Course Business

- Midterm is on March 1
  - Allowed to bring one index card (double sided)
- Final Exam is Monday, May 1 (7 PM)
  - Location: Right here

Cryptography
CS 555

Topic 19: One Way Functions, Pseudorandomness

Recap

Last Week+:

- Practical Constructions of Symmetric Key Primitives

Remainder of the Weak:

- Theoretical Foundations for Cryptography

- **Today:**
  - One Way Functions, PRGs, PRFs

One-Way Functions (OWFs)

$$\mathrm{f}(x) = y$$

Definition: A function $\mathrm{f}: \{0,1\}^* \rightarrow \{0,1\}^*$ is one way if it is

1. **(Easy to compute)** There is a polynomial time algorithm (in |x|) for computing f(x).
2. **(Hard to Invert)** Select $\mathrm{x} \leftarrow \{0,1\}^n$ uniformly at random and give the attacker input 1^n, f(x). The probability that a PPT attacker outputs x' such that $\mathrm{f}(x') = f(x)$ is negligible.

One-Way Functions (OWFs)

$$\mathrm{f}(x) = y$$

Remarks:

- **A function that is not one-way is not necessarily always easy to invert (even often)**
- **Any such function can be inverted in time 2^n (brute force)**
- **Length-preserving OWF: $|f(x)| = |x|$**
- **One way permutation: Length-preserving + one-to-one**

One-Way Functions (OWFs)

$$\mathrm{f}(x) = y$$

Remarks:

1. **f(x) does not necessarily hide all information about x.**
2. **If f(x) is one way then so is** $\mathbf{f}'(\mathbf{x}) = \mathbf{f}(\mathbf{x}) \parallel \boldsymbol{LSB}(\boldsymbol{x})$.

One-Way Functions (OWFs)

$$\mathrm{f}(x) = y$$

Remarks:

1. **Actually we usually consider a family of one-way functions**

$$\boldsymbol{f_I}\colon \{\mathbf{0},\mathbf{1}\}^{\boldsymbol{I}} \rightarrow \{\mathbf{0},\mathbf{1}\}^{\boldsymbol{I}}$$

Candidate One-Way Functions

$$f_{ss}(x_1, \ldots, x_n, J) = \left(x_1, \ldots, x_n, \sum_{i \in J} x_i \mod 2^n \right)$$

(Subset Sum Problem is NP-Complete)

Note: $J \subset [n]$ **and** $\mathbf{0} \leq \boldsymbol{x_i} \leq \mathbf{2^n - 1}$

Candidate One-Way Functions

$$f_{ss}(x_1, \dots, x_n, J) = \left(x_1, \dots, x_n, \sum_{i \in J} x_i \mod 2^n \right)$$

(Subset Sum Problem is NP-Complete)

Question: Does $\mathrm{P} \neq NP$ imply this is a OWF?

Answer: No! $\mathrm{P} \neq NP$ only implies that any polynomial-time algorithm fails to solve “some instance” of subset sum. By contrast, we require that PPT attacker fails to solve “almost all instances” of subset sum.

Candidate One-Way Functions (OWFs)

$$f_{p,g}(x) = [g^x \mod p]$$

(Discrete Logarithm Problem)

Hard Core Predicates

- Recall that a one-way function f may potentially reveal lots of information about input

- **Example**: $f(x_1,x_2)=(x_1,g(x_2))$, where g is a one-way function.
- **Claim**: f is one-way (even if $f(x_1,x_2)$ reveals half of the input bits!)

Hard Core Predicates

Definition: A predicate $\mathrm{hc}\colon \{0,1\}^* \to \{0,1\}$ is called a hard-core predicate of a function f if

1. (Easy to Compute) hc can be computed in polynomial time
2. (Hard to Guess) For all PPT attacker A there is a negligible function negl such that we have

$$\mathbf{Pr}_{x\leftarrow\{0,1\}^n}[A(1^n, f(x)) = \mathrm{hc}(x)] \le \frac{1}{2} + negl(n)$$

Attempt 1: Hard-Core Predicate

Consider the predicate

$$\mathrm{hc}(\mathrm{x}) = \bigoplus_{i=1}^{n} x_i$$

Hope: hc is hard core predicate for any OWF.

Counter-example:

$$\mathrm{f(x)} = (\mathrm{g(x)}, \bigoplus_{i=1}^{n} x_i)$$

Trivial Hard-Core Predicate

Consider the function

$$f(x_1,\ldots,x_n) = x_1,\ldots,x_{n-1}$$

f has a trivial hard core predicate

$$\mathrm{hc}(\mathrm{x}) = x_n$$

Not useful for crypto applications (e.g., f is not a OWF)

Attempt 3: Hard-Core Predicate

Consider the predicate

$$\mathrm{hc}(\mathrm{x}, \mathrm{r}) = \bigoplus_{i=1}^{n} x_i r_i$$

(the bits r_1,…, r_n will be selected uniformly at random)

Goldreich-Levin Theorem: (Assume OWFs exist) For any OWF f, hc is a hard-core predicate of g(x,r)=(f(x),r).

