Lecture 3: Constructing Pseudorandom Generators

MIT - 6.5620
Lecture 3 (September 16, 2026)

> ***Warning:*** This document is a rough draft and may contain bugs. Please feel free to email me with corrections.

Recap

- Computationally secure encryption.
- Computational Indistinguishability.

Definition 1. An encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$, associated with key space $\mathcal{K} = \{\mathcal{K}_\lambda\}$, message space $\mathcal{M} = \{\mathcal{M}_\lambda\}$ and ciphertext space $\mathcal{C} = \{\mathcal{C}_\lambda\}$, is computationally secure if for every polynomial size adversary $\mathcal{A}$ there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and every $m_0, m_1 \in \mathcal{M}_\lambda$

$$\Pr_{k \leftarrow \mathcal{K}_\lambda, b \leftarrow \{0,1\}}[\mathcal{A}[\mathsf{Enc}(k, m_b) = b] = 1/2 + \mu(\lambda).$$

An equivalent formulation is using the following notion of computational indistinguishability.

Definition 2. Two distribution ensembles $\{\mathcal{D}_{0,\lambda}\}_{\lambda \in \mathbb{N}}$ and $\{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ are said to be computationally indistinguishable, denoted by

$$\{\mathcal{D}_{0,\lambda}\}_{\lambda \in \mathbb{N}} \approx \{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}},$$

if for every poly-size adversary $\mathcal{A}$ there exists a negligible function $\mu : \mathbb{N} \to [0,1]$ such that for every $\lambda \in \mathbb{N}$

$$\Pr_{x_b \leftarrow \mathcal{D}_{b,\lambda}}[\mathcal{A}(x_b) = b] \leq 1/2 + \mu(\lambda).$$

Equivalently, for every poly-size adversary $\mathcal{A}$ there exists a negligible function $\mu : \mathbb{N} \to [0,1]$ such that for every $\lambda \in \mathbb{N}$

$$|\Pr_{x_0 \leftarrow \mathcal{D}_{0,\lambda}}[\mathcal{A}(x_0) = 1] - \Pr_{x_1 \leftarrow \mathcal{D}_{1,\lambda}}[\mathcal{A}(x_1) = 1]| \leq \mu(\lambda)$$

Using this terminology, an equivalent way of stating Definition 1 is that for every sequence of messages $\{m_{0,\lambda}\}$ and $\{m_{1,\lambda}\}$, where $m_{0,\lambda}, m_{1,\lambda} \in \mathcal{M}_\lambda$, it holds that

$$\{\mathsf{Enc}(k, m_{0,\lambda})\} \approx \{\mathsf{Enc}(k, m_{1,\lambda})\}$$

In the last lecture we defined a cryptographic tool that allows us to construct a computationally secure encryption scheme with keys $k \in \{0,1\}^\lambda$ of size λ and messages $m \in \{0,1\}^n$ of large size $n = \text{poly}(\lambda)$: a PRG. We showed two definitions of a PRG and argued they are equivalent. We now provide the full proof.

Equivalence of PRG Definitions

We prove Theorem 7 from the previous lecture. The formal definition of a PRG and next-bit unpredictability appear there (i.e., in the lecture notes of lecture 2) as well, in Definitions 5 and 6, respectively.

Theorem 3. *An expanding function $G : \{0,1\}^* \to \{0,1\}^*$, with expansion $n = n(\lambda)$, is pseudorandom if and only if it is next-bit unpredictable.*

Proof. The fact that pseudorandomness implies next-bit unpredictability is trivial. Namely, breaking next-bit unpredictability implies a break to the pseudorandomness: Simply try to predict the next bit, if predicted correctly, then guess pseudorandom, and otherwise guess random.

We will focus on the other direction, which follows from a hybrid argument. Hybrid arguments are used a lot in cryptography! Let $G : \{0,1\}^\lambda \to \{0,1\}^{n(\lambda)}$ be an expanding function that is next-bit unpredictable. Suppose for contradiction that it is not pseudorandom. Namely, there exists a poly-size adversary $\mathcal{A}$ and a non-negligible function $\epsilon = \epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$

$$|\Pr[\mathcal{A}(G(U_\lambda)) = 1] - \Pr[\mathcal{A}(U_n) = 1]| > \epsilon(\lambda)$$

