Fundamentals of Cryptography: Problem Set 2

Due Wednesday Sep 25 3PM

Collaboration is permitted (and encouraged); however, you must write up your own solutions and acknowledge your collaborators.

If a problem has **0pt**, it will not be graded.

Problem 0 Read Section 3.1, 3.2, 3.3 of "Introduction to Modern Cryptography (2nd ed)" by Katz & Lindell **or** Section 3.1 to 3.5 of "A Graduate Course in Applied Cryptography" by Dan Boneh and Victor Shoup.

You are also recommended to read Section 3.7 of "A Graduate Course in Applied Cryptography", which analyzes a widely-used practical PRG.

Problem 1 (3pt) A integer set $\mathcal{I} \subseteq \mathbb{N}$ is called *polynomial-time-enumerable* if (a) there exists a deterministic polynomial-time algorithm A, such that $A(1^n)$ outputs whether $n \in \mathcal{I}$; (b) Let $s_{\mathcal{I}}(n) := \min\{i : i \in \mathcal{I} \text{ and } i \geq n\}$, then $s_{\mathcal{I}}(n) = \text{poly}(n)$.

For any integer set $\mathcal{I} \subseteq \mathbb{N}$, a function $G : \{0,1\}^* \to \{0,1\}^*$ is a PRG on length in $\mathcal{I}$, if

- G can be computed in polynomial time;
- there exists $\ell : \mathbb{N} \to \mathbb{N}$ such that $\forall n \in \mathbb{N}, \ell(n) > n$, and $|G(x)| = \ell(|x|)$;
- for all PPT distinguisher D, there is a negligible function $\varepsilon(n) = \text{negl}(n)$, such that for all $n \in \mathcal{I}$

$$\left| \Pr_{s \leftarrow \{0,1\}^n}[D(G(s)) = 1] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}[D(r) = 1] \right| \leq \varepsilon(n).$$

Assume G is a PRG on length in a polynomial-time-enumerable set $\mathcal{I}$, construct a PRG based on G. (You may assume w.l.o.g. that $\ell(n) = n + 1$.)

Problem 2 (10pt) Assume G, G_0, G_1 are secure PRGs with same stretch function ℓ. (That is, $|G(s)| = |G_0(s)| = |G_1(s)| = \ell(|s|)$.) In each of the following cases, either prove that G' is a PRG, or show a counterexample.

Part A (0pt) $G'(s) := G(0\|s)$, where $\|$ donates concatenation.

Part B $G'(s) := G(s_{2:n})$, where n denote the bit length of s.

Part C $G'(s) := G(s)\|G(s + 1 \bmod 2^n)$.

Part D (0pt) $G'(s) := G_1(G_2(s))$.

Part E $G'(s) := G_{s_1}(s)$, where s_1 is the first bit of s.

Part F $G'(s) := s_1 \| G_{s_1}(s_{2:n})$, where $s_{2:n}$ denotes all but the first bit of s.

Part G $G'(s) := G_1(s_{1:\lfloor \log n \rfloor}) \| G_2(s_{\lfloor \log n+1 \rfloor:n})$, where $s_{a:b}$ denotes the substring of s that starts from the a-th bit and ends at the b-th bit.

Problem 3 (5pt) An important application of PRG in complexity is derandomization.

Part A (4pt) Assume PRG exists. For any language $\mathcal{L}$ in BPP, show that there exists a p.p.t. Turing machine M such that

- For every $x \in \mathcal{L}$, the probability $M(x)$ outputs 1 is at least 2/3.
- For every $x \notin \mathcal{L}$, the probability $M(x)$ outputs 1 is at most 1/3.
- For every $x \in \{0,1\}^n$, the execution of $M(x)$ consumes at most n random bits.

Part B (1pt) Your construction in Part A is likely to be false, unless the definition of PRG is slightly strengthed. How to strength the definition of PRG so that the construction becomes correct?

Remark: As a consequence, assume the existence of PRG (under the strengthened definition of Part B), any $\mathcal{L} \in \boldsymbol{BPP}$ can be decided by a deterministic $2^n \operatorname{poly}(n)$-time algorithm. Moreover, complexitists are willing to consider stronger assumptions, e.g., existence of PRGs that fool any sub-exponential distinguisher. Such stronger assumption implies that $\mathcal{L}$ can be decide by a deterministic $2^{\operatorname{poly}(\log(n))}$-time algorithm.

Problem 4 (3pt) Yao's Hybrid Argument Assuming G is a secure PRG satisfying $|G(s)| = |s| + 1$, we can construct a PRG G' for any polynomial stretch function $\ell(\lambda) \in \operatorname{poly}(\lambda)$. To simplify the proof, define auxiliary functions G_i as

$$\begin{aligned} G_0 \text{ is } G, \\ G_i(s) = s_1 \| G_{i-1}(s_{2:|s|}) = s_{1:i} \| G(s_{i+1:|s|}). \end{aligned}$$

Then

$$G'(s) := G_{\ell(|s|)-|s|-1}(\dots G_2(G_1(G_0(s))) \dots)$$

is a secure PRG for stretch function ℓ. (We considered a very similar construction in the class.)