Note: The existence of OWFs implies $\mathrm{P} \neq NP$ so we cannot be absolutely certain that they do exist.

Using Hard-Core Predicates

Theorem: Given a one-way-permutation f and a hard-core predicate hc we can construct a PRG G with expansion factor $\ell(n) = n + 1$.

Construction:

$$G(s) = f(s) \parallel \mathrm{hc}(s)$$

Intuition: f(s) is actually uniformly distributed

- s is random
- f(s) is a permutation
- Last bit is hard to predict given f(s) (since hc is hard-core for f)

Arbitrary Expansion

Theorem: Suppose that there is a PRG G with expansion factor $\ell(n) = n + 1$. Then for any polynomial p(.) there is a PRG with expansion factor p(n).

Construction:

- $G(x) = y||b.$ (n+1 bits)
- $G^1(x) = G(y)||b$ (n+2 bits)
- $G^{i+1}(x) = G(y)||b$ where $G^i(x) = y||b$ (n+2 bits)

Any Beyond

Theorem: Suppose that there is a PRG G with expansion factor $\ell(n) = n + 1$. Then for any polynomial p(.) there is a PRG with expansion factor p(n).

Theorem: Suppose that there is a PRG G with expansion factor $\ell(n) = 2n$. Then there is a secure PRF.

Theorem: Suppose that there is a secure PRF then there is a strong pseudorandom permutation.

Any Beyond

Corollary: If one-way functions exist then PRGs, PRFs and strong PRPs all exist.

Corollary: If one-way functions exist then there exist CCA-secure encryption schemes and secure MACs.

PRFs from PRGs

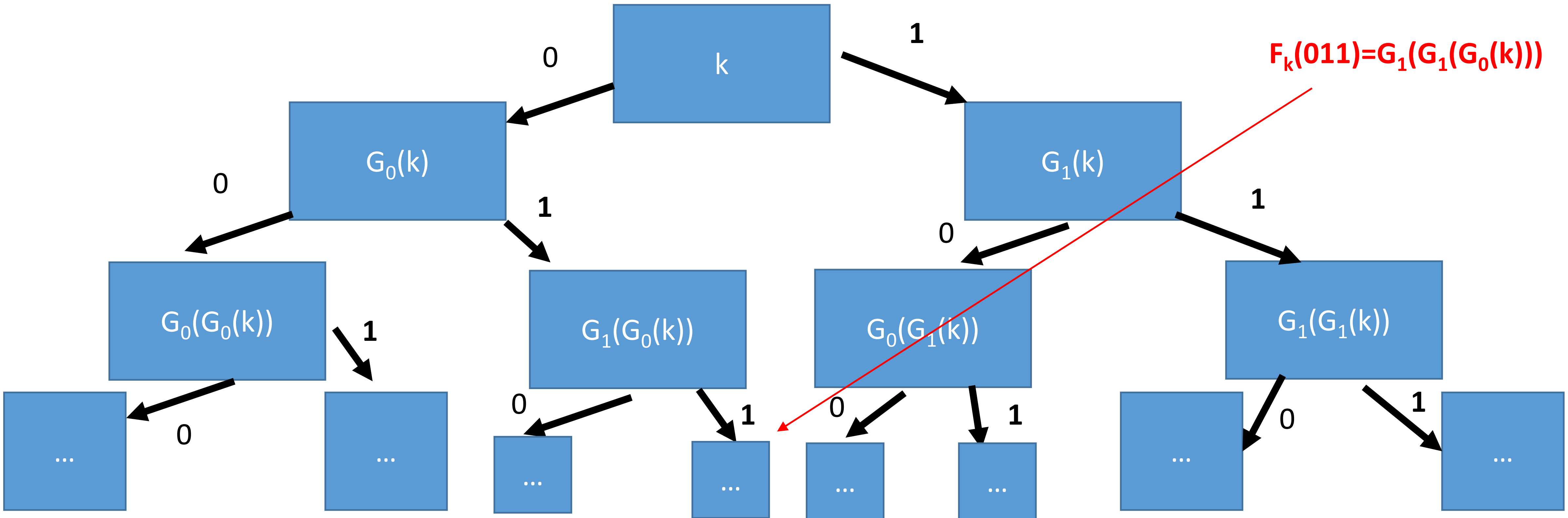
Theorem: Suppose that there is a PRG G with expansion factor $\ell(n) = 2n$. Then there is a secure PRF.

Let $\mathrm{G(x) = G_0(x)||G_1(x)}$ (first/last n bits of output)

$$\boldsymbol{F_K(x_1, \ldots, x_n) = G_{x_n}\left(\ldots\left(G_{x_2}\left(G_{x_1}(K)\right)\right)\ldots\right)}$$

PRFs from PRGs

Theorem: Suppose that there is a PRG G with expansion factor $\ell(n) = 2n$. Then there is a secure PRF.



PRFs from PRGs

Theorem: Suppose that there is a PRG G with expansion factor $\ell(n) = 2n$. Then there is a secure PRF.

