Notes for Lecture 26

These notes follow quite literally the notes that I wrote for the PCMI school on computational complexity in Summer 2000. There is some redundancy between these notes and the content of past lectures.

In this lecture we prove the following result.

Theorem 1 (Special case of [NW94]) *Suppose there is $L \in \mathbf{E}$ and $\delta > 0$ such that, for all sufficiently large n, $H(L_n) \geq 2^{\delta n}$; then there is a family of generators $G_m : \{0,1\}^{O(\log m)} \rightarrow \{0,1\}^m$ that are computable in poly(m) time and that are $(2m, 1/8)$-pseudorandom (in particular, $\mathbf{P} = \mathbf{BPP}$).*

1 The Nisan-Wigderson Construction

The Nisan-Wigderson generator is based on the existence of a decision problem L in $\mathbf{E}$ such that for almost every input length l we have $H(L_l) \geq 2^{\delta l}$, yet there is a uniform algorithm that solves L_l in $2^{O(l)}$ time. Our goal is to use these assumptions on L_l to build a generator whose input seed is of length $O(l)$, whose output is of length $2^{\Theta(l)}$ and indistinguishable from uniform by adversaries of size $2^{\Theta(l)}$, and the generator should be computable in time $2^{O(l)}$.

As we will see in a moment, it is not too hard to construct a generator that maps l bits into $l+1$ bits, and whose running time and pseudorandomness are as required. We will then present the Nisan-Wigderson construction, and present its analysis.

1.1 Impredictability versus Pseudorandomness

Let $f : \{0,1\}^l \rightarrow \{0,1\}$ be a function such that $H(f) \geq s$, and consider the pseudorandom generator $G : \{0,1\}^l \rightarrow \{0,1\}^{l+1}$ defined as $G(x) = x \cdot f(x)$, where '$\cdot$' is used to denote concatenation. We want to argue that G is a $(s-3, 1/s)$-pseudorandom generator.

The argument works by contradiction, and consists in the proof of the following result.

Lemma 2 *Let $f : \{0,1\}^l \rightarrow \{0,1\}$. Suppose that there is a circuit D of size s such that*

$$|\mathbf{Pr}_x[D(x \cdot f(x)) = 1] - \mathbf{Pr}_{x,b}[D(x \cdot b) = 1]| > \epsilon$$

then there is a circuit A of size $s+3$ such that

$$\mathbf{Pr}_x[A(x) = f(x)] > \frac{1}{2} + \epsilon$$

Proof: First of all, we observe that there is a circuit D' of size at most $s+1$ such that

$$\mathbf{Pr}_z[D'(x \cdot f(x)) = 1] - \mathbf{Pr}_{x,b}[D'(x \cdot b) = 1] > \epsilon \qquad (1)$$

This is because Expression (1) is satisfied either by taking $D = D'$ or by taking $D = \neg D'$. A way to interpret Expression (1) is to observe that when the first l bits of the input of $D'()$ are a random string x, D' is more likely to accept if the last bit is $f(x)$ than if the last bit is random (and, for a stronger reason, if the last bit is $1 - f(x)$). This observation suggests the following strategy in order to use D' to predict f: given an input x, for which we want to compute $f(x)$, we guess a value b, and we compute $D'(x, b)$. If $D'(x, b) = 1$, we take it as evidence that b was a good guess for $f(x)$, and we output b. If $D'(x, b) = 0$, we take it as evidence that b was the wrong guess, and we output $1 - b$. Let A_b be the procedure that we just described. We claim that

$$\mathbf{Pr}_{x,b}[A_b(x) = f(x)] > \frac{1}{2} + \epsilon \tag{2}$$

The claim is proved by the following derivation

$$\begin{aligned}
& \mathbf{Pr}_{x,b}[A_b(x) = f(x)] \\
= \ & \mathbf{Pr}_{x,b}[A_b(x) = f(x) | b = f(x)] \mathbf{Pr}_{x,b}[b = f(x)] \\
& + \mathbf{Pr}_{x,b}[A_b(x) = f(x) | b \neq f(x)] \mathbf{Pr}_{x,b}[b \neq f(x)] \\
= \ & \frac{1}{2} \mathbf{Pr}_{x,b}[A_b(x) = f(x) | b = f(x)] + \frac{1}{2} \mathbf{Pr}_{x,b}[A_b(x) = f(x) | b \neq f(x)] \\
= \ & \frac{1}{2} \mathbf{Pr}_{x,b}[D'(x, b) = 1 | b = f(x)] + \frac{1}{2} \mathbf{Pr}_{x,b}[D'(x, b) = 0 | b \neq f(x)] \\
= \ & \frac{1}{2} + \frac{1}{2} \mathbf{Pr}_{x,b}[D'(x, b) = 1 | b = f(x)] - \frac{1}{2} \mathbf{Pr}_{x,b}[D'(x, b) = 1 | b \neq f(x)] \\
= \ & \frac{1}{2} + \mathbf{Pr}_{x,b}[D'(x, b) = 1 | b = f(x)] \\
& - \frac{1}{2} \left(\mathbf{Pr}_{x,b}[D'(x, b) = 1 | b = f(x)] + \mathbf{Pr}_{x,b}[D'(x, b) = 1 | b \neq f(x)] \right) \\
= \ & \frac{1}{2} + \mathbf{Pr}_{x}[D'(x, f(x)) = 1] - \mathbf{Pr}_{x,b}[D'(x, b) = 1] \\
> \ & \frac{1}{2} + \epsilon
\end{aligned}$$

From Expression (2) we can observe that there must be a $b_0 \in \{0, 1\}$ such that

$$\mathbf{Pr}_{x}[A_{b_0}(x) = f(x)] > \frac{1}{2} + \epsilon$$

And A_{b_0} is computed by a circuit of size at most $s + 3$ because $A_{b_0}(x) = b_0 \oplus (\neg D'(x, b_0))$, which can be implemented with two more gates given a circuit for D'. $\Box$

1.2 Combinatorial Designs

Consider a family $(S_1, \ldots, S_m)$ of subsets of an universe U. We say the family is a (l, α)-design if, for every i, $|S_i| = l$, and, for every $i \neq j$, $|S_i \cap S_j| \leq \alpha$.

Theorem 3 *For every integer l, fraction $\gamma > 0$, there is an $(l, \log m)$ design $(S_1, \ldots, S_m)$ over the universe $[t]$, where $t = O(l/\gamma)$ and $m = 2^{\gamma l}$; such a design can be constructed in $O(2^t t m^2)$ time.*

We will use the following notation: if z is a string in $\{0,1\}^t$ and $S \subset [t]$, then we denote by $z_{|S}$ the string of length $|S|$ obtained from z by selecting the bits indexed by S. For example if $z = (0,0,1,0,1,0)$ and $S = \{1,2,3,5\}$ then $z_{|S} = (0,0,1,1)$.

1.3 The Nisan-Wigderson Generator

For a Boolean function $f : \{0,1\}^l \to \{0,1\}$, and a design $\mathcal{S} = (S_1,\ldots,S_m)$ over $[t]$, the Nisan-Wigderson generator is a function $NW_{f,\mathcal{S}} : \{0,1\}^t \to \{0,1\}^m$ defined as follows:

$$NW_{f,\mathcal{S}}(z) = f(z_{|S_1}) \cdot f(z_{|S_2}) \cdots f(z_{|S_m})$$

2 The Reduction from Distinguishing to Predicting

The following lemma implies Theorem 1.

Lemma 4 *Let $f : \{0,1\}^l \to \{0,1\}$ be a Boolean function and $\mathcal{S} = (S_1,\ldots,S_m)$ be a $(l, \log m)$ design over $[t]$. Suppose $D : \{0,1\}^m \to \{0,1\}$ is such that*

$$|\mathbf{Pr}_r[D(r) = 1] - \mathbf{Pr}_z[D(NW_{f,\mathcal{S}}(z)) = 1]| > \epsilon\ .$$

Then there exists a circuit C of size $O(m^2)$ such that

$$|\mathbf{Pr}_x[D(C(x)) = f(x)] - 1/2| \geq \frac{\epsilon}{m}$$

Proof: The main idea is that if D distinguishes $NW_{f,\mathcal{S}}(\cdot)$ from the uniform distribution, then we can find a bit of the output of the generator where this distinction is noticeable. On such a bit, D is distinguishing $f(x)$ from a random bit, and such a distinguisher can be turned into a predictor for f. In order to find the "right bit", we will use the *hybrid argument*. At this level of abstraction, the analysis is the same as the analysis of the Blum-Micali-Yao generator, however, as the analysis unfolds, we will see major differences.

Let us start with the hybrid argument. We define $m+1$ distributions $H_0,\ldots,H_m$; H_i is defined as follows: sample a string $v = NW_{f,\mathcal{S}}(z)$ for a random z, and then sample a string $r \in \{0,1\}^m$ according to the uniform distribution, then concatenate the first i bits of v with the last $m-i$ bits of r. By definition, H_m is distributed as $NW_{f,\mathcal{S}}(y)$ and H_0 is the uniform distribution over $\{0,1\}^m$.

Using the hypothesis of the Lemma, we know that there is a bit $b_0 \in \{0,1\}$ such that

$$\mathbf{Pr}_y[D'(NW_{f,\mathcal{S}}(y)) = 1] - \mathbf{Pr}_r[D'(r)] > \epsilon$$

where $D'(x) = b_0 \oplus D(x)$.

We then observe that

$$\begin{aligned}
\epsilon &\leq \mathbf{Pr}_z[D'(NW_{f,\mathcal{S}}(z)) = 1] - \mathbf{Pr}_r[D'(r)] \\
&= \mathbf{Pr}[D'(H_m) = 1] - \mathbf{Pr}[D'(H_0) = 1] \\
&= \sum_{i=1}^{m} (\mathbf{Pr}[D'(H_i) = 1] - \mathbf{Pr}[D'(H_{i-1}) = 1])
\end{aligned}$$

In particular, there exists an index i such that

$$\mathbf{Pr}[D'(H_i) = 1] - \mathbf{Pr}[D'(H_{i-1}) = 1] \geq \epsilon/m \tag{3}$$

Now, recall that

$$H_{i-1} = f(z_{|S_1}) \cdots f(z_{|S_{i-1}}) r_i r_{i+1} \cdot r_m$$

and

$$H_i = f(z_{|S_1}) \cdots f(y_{|S_{i-1}}) f(y_{|S_i}) r_{i+1} \cdot r_m \ .$$

We can assume without loss of generality (up to a renaming of the indices) that $S_i = \{1, \dots, l\}$. Then we can see $z \in \{0,1\}^t$ as a pair (x, y) where $x = z_{|S_i} \in \{0,1\}^l$ and $y = z_{|[t]-S_i} \in \{0,1\}^{t-l}$. For every $j < i$ and $z = (x, y)$, let us define $f_j(x, y) = f(z_{|S_j})$: note that $f_j(x, y)$ depends on $|S_i \cap S_j| \leq \log m$ bits of x and on $l - |S_i \cap S_j| \geq l - \log m$ bits of y. With this notation we have

$$\begin{aligned} &\mathbf{Pr}_{x,y,r_{i+1},\dots,r_m}[D'(f_1(x,y), \dots, f_{i-1}(x,y), f(x), \dots, r_m) = 1] \\ -&\mathbf{Pr}_{x,y,r_{i+1},\dots,r_m} D'(f_1(x,y), \dots, f_{i-1}(x,y), r_i, \dots, r_m) = 1] > \epsilon/m \end{aligned}$$

That is, when D' is given a string that contains $f_j(x, y)$ for $j < i$ in the first $i-1$ entries, and random bits in the last $m - i$ entries, then D' is more likely to accept the string if it contains $f(x)$ in the i-th entry than if it contains a random bit in the i-th entry. This is good enough to (almost) get a predictor for f. Consider the following algorithm:

Algorithm A
Input: $x \in \{0,1\}^l$
Pick random $r_i, \dots, r_m \in \{0,1\}$
Pick random $y \in \{0,1\}^{t-l}$
Compute $f_1(x,y), \dots, f_{i-1}(x,y)$
If $D'(f_1(x,y), \dots, f_{i-1}(x,y), r_i, \dots, r_m) = 1$ output r_i
Else output $1 - r_i$

Let us forget for a moment about the fact that the step of computing $f_1(x, y), \dots, f_{i-1}(x, y)$ looks very hard, and let us check that A is good predictor. Let us denote by $A(x, y, r_1, \dots, r_m)$

the output of A on input x and random choices $y, r_1, \ldots, r_m$.

$$
\begin{aligned}
& \mathbf{Pr}_{x,y,r}[A(x,y,r) = f(x)] \\
= \quad & \mathbf{Pr}_{x,y,r}[A(x,y,r) = f(x) | r_i = f(x)] \mathbf{Pr}_{x,r_i}[r_i = f(x)] \\
& + \mathbf{Pr}_{x,y,r}[A(x,y,r) = f(x) | r_i \neq f(x)] \mathbf{Pr}_{x,r_i}[r_i \neq f(x)] \\
= \quad & \frac{1}{2} \mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 1 | f(x) = r_i] \\
& + \frac{1}{2} \mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 0 | f(x) \neq r_i] \\
= \quad & \frac{1}{2} + \frac{1}{2} \big(\mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 1 | f(x) = b] \\
& - \mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 1 | f(x) \neq b] \big) \\
= \quad & \frac{1}{2} + \mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 1 | f(x) = b] \\
& - \frac{1}{2} \big(\mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 1 | f(x) = b] \\
& + \mathbf{Pr}_{x,y,r}[D'(f_1(x,y), \ldots, f_{i-1}(x,y), r_i, \ldots, r_m) = 1 | f(x) \neq b] \big) \\
= \quad & \frac{1}{2} + \mathbf{Pr}[D'(H_i) = 1] - \mathbf{Pr}[D'(H_{i-1}) = 1] \\
\geq \quad & \frac{1}{2} + \frac{\epsilon}{m}
\end{aligned}
$$

So A is good, and it is worthwhile to see whether we can get an efficient implementation. We said we have

$$\mathbf{Pr}_{x,y,r_i,\ldots,r_m}[A(x,y,r) = f(x)] \geq \frac{1}{2} + \frac{\epsilon}{m}$$

so there surely exist fixed values $c_i, \ldots, c_m$ to give to $r_i, \ldots, r_m$, and a fixed value w to give to y such that

$$\mathbf{Pr}_{x,r}[A(x, w, c_i, c_{i+1}, \ldots, c_m) = f(x)] \geq \frac{1}{2} + \frac{\epsilon}{m}$$

At this point we are essentially done. Since