Prove that G' is a secure PRG. In particular, for any p.p.t. distinguisher D', show how to construct *a* p.p.t. distinguisher D such that

$$\left| \Pr_{s \leftarrow \{0,1\}^\lambda}[D(G(s))] - \Pr_{r \leftarrow \{0,1\}^{\lambda+1}}[D(r)] \right| \geq \frac{1}{\operatorname{poly}(\lambda)} \left| \Pr_{s \leftarrow \{0,1\}^\lambda}[D'(G'(s))] - \Pr_{r \leftarrow \{0,1\}^{\ell(\lambda)}}[D'(r)] \right|.$$

Problem 5 (0pt): Asymptotic Notations Let $g : \mathbb{N} \to \mathbb{R}_+$ be a function mapping natural numbers to positive real numbers. We define a few function classes. For any $f : \mathbb{N} \to \mathbb{R}$, we say

- $f(n) \in O(g(n))$ if and only if $\exists c \in \mathbb{R}_+, N \in \mathbb{N}$ s.t. $\forall n \geq N, |f(n)| \leq c \cdot g(n)$;
- $f(n) \in o(g(n))$ if and only if $\forall c \in \mathbb{R}_+, \exists N \in \mathbb{N}$ s.t. $\forall n \geq N, |f(n)| \leq c \cdot g(n)$;

As equivalent definitions,

$$f(n) \in O(g(n)) \iff \limsup_{n\to\infty} \frac{|f(n)|}{g(n)} \in [0, +\infty),$$
$$f(n) \in o(g(n)) \iff \limsup_{n\to\infty} \frac{|f(n)|}{g(n)} = 0.$$

For any $f : \mathbb{N} \to \mathbb{R}_+$, we say

- $f(n) \in \Omega(g(n))$ if and only if $\exists c \in \mathbb{R}_+, N \in \mathbb{N}$ s.t. $\forall n \geq N, f(n) \geq c \cdot g(n)$;
- $f(n) \in \omega(g(n))$ if and only if $\forall c \in \mathbb{R}_+, \exists N \in \mathbb{N}$ s.t. $\forall n \geq N, f(n) \geq c \cdot g(n)$;
- $f(n) \in \Theta(g(n))$ if and only if $\exists c_1, c_2 \in \mathbb{R}_+, N \in \mathbb{N}$ s.t. $\forall n \geq N, c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$.

As equivalent definitions,

$$f(n) \in \Omega(g(n)) \iff \liminf_{n\to\infty} \frac{f(n)}{g(n)} \in (0, +\infty],$$
$$f(n) \in \omega(g(n)) \iff \liminf_{n\to\infty} \frac{f(n)}{g(n)} \to +\infty,$$
$$f(n) \in \Theta(g(n)) \iff f(n) \in O(g(n)) \text{ and } f(n) \in \Omega(g(n)).$$

Remark: As a common abuse of notation, people often write "$f(n) = O(g(n))$" instead of "$f(n) \in O(g(n))$". In such case, "=" is not symmetric. Exchanging the two sides gives "$O(g(n)) = f(n)$", which is not a valid statement.

Part A Say $f(n) = n^c$ for a constant $c \in \mathbb{N}$. Prove that $f(n) \in o(2^n)$. As mentioned in the remark, this is typically written as $n^c = o(2^n)$.

Part B Prove or disprove: for any function $f : \mathbb{N} \to \mathbb{R}_+$, if $f \notin O(2^n)$ then $f \in \Omega(2^n)$.

Part C Say $f_1(n) \in O(g(n))$, $f_2(n) \in \Theta(g(n))$, prove or disprove $f_1(n)+f_2(n) \in \Theta(g(n))$.

Part D Prove that: for $f : \mathbb{N} \to \mathbb{R}_+$ and $g : \mathbb{N} \to \mathbb{R}_+$, we have

$$f(n) = O(g(n)) \iff g(n) = \Omega(f(n)),$$
$$f(n) = o(g(n)) \iff g(n) = \omega(f(n)),$$
$$f(n) = \Theta(g(n)) \iff g(n) = \Theta(f(n)).$$

Part E There is an infinite sequence of functions $f_1, f_2, \ldots$ For each $i \in \mathbb{N}$, $f_i \in O(g(n))$. Prove or disprove $\sum_{i=n} f_i(n) = O(n \cdot g(n))$.

Cryptographers also consider the function classes of polynomial functions, denoted by $\text{poly}(n)$, and negligible functions, denoted $\text{negl}(n)$. They are defined as

$$\text{poly}(n) := \bigcup_{i\in\mathbb{N}} O(n^i), \qquad \text{negl}(n) := \bigcap_{i\in\mathbb{N}} o\Big(\frac{1}{n^i}\Big).$$

In other words, $f(n) \in \text{poly}(n)$ if $f(n)$ is smaller than some polynomial, $g(n) \in \text{negl}(n)$ if $g(n)$ is smaller than every inverse-polynomial.
Remark: As a common abuse of notation, people often write "$f(n) = \text{poly}(n)$ and $g(n) = \text{negl}(n)$" instead of "$f(n) \in \text{poly}(n)$ and $g(n) \in \text{negl}(n)$".

Part F For any $f : \mathbb{N} \to \mathbb{R}_+$, $g : \mathbb{N} \to \mathbb{R}_+$ such that $f(n) \in \text{poly}(n)$ and $g(n) \in \text{negl}(n)$, prove that:

$$f(n) \cdot g(n) \in \text{negl}(n), \qquad \log(1 + f(n)) \in O(\log n),$$
$$\frac{1}{f(n)} \notin \text{negl}(n), \qquad -\log(g(n)) \in \omega(\log n).$$