Writeup for Thursday, Oct 3

Sadhika Dhanasekar

October 3, 2024

1 Recap on One Way Function

A One Way Function (OWF) is easy to compute and hard to invert: $\forall$ NUPPT$A, \exists$ negl $\epsilon(\cdot), \forall\, n \in N, Pr[A(y) \leq f^{-1}(y) * y * f(x)] < \epsilon(n)$ for all $x \leftarrow \{\}^n$

2 OWF $\Rightarrow$ PRG

- f(x) is rand if x is
- given f(x) hard to find x

Permutation: one-to-one/onto $\rightarrow$ every input in the domain has a unique output in the range and vice versa.
OWF f is a one-way permutation is f is permutation. In addition, one way permutation is PRG but not with expansion. The last bit of input x is hard to find and thus contributes to the pseudo-randomness.
$Pr[a = f(x)] - Pr[a = x]$

3 Hard Core Predicate

$f : \{\}^n \rightarrow \{\}^n$ function, $h : \{\}^n\{0, 1\}$ is Hard Core Predicate (HCP) with respect to f if $\forall\ NUPPTA, \exists\ negl\epsilon, \forall\ n, Pr[A(a^n, f(x) = h(x)] \leq \frac{1}{2} + \epsilon(n)$ for $x \leftarrow \{\}^n$
Suppose f is a negligible permutation with given a n-bit input, x, we get an (n+1) bit input y that is the PRG when combined with the HCP. Using hard-core predicate h for the function f, we get one extra bit that is hard to predict from y and exhibits a close-to uniformly random distribution.

Theorem: $\exists f, h$ such that f is OWP and h is HCP with respect to f, $\Rightarrow g(x) := f(x)||h(x)$ is PRG. Proof by picture: Assume by contraction that this is not true. There exists an adversary A that takes in a (n+1) bit string that can tell you whether the input, t is from $g(x)$ or a uniform distribution. Now, say that the first n bits of t is from $y = f(x)$ and the last bit is sampled from a uniform distribution of 0,1 to create t. The work the adversary is doing is a reduction of B against h. If A says t is from $g(x)$, it outputs b; if it says t is a uniformly random string, then it outputs $\neg b$. This output will just equal the HCP with high probability.

Ex. $f(x) := GPT(x), h(x) :=$ parity$(y(x))$? This is unknown. $f(x, z) = GPT(x)$ where x is an n bit string and z is a 1 bit string, where $h(x, z) := z$.

Construction of Hard Core Predicate: Suppose f is OWP, then $f'(x, r) := f(x)||r$; $x, r \leftarrow \{0,1\}^n$; $h(x, r) := x \odot r = \sum x_i * r_i$ mod 2 where $\odot$ is the inner product. Theorem: f' is OWP , h is HCP with respect to f'. Goldreich and Levin in 1987 constructed a similar formula for OWF instead of OWP.

- f' is permutation $(f')(a, b) := f^{-1}(n), a$
- f' is OWF, easy to compute

Similar to proof from the previous section using reduction B, whatever A outputs, we just take the first n bits to be x.

Given input n with some hard bits in the middle, after applying f, can we determine the hard bits in this new n. The question is determining many hard bits do we need. $Z \rightarrow Z$ given $log(n)$ bits. Let's sample another input string, r, that is a uniformly random string with random bits of 1 that we are sampling out of the input string with is the intuition behind the HCP construction.

Proof: Assume h is not HCP. $\exists$ A, poly p for infinitely many n, $Pr[A(f'(x, r)) = h(x), r)] \geq \frac{1}{2} + \frac{1}{p(n)}$ where x and r are uniformly and randomly sampled. We want an Algorithm, B, that inverts f.

Warm up: $Pr[A(f(x)||r) = h...)] = 1$ where x and r are uniformly and randomly sampled. This can be simplified to: $\forall$ $x, r, A(f(x)||r) = x \odot r$ as we only care about the random bit, r. If we have a good A, then given the input 1 followed by n number of 0s, $A(f(x)||1000....0) = x \odot 100.000 = x_1$. The concatenation is referred to as $e_1 where e_i := 000....10.....$ where there is a i number of 0s on either end. Thus, for all i, $A = (y||e_i) = x_i$

Warm up 2: $Pr[....] \geq \frac{3}{4} + \frac{1}{p(n)}$ For B(y),

1. for all $i = 1....n$, $z_i \leftarrow A(y||e_i) \oplus A(y||r \oplus e_i)$ which is similar to $(x \odot r) \oplus (x \odot (r \oplus e_i)$ which equals $x \odot) e_i = x_i$
2. will continue next class
3. output: $z_1...z_n$