtract and take absolute value:

Proof that G is a 1-bit stretch PRG (contd.)

$$\left| \Pr_{u \leftarrow U_{n+1}} \left[D(u) = 1 \right] - \Pr_{s \leftarrow \{0,1\}^n} \left[D(G(s)) = 1 \right] \right|$$

$$= \frac{1}{2} \cdot \left| \Pr_{s \leftarrow \{0,1\}^n} \left[D(f(s) \| h(s)) = 1 \right] - \Pr_{s \leftarrow \{0,1\}^n} \left[D(f(s) \| \overline{h(s)}) = 1 \right] \right|$$

By equivalence claim, this is:

$$= \left| \Pr \left[b \leftarrow \{0,1\}; z \leftarrow X^b; D(z) = b \right] - \frac{1}{2} \right|$$

where:

- $X^0 := \left\{ s \leftarrow \{0,1\}^n : f(s) \| h(s) \right\}$
- $X^1 := \left\{ s \leftarrow \{0,1\}^n : f(s) \| \overline{h(s)} \right\}$
- $z = h(s) \oplus b$

Substitute and rewrite:

Proof that G is a 1-bit stretch PRG (contd.)

$$\left| \Pr_{u \leftarrow U_{n+1}} \left[D(u) = 1\right] - \Pr_{s \leftarrow \{0,1\}^n} \left[D(G(s)) = 1\right] \right|$$

$$= \left| \Pr \left[b \leftarrow \{0,1\}; s \leftarrow \{0,1\}^n; D(f(s) \| (h(s) \oplus b)) = b\right] - \tfrac{1}{2} \right|$$

$$= \left| \Pr_{b,s} \left[D(f(s) \| (h(s) \oplus b)) = b\right] - \tfrac{1}{2} \right|$$

But we are given that: L.H.S. $\geqslant \frac{1}{q(n)}$

Therefore: $\left| \Pr_{b,s} \left[D(f(s) \| (h(s) \oplus b)) = b\right] - \tfrac{1}{2} \right| \geqslant \frac{1}{q(n)}$

Write $r = h(s) \oplus b$ so that r is uniform if b is and $h(s) = r \oplus b$.

Substitute above and rewrite:

Proof that G is a 1-bit stretch PRG (contd.)

We get: $\left|\Pr_{r,s}\left[D(f(s)\|r) = b \;\bigwedge\; h(s) = r \oplus b\right] - \frac{1}{2}\right| \geqslant \frac{1}{q(n)}$

Without loss of generality, we can assume that probability is $\geqslant 1/2$.

Therefore: $\Pr_{r,s}\left[D(f(s)\|r) = b \;\bigwedge\; h(s) = r \oplus b\right] \geqslant \frac{1}{2} + \frac{1}{q(n)}$

Use D to predict hardcore bit as follows:

Algorithm $\mathcal{A}(f(s))$:

- sample bit r uniformly and compute $b \leftarrow D(f(s)\|r)$
- output $r \oplus b$.

$$\begin{aligned}\Pr_s\left[\mathcal{A}(f(s)) = h(s)\right] &= \Pr_{r,s}\left[D(f(s)\|r) = b \;\bigwedge\; h(s) = r \oplus b\right] \\ &\geqslant \frac{1}{2} + \frac{1}{q(n)} \qquad (\textit{contradiction}) \qquad \square\end{aligned}$$

One-bit stretch PRG $\implies$ Poly-stretch PRG

Intuition: Iterate the one-bit stretch PRG poly times

Construction of $G_{poly} : \{0,1\}^n \to \{0,1\}^{\ell(n)}$:

- Let $G : \{0,1\}^n \to \{0,1\}^{n+1}$ be a one-bit stretch PRG

$$\begin{array}{rcl} s & = & X_0 \\ G(X_0) & = & X_1 \| b_1 \\ & \vdots & \\ G(X_{\ell(n)-1}) & = & X_{\ell(n)} \| b_{\ell(n)} \end{array}$$

- $G_{poly}(s) := b_1 \ldots b_{\ell(n)}$

Think: Proof?