In what follows, denote by H_i the distribution where the first i bits are distributed according to $G(U_\lambda)$ and the rest are distributed uniformly at random. The equation above implies that for every $\lambda \in \mathbb{N}$

$$|\Pr[\mathcal{A}(H_n) = 1] - \Pr[\mathcal{A}(H_0) = 1]| > \epsilon(\lambda)$$

Note that for every $\lambda \in \mathbb{N}$

$$\begin{aligned}
&|\Pr[\mathcal{A}(H_n) = 1] - \Pr[\mathcal{A}(H_0) = 1]| = \\
&\left|\sum_{i\in[n]} (\Pr[\mathcal{A}(H_i) = 1] - \Pr[\mathcal{A}(H_{i-1}) = 1])\right| \leq \\
&\sum_{i\in[n]} |(\Pr[\mathcal{A}(H_i) = 1] - \Pr[\mathcal{A}(H_{i-1}) = 1])|
\end{aligned}$$

where the first equation follows from the fact that the sum is telescopic, and the second equation follows from the triangle inequality. This implies that for every $\lambda \in \mathbb{N}$ there exists $i = i(\lambda) \in [n]$ such that

$$|\Pr[\mathcal{A}(H_i) = 1] - \Pr[\mathcal{A}(H_{i-1}) = 1]| > \epsilon(\lambda)/n.$$

We now use $\mathcal{A}$ to construct a predictor $\mathcal{B}$ for the i-th bit of $G(U_\lambda)$. By replacing $\mathcal{A}$ with its complement if necessary, we may assume that

$$\Pr[\mathcal{A}(H_i) = 1] - \Pr[\mathcal{A}(H_{i-1}) = 1] > \frac{\epsilon(\lambda)}{n}.$$

Given the first $i-1$ bits $z = G(s)_{[i-1]}$, the predictor $\mathcal{B}$ chooses a uniformly random bit $b \leftarrow \{0,1\}$ and a uniformly random string $r \leftarrow U_{n-i}$, and computes

$$a = \mathcal{A}(z \circ b \circ r).$$

If $a = 1$, then $\mathcal{B}$ outputs b, and otherwise it outputs $1 - b$.

To analyze $\mathcal{B}$, let

$$p_i = \Pr[\mathcal{A}(H_i) = 1] \qquad \text{and} \qquad p_{i-1} = \Pr[\mathcal{A}(H_{i-1}) = 1].$$

When b is the correct i-th bit, the input to $\mathcal{A}$ is distributed according to H_i. If p_{wrong} denotes the probability that $\mathcal{A}$ outputs 1 when b is the wrong i-th bit, then

$$p_{i-1} = \frac{p_i + p_{\text{wrong}}}{2}.$$

Therefore,

$$\begin{aligned}\Pr[\mathcal{B}(G(U_\lambda)_{[i-1]}) = G(U_\lambda)_i] &= \frac{1}{2}p_i + \frac{1}{2}(1 - p_{\text{wrong}}) \\ &= \frac{1}{2} + p_i - p_{i-1} \\ &> \frac{1}{2} + \frac{\epsilon(\lambda)}{n}.\end{aligned}$$

Since $n = n(\lambda) \leq \text{poly}(\lambda)$, the function $\epsilon(\lambda)/n$ is non-negligible, contradicting the next-bit unpredictability of G. □

We next show how to construct a PRG with a single bit stretch.

Two PRG *Constructions*

Recall the definition of a one-way function that we saw in the previous lecture (Definition 4 there).

Theorem 4. *[1] Pseudorandom generators exist assuming the existence of one-way functions.*

This theorem has a beautiful but very complicated proof. We will prove a simplified version that assumes the existence of one-way permutations, defined below.

Definition 5. A function $f : \{0,1\}^* \to \{0,1\}^*$ is a permutation if it is length preserving and bijective; namely, for every $\lambda \in \mathbb{N}$ and for every $x \in \{0,1\}^\lambda$ it holds that $f(x) \in \{0,1\}^\lambda$, and for every $y \in \{0,1\}^\lambda$ there is a unique $x \in \{0,1\}^\lambda$ such that $f(x) = y$.

Definition 6. A one-way permutation (OWP) $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function that is also a permutation.

We prove the following theorem.

Theorem 7. *Pseudorandom generators exist assuming the existence of a* OWP.

We will see two proofs for this. The first construction, which is also easier, would have a 1-bit stretch (namely, the PRG would map λ bits to $(\lambda + 1)$ bits). Later, in Theorem 9, we will see that this suffices for *any* polynomial stretch. The second construction, which is slightly more involved, would already achieve any desired polynomial stretch.

Both constructions assume the existence of one-way permutations, and use the notion of a hard-core predicate:

Definition 8. $P : \{0,1\}^* \to \{0,1\}$ is a *hardcore predicate* of $f : \{0,1\}^* \to \{0,1\}^*$ if it is efficiently computable and for every poly-size $\mathcal{A}$ there exists a negligible function $\mu : \mathbb{N} \to [0,1]$ such that for every $\lambda \in \mathbb{N}$,

$$\Pr[\mathcal{A}(f(U_\lambda)) = P(U_\lambda)] \leq \frac{1}{2} + \mu(\lambda).$$

Note that we did not yet establish the existence of hard-core predicates: we will do that next week. For the two proofs below, we assume that the one-way permutation is equipped with a hard-core predicate. We will establish the required existence result next week.