Proof:

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right| < \boldsymbol{negl(n)}$$

PRFs from PRGs

Claim 1: For any t(n) and any PPT attacker A we have
$$\left|Pr\left[A\left(r_1 \parallel \cdots \parallel r_{t(n)}\right)\right] - Pr\left[A\left(G(s_1) \parallel \cdots \parallel G\left(s_{t(n)}\right)\right)\right]\right| < negl(n)$$

Proof by Hybrids: Fix j
$$Adv_j = \left|Pr\left[A\left(r_1 \parallel \cdots \parallel r_{j+1} \parallel G\left(s_{j+2}\right) \ldots \parallel G\left(s_{t(n)}\right)\right)\right]\right.$$

PRFs from PRGs

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right| < \boldsymbol{negl(n)}$$

Proof

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right|$$

$$\leq \sum_{j<t(n)} \boldsymbol{Adv_j}$$

$$\leq \boldsymbol{t(n)} \times \boldsymbol{negl(n)} = \boldsymbol{negl(n)}$$

PRFs from PRGs

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right| < \boldsymbol{negl(n)}$$

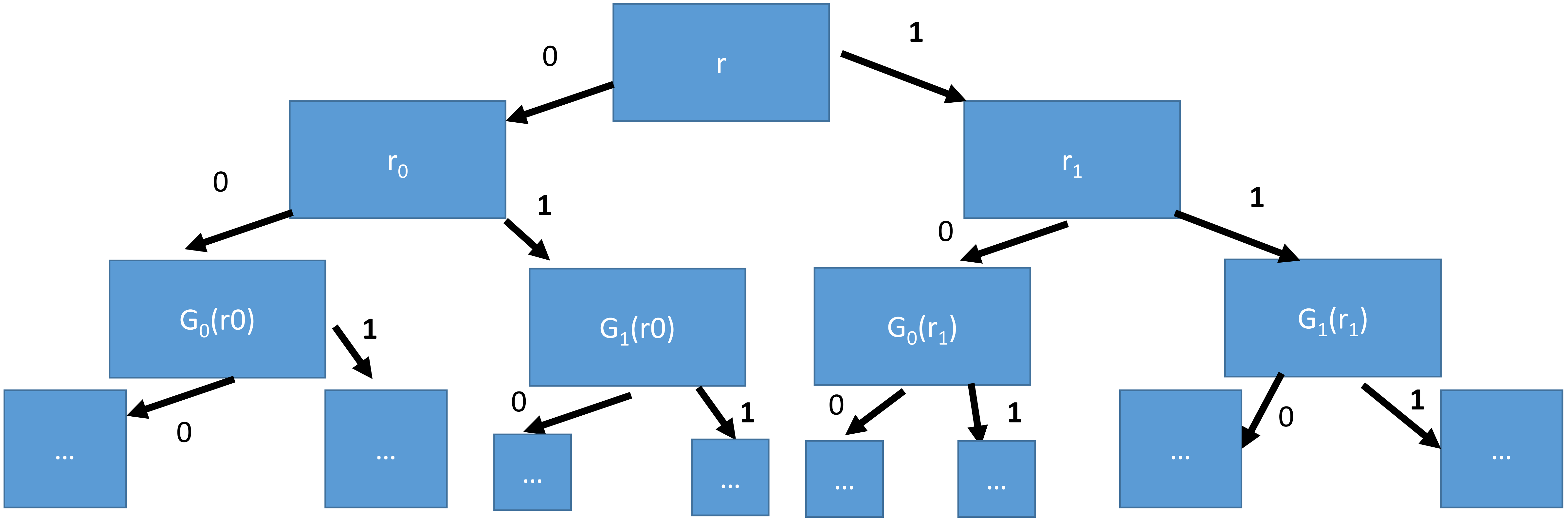
Proof

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right|$$

$$\leq \sum_{\boldsymbol{j<t(n)}} \boldsymbol{Adv_j}$$

$$\leq \boldsymbol{t(n)} \times \boldsymbol{negl(n)} = \boldsymbol{negl(n)}$$

Hybrid H_1



Hybrid H_1 vs H_2

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{n})$$

Claim 2: ***Attacker who makes t(n) queries to F_k (or f) cannot distinguish H_2 from the real game (except with negligible probability).***

Proof: Follows by Claim 1

Hybrid H_2

Claim 1: For any t(n) and any PPT attacker A we have

$$\left|\boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{r_1} \parallel \cdots \parallel \boldsymbol{r_{t(n)}}\right)\right] - \boldsymbol{Pr}\left[\boldsymbol{A}\left(\boldsymbol{G}(\boldsymbol{s_1}) \parallel \cdots \parallel \boldsymbol{G}\left(\boldsymbol{s_{t(n)}}\right)\right)\right]\right| < \boldsymbol{negl}(\boldsymbol{n})$$

Claim 2: *Attacker who makes t(n) queries to F_k (or f) cannot distinguish H_2 from the real game (except with negligible probability).*

Similarly, attacker cannot distinguish H_2 from H_3 etc…

→ Attacker cannot distinguish F_k from f.

Next Class

- Read Katz and Lindell 7.7-7.8
- Theoretical Foundations for Symmetric Key Cryptography
  - Private Key Crypto from OWFs
  - Computational Indistinguishability