Proof that G_{poly} is pseudorandom

- Want: $\left\{s \leftarrow \{0,1\}^n : G_{poly}(s)\right\} \approx_c \left\{U_{\ell(n)}\right\}$
- Let D be any non-uniform PPT algorithm.

Step 0:

$$\begin{array}{rcl} \multicolumn{3}{c}{\underline{\text{Experiment } H_0}} \\ s & = & X_0 \\ G(X_0) & = & X_1 \| b_1 \\ G(X_1) & = & X_2 \| b_2 \\ & \vdots & \\ G(X_{\ell-1}) & = & X_\ell \| b_\ell \end{array}$$

Output $D(b_1 b_2 \ldots b_\ell)$

Claim: $\left|\Pr_s[D(G_{poly}(s)) = 1] - \Pr_s[H_0 = 1]\right| = 0.$

Proof: Input of D is identically distributed in both cases. □

Proof that G_{poly} is pseudorandom

Step 1: modify H_0 one line at a time.

$$\begin{array}{rcl} \multicolumn{3}{c}{\underline{\text{Experiment } H_0}} \\ s & = & X_0 \\ G(X_0) & = & X_1 \| b_1 \\ G(X_1) & = & X_2 \| b_2 \\ & \vdots & \\ G(X_{\ell-1}) & = & X_\ell \| b_\ell \end{array}$$

Output $D(b_1 b_2 \dots b_\ell)$.

$$\begin{array}{rcl} \multicolumn{3}{c}{\underline{\text{Experiment } H_1}} \\ s & = & X_0 \\ X_1 \| b_1 & = & s_1 \| u_1 \\ G(s_1) & = & X_2 \| b_2 \\ & \vdots & \\ G(X_{\ell-1}) & = & X_\ell \| b_\ell \end{array}$$

Proof that G_{poly} is pseudorandom (contd.)

Step 2: Hybrid Lemma

- For contradiction, suppose that G_{poly} is not a PRG, i.e., H_0 and H_ℓ are distinguishable with non-negligible probability $\frac{1}{p(n)}$
- By Hybrid Lemma, there exists i s.t. H_i and H_{i+1} are distinguishable with probability $\frac{1}{p(n)\ell(n)}$
- Idea: Contradict the security of G

Proof that G_{poly} is pseudorandom (contd.)

Step 3: Breaking security of G

- For simplicity, suppose that $i = 0$ (proof works for any i)
- Construct D to break the pseudorandomness of G as follows
  - D gets as input $Z \| r$ sampled either as $X_1 \| b_1$ or as $s_1 \| u_1$
  - Compute $X_2 \| b_2 = G(Z)$ and continue as the rest of the experiment(s)
  - Output $D(rb_2 \dots b_\ell)$
- If $Z \| r$ is pseudorandom, i.e., sampled as $X_1 \| b_1 = G(s)$, then output of D is distributed identically to the output of H_0
- Otherwise, i.e., $Z \| r$ is (truly) random, and therefore output of D is is distributed identically to the output of H_1
- Hence: D distinguishes the output of G with advantage $\frac{1}{p(n)\ell(n)}$ and runs in polynomial time. This is a contradiction $\square$

Concluding Remarks on PRG

- So far we relied on OW *Permutations*. What about OWF?
- OWF $\implies$ PRG: [Impagliazzo-Levin-Luby-89] and [Hastad-90]
  - Celebrated result! Good to read.
- More Efficient Constructions: [Vadhan-Zheng-12]
- Computational analogues of Entropy
- Non-cryptographic PRGs and Derandomization: [Nisan-Wigderson-88]

Functions vs Generators

- PRGs convert **one** short random string s into **one** long pseudorandom string.
  - s is called the seed of the PRG.
- Can we instead get many pseudorandom strings from a single seed?
- Think of a random *function* which maps inputs to outputs as usual.

Pseudorandom Functions (PRF): Next class!