Proof 1: one-bit stretch

Given a one-way permutation f with a hardcore predicate P, let $G : \{0,1\}^\lambda \to \{0,1\}^{\lambda+1}$ be defined as:

$$G(U_\lambda) = f(U_\lambda) \circ P(U_\lambda),$$

where "$\circ$" denotes concatenation. We argue that G is a pseudorandom generator.

First, G is expanding by definition, and the fact that it is efficiently computable follows from the fact that both f and P are efficiently computable. Thus, it remains to prove that G is pseudorandom, or equivalently that G satisfies the next-bit unpredictability property.

To this end fix any poly-size adversary $\mathcal{A}$. We need to prove that there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and every $i \in [\lambda + 1]$

$$\Pr[\mathcal{A}(G(U)_{[i-1]}) = G(U)_i] \leq \frac{1}{2} + \mu(\lambda),$$

where $U = U_\lambda$. Here we use the fact that f is a permutation (and therefore every output occurs with the same probability under a uniform input). Since U is uniform and f is a permutation on $\{0,1\}^\lambda$, we have

$$f(U) \equiv U.$$

Thus, for every $i \in [\lambda]$, the i-th bit of $f(U)$ is uniform and independent of the preceding $i-1$ bits, and therefore

$$\Pr[\mathcal{A}(G(U)_{[i-1]}) = G(U)_i] = \frac{1}{2}.$$

For $i = \lambda + 1$, the fact that P is a hardcore predicate of f implies that there exists a negligible function μ such that

$$\Pr[\mathcal{A}(G(U)_{[\lambda]}) = G(U)_{\lambda+1}] = \Pr[\mathcal{A}(f(U_\lambda)) = P(U_\lambda)] \leq \frac{1}{2} + \mu(\lambda),$$

as desired.

Proof 2: polynomial stretch

Our next goal is to construct a PRG

$$G : \{0,1\}^\lambda \to \{0,1\}^{n(\lambda)}$$

for an arbitrary polynomial expanding function $n : \mathbb{N} \to \mathbb{N}$.

Let $m = n(\lambda)$. Given a one-way permutation f with a hard-core predicate P, define

$$G : \{0,1\}^\lambda \to \{0,1\}^m$$

as follows. Starting from a seed $s \in \{0,1\}^\lambda$, compute

$$s,\ f(s),\ f^2(s), \ldots, f^{m-1}(s),$$

and output the hard-core bits in reverse order:

$$G(s) = P(f^{m-1}(s)) \circ P(f^{m-2}(s)) \circ \cdots \circ P(f(s)) \circ P(s).$$

The reverse order is actually not needed, but it will make the proof of next-bit unpredictability simpler.

The fact that G is efficiently computable follows from the fact that $m = \text{poly}(\lambda)$ and both f and P are efficiently computable. Thus, it remains to prove that G is pseudorandom. We do so by proving that it satisfies the next-bit unpredictability property.

Suppose for contradiction that there exists a poly-size adversary $\mathcal{A}$, an index $i = i(\lambda) \in [m]$, and a non-negligible function $\epsilon = \epsilon(\lambda)$ such that

$$\Pr_{s \leftarrow U_\lambda}\left[\mathcal{A}(G(s)_{[i-1]}) = G(s)_i\right] > \frac{1}{2} + \epsilon(\lambda).$$

We construct a poly-size adversary $\mathcal{B}$ that predicts the hard-core predicate $P(x)$ given $f(x)$. On input $f(x)$, $\mathcal{B}$ computes

$$f(x), f^2(x), \ldots, f^{i-1}(x),$$

and outputs

$$\mathcal{A}\left(P(f^{i-1}(x)) \circ P(f^{i-2}(x)) \circ \cdots \circ P(f(x))\right).$$

To see why this works, let

$$x = f^{m-i}(s).$$

Here we use the fact that f is a permutation: if $s \leftarrow U_\lambda$, then $x \leftarrow U_\lambda$ as well. Moreover,

$$G(s)_{[i-1]} = P(f^{i-1}(x)) \circ P(f^{i-2}(x)) \circ \cdots \circ P(f(x)),$$

and

$$G(s)_i = P(x).$$

Therefore,

$$\Pr_{x \leftarrow U_\lambda}\left[\mathcal{B}(f(x)) = P(x)\right] > \frac{1}{2} + \epsilon(\lambda),$$

contradicting the fact that P is a hard-core predicate of f. Thus, G satisfies next-bit unpredictability, and hence is a PRG.

Stretching a PRG

As already hinted above, one can increase the stretch of a PRG by repeatedly applying the PRG to the state part of its output. Namely, given a PRG

$$G : \{0,1\}^\lambda \to \{0,1\}^{\lambda+1},$$

starting from $s_0 = s \in \{0,1\}^\lambda$, for every $i = 0, \ldots, k-1$ write

$$G(s_i) = s_{i+1} \circ b_i,$$

where here as well "$\circ$" denotes concatenation. Here $s_{i+1} \in \{0,1\}^\lambda$ is the new state and $b_i \in \{0,1\}$ is the new output bit.

Thus, the construction can be pictured as

$$s_0 = s \xrightarrow{G} s_1 \circ b_0, \qquad s_1 \xrightarrow{G} s_2 \circ b_1, \qquad \cdots, \qquad s_{k-1} \xrightarrow{G} s_k \circ b_{k-1}.$$

We define

$$G^k : \{0,1\}^\lambda \to \{0,1\}^{\lambda+k}$$

by

$$G^k(s) = s_k \circ b_{k-1} \circ \cdots \circ b_0.$$

Theorem 9. *[1] If G is a* PRG *then for every polynomial $k = k(\lambda)$, G^k is a* PRG.

Proof. Fix a polynomial $k = k(\lambda) \geq 1$. The fact that G^k is stretching and efficiently computable follows from the fact that G satisfies these properties, together with the fact that $k(\lambda) \leq \text{poly}(\lambda)$.

It is tempting to prove that it is pseudorandom by induction on k, as follows:

Base case: $k = 1$. By assumption, $G^1 = G$ is pseudorandom.

Induction step: Suppose $G^{k-1} : \{0,1\}^\lambda \to \{0,1\}^{\lambda+k-1}$ is pseudorandom, and we will prove that $G^k : \{0,1\}^\lambda \to \{0,1\}^{\lambda+k}$ is pseudorandom.

Let

$$G(U_\lambda) = S \circ B,$$

where $S \in \{0,1\}^\lambda$ and $B \in \{0,1\}$. By the definition above,

$$G^k(U_\lambda) = G^{k-1}(S) \circ B.$$

Since

$$G(U_\lambda) \approx U_{\lambda+1},$$

and applying an efficiently computable function preserves computational indistinguishability, we get

$$G^{k-1}(S) \circ B \approx G^{k-1}(U_\lambda) \circ U_1.$$

By the induction hypothesis,

$$G^{k-1}(U_\lambda) \circ U_1 \approx U_{\lambda+k}.$$

Thus, $G^k(U_\lambda) \approx U_{\lambda+k}$.

Why is this argument flawed? Induction only proves the statement for a constant k. It does not prove the statement when $k = k(\lambda)$ grows with the security parameter. The correct way to prove this is via a hybrid argument, as follows.

For every $i = 0, \ldots, k$, define

$$H_i = G^{k-i}(U_\lambda) \circ U_i,$$

where $G^0(U_\lambda) = U_\lambda$. Thus,

$$H_0 = G^k(U_\lambda) \qquad \text{and} \qquad H_k = U_{\lambda+k}.$$

Suppose for contradiction that there exists a polynomial adversary $\mathcal{A}$ and a non-negligible function $\epsilon = \epsilon(\lambda)$ such that

$$|\Pr[\mathcal{A}(H_0) = 1] - \Pr[\mathcal{A}(H_k) = 1]| > \epsilon(\lambda).$$

By the triangle inequality (in class we called this argument "averaging"), there exists $i = i(\lambda) \in [k]$ such that

$$|\Pr[\mathcal{A}(H_{i-1}) = 1] - \Pr[\mathcal{A}(H_i) = 1]| > \frac{\epsilon(\lambda)}{k}.$$

We use this to distinguish $G(U_\lambda)$ from $U_{\lambda+1}$. Given an input

$$s \circ b \in \{0,1\}^{\lambda+1},$$

compute

$$G^{k-i}(s) \circ b \circ U_{i-1}$$

and run $\mathcal{A}$ on the resulting string.

If $s \circ b$ is distributed according to $G(U_\lambda)$, then the input to $\mathcal{A}$ is distributed according to H_{i-1}. If $s \circ b$ is distributed according to $U_{\lambda+1}$, then the input to $\mathcal{A}$ is distributed according to H_i. This is very similar in spirit to the argument we used when proving Theorem 3.

Thus, we distinguish $G(U_\lambda)$ from $U_{\lambda+1}$ with advantage greater than

$$\frac{\epsilon(\lambda)}{k}.$$

Since $k(\lambda) \leq \text{poly}(\lambda)$, this advantage is non-negligible, contradicting the fact that G is a PRG.

□

At the end of the lecture, we took a brief look at stateful encryption and pseudorandom functions (PRFs); we will revisit these topics from the beginning in the next lecture.

References

[1] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.