CS 6810: Theory of Computing **Fall 2023**

Lecture 21: Nov 2, 2023

Lecturer: Eshan Chattopadhyay *Scribe: Jiho Cha*

1 Hard Function Implies PRG

Claim 1.1. *Suppose $f : \{0,1\}^n \to \{0,1\}$ is (S,ε)-hard. Then $G : \{0,1\}^n \to \{0,1\}^{n+1}$ defined as $G(x) := (x, f(x))$ is a (S', ε')-PRG, where $S' = S - 1$ and $\varepsilon' = \varepsilon$.*

Remark 1.2. *One might question why we introduce a new variable ε' in the claim. This claim is a particular case of a more general theorem where ε' need not equal ε.*

Proof of 1.1: We proceed by contradiction, that is, we assume there exists a distinguisher circuit D of size $\leq S'$ such that

$$|\Pr[D(x, f(x)) = 1] - \Pr[D(x, b) = 1]| > \varepsilon'$$

where x is uniformly sampled from U_n and b is a random bit. Observe that by using law of total probability (and conditioning over whether or not $f(x) = b$ or $f(x) = \overline{b}$), the preceding condition is equivalent to

$$|\Pr[D(x, f(x)) = 1] - \Pr[D(x, \overline{f(x)}) = 1]| > 2\varepsilon'$$

Next, consider the following randomized algorithm A with oracle access to D for computing f:

Algorithm 1 A

Input: x
1: Flip a fair coin b
2: **if** $D(x, b) = 1$ **then**
3: Output b
4: **else**
5: Output $\overline{b}$
6: **end if**

We are interested in the probability that $A(x) = f(x)$. This event occurs either when $D(x, b) = 1 \wedge b = f(x)$ or when $D(x, b) = 0 \wedge b = \overline{f(x)}$. Since b is equally likely to be $f(x)$ or $\overline{f(x)}$, we see that:

$$\begin{aligned}
\Pr_{x \sim U_n}[A(x) = f(x)] &= \frac{1}{2}\Big[\Pr[D(x, f(x)) = 1] + \Pr[D(x, \overline{f(x)}) = 0]\Big] \\
&= \frac{1}{2}\Big[\Pr[D(x, f(x)) = 1] + (1 - \Pr[D(x, \overline{f(x)}) = 1])\Big] \\
&\geq \frac{1}{2} + \varepsilon'.
\end{aligned}$$

We now attempt to derandomize A by considering variants A_1 and A_0, which are identical to A, except we manually set b to 1 and 0, respectively. By the averaging principle, either $\Pr[A_1(x) = f(x)] \geq \frac{1}{2} + \varepsilon'$ or $\Pr[A_0(x) = f(x)] \geq \frac{1}{2} + \varepsilon'$ (if both inequalities were false, then there is no way the probability we derived above holds). We then give b as advice to A, where b is a bit such that the algorithm $A_b(x)$ has non-negligible advantage at computing $f(x)$. It is important to note is that b is independent from x, so our advice stays constant despite x. Observe that A_1 is directly computed by $D(x,1)$, and A_0 is directly computed by $\overline{D(x,0)}$, which can be computed by attaching a not gate to the output of $D(x,0)$. So we need a circuit of size at most $S'+1=S$ to have at least an $\varepsilon'=\varepsilon$ advantage in computing f, which breaks the assumption that f is (S,ε)-hard.

2 Towards a Better PRG

We will now perform a similar technique to construct a "better" PRG (one with longer stretch). This construction comes courtesy of Nisan and Wigderson.

Theorem 2.1. *Suppose $f:\{0,1\}^n\to\{0,1\}$ is (S,ε)-hard and we have an (n,k)-design over a universe $[r]=\{0,1,...,r\}$ (defined below). Then there exists a (S',ε')-PRG $G:\{0,1\}^r\to\{0,1\}^m$, where $S'=S-O(2^km)$ and $\varepsilon'=m\cdot\varepsilon$.*

Definition 2.2. *An (n,k)**-design** over a universe $[r]$ is a collection of sets $S_1,...,S_m\subseteq[r]$, where $\forall i\in[m],\#S_i=n$ and $\forall i\neq j,\#(S_i\cap S_j)\leq k$. All of these sets can be assumed to have the elements arranged in ascending order. (Note this is an equivalent notion to a n-uniform undirected hypergraph with m hyperedges with nodes labelled by $[r]$, such that the intersection of any two distinct hyperedges has cardinality at most k).*

Remark 2.3. *We can think of r as being linear in n, and m as being exponential in n, which suggests that G has a very large stretch.*

