Exercise 2 – Foundations of Cryptography 89-856

Due Date: 7th April 2019

March 24, 2019

Exercise 1: Prove that if an efficiently-computable 1–1 function f has a hard-core predicate, then it is one-way. Why is the 1–1 requirement necessary?

Exercise 2: Assuming the existence of one-way functions, prove that there does *not* exist a function b such that b is a hard-core predicate for every one-way function.

Exercise 3: Let $X = \{X_n\}_{n\in\mathsf{N}}$ and $Y = \{Y_n\}_{n\in\mathsf{N}}$ be computationally indistinguishable probability ensembles.

1. Prove that for any probabilistic polynomial-time algorithm A it holds that $\{A(X_n)\}_{n\in\mathsf{N}}$ and $\{A(Y_n)\}_{n\in\mathsf{N}}$ are computationally indistinguishable.
2. Prove that the above does not hold if A does not run in polynomial-time.

Exercise 4: Let f be a length-preserving one-way function, and let b be a hard-core predicate of f. Prove or refute: $G(x) = (f(x), b(x))$ is a pseudorandom generator.

Exercise 5:

1. Prove that if there exist pseudorandom generators, then there exist pseudorandom generators that are not 1–1.
2. Prove that if there exist one-way permutations, then there exist pseudorandom generators (with any expansion factor) that are 1–1.

Exercise 6: Prove that the existence of pseudorandom generators with expansion factor $l(n) = 2n$ implies the existence of one-way functions.[1] You may *not* copy the answer from a text (or the Internet), but must prove the theorem by yourselves.

Hint: Define $f(x, y) = G(x)$, where $|x| = |y|$.

Exercise 7: Consider pseudorandom functions with input length $\ell(n)$ and output length $\ell(n)$, and with a function-sampling algorithm I that uses at most $r_I(n)$ random coins when invoked upon input 1^n:

1. Prove that if there exist pseudorandom functions such that $2^{\ell(n)} \cdot \ell(n) > r_I(n)$, then there exist pseudorandom generators for any polynomial expansion factor $l(n)$.
2. Present a construction of pseudorandom functions where $2^{\ell(n)} \cdot \ell(n) \leq r_I(n)$, without relying on any assumptions.

[1] We will see in class that the assumption is equivalent to the existence of any pseudorandom generator.

Exercise 8: Prove that if there exist pseudorandom functions F_n that map $k(n)$ bits to one bit, then there exist pseudorandom functions that map $k(n)$ bits to n bits. Note: n denotes the security parameter, and there is no restriction on $k(\cdot)$ (in particular, k may be a constant function, or it may be poly(n)). (Hint: use a hybrid argument.)