Proof of Theorem 2.1: Consider the following function G: upon input $z=b_1\circ b_2\circ...\circ b_r$: The first bit of its output will be $f(b_{\ell_1}\circ b_{\ell_2}\circ...\circ b_{\ell_n})$, where $\{\ell_1,...,\ell_n\}=S_1$. The ith bit of its output will be obtained by the analogous procedure on S_i. For notational convenience, given a set $S_i=\{i_1,i_2,...,i_n\}$, we write $z|_{S_i}:=b_{i_1}\circ...\circ b_{i_n}$. So we can equivalently define $G:=f(z|_{S_1})\circ f(z|_{S_2})\circ...\circ f(z|_{S_m})$. We will show that G is our desired PRG via contradiction and hybrid argument. Suppose there exists a distinguisher circuit D of size $\leq S'$ such that

$$\Big|\Pr_{z\sim\{0,1\}^r}[D(G(z))=1]-\Pr_{x\sim\{0,1\}^m}[D(x)=1]\Big|>\varepsilon'.$$

Now, we consider a series of strings (aka "hybrids") $H_0,H_1,...H_m$, where $H_0=f(z|_{S_1})\circ f(z|_{S_2})\circ...\circ f(z|_{S_m})$ and $H_m=b_1\circ b_2\circ...\circ b_m$. In general, H_i is an m bit string where the first i bits are sampled uniformly at random, and for all $i<j\leq m$, the jth bit of H_i is $f(z|_{S_j})$.

Observe that

$$\Big|\sum_{i=0}^{m-1}\Pr[D(H_i)=1]-\Pr[D(H_{i+1})=1]\Big|>\varepsilon'.$$

This can be verified by noting that the LHS is a telescoping sum where the only terms that survive are $\Pr[D(H_0)=1]-\Pr[D(H_m)=1]$, which is just a reformulation of our original assumption. By Triangle Inequality, we have that

$$\sum_{i=0}^{m-1}\Big|\Pr[D(H_i)=1]-\Pr[D(H_{i+1})=1]\Big|>\varepsilon'.$$

By a simple argument by contradiction, we see this implies that there exists an i such that $\Big|\Pr[D(H_i)=1]-\Pr[D(H_{i+1})=1]\Big|>\frac{\varepsilon'}{m}$.

We now design a randomized algorithm B with oracle access to D for computing f. We start by designing the following algorithm B'. Note that in addition to input string x, it is given a bit b', which is either the output of $f(x)$ or a random bit. In the second line, we sample a set of $r - n$ bits to occupy the bits of z that are not indexed by S_{i+1}. The third line inserts or "concatenates" (please excuse the gross abuse of notation) the bits of x into the n bits of z indexed by S_{i+1}:

Algorithm 2 B'

Input: x, b'

1: Sample random bits $b_1, ..., b_i$
2: Sample $z|_{\overline{S_{i+1}}}$
3: Set $z = x$ " $\circ$ " $z|_{\overline{S_{i+1}}}$
4: Set H $= b_1 \circ ... \circ b_i \circ b' \circ f(z|_{S_{i+2}}) \circ ... \circ f(z|_{S_m})$
5: Output $D(H)$

Observe that if b' is $f(x)$, then H is distributed like H_i, and if b' is a random bit, then H is distributed like H_{i+1}. Thus, B' has the property that

$$\Pr[B'(x, f(x)) = 1] - \Pr[B'(x, b) = 1] \geq \frac{\varepsilon'}{m}.$$

(Note that we can get rid of the absolute value without loss of generality, because we could always flip the output of our distinguisher). In particular, because a random setting of $b_1, ...b_i, z|_{\overline{S_{i+1}}}$ has this advantage, by the averaging principle, there must be a specific setting of these bits that also achieves this advantage. We can give this setting as advice to our algorithm B', and again, it is important to note that this setting is independent from x. This immediately gives us an algorithm B that can compute $f(x)$ with non-negligible advantage - we can use a similar formulation as in Section 1, using our algorithm B' as the distinguisher.

It remains to compute the size of a circuit for B (i.e. computing f). Naively, we would need at most m edges to feed in $b_1, ...b_i, b', f(z|_{S_{i+2}}), ..., f(z|_{S_m})$ into D, but we run into trouble when we wish to compute $f(z|_{S_{i+2}}), ..., f(z|_{S_m})$. At a cursory glance, it appears as though we need to use many circuits that compute f in order to compute f. However, we are saved by the fact that at most there are k bits of overlap between S_i and any other set S_j (these sets are part of a design). Because we are given $z|_{\overline{S_{i+1}}}$, for any set S_j, at most k of its bits are not fixed (those that are in the intersection of S_j and S_i). Thus, we just need a circuit of size 2^k to compute $f(z|_{S_j})$ for any $j \neq i$. In reality, we just need $O(m2^k)$ edges to feed in the input to our distinguisher circuit, which itself uses S' edges. This breaks the assumption that f is (S, ε)-hard.

In the next class, we show how to construct our designs and how to set our parameters to show that BPP=P under reasonable assumptions on circuit lower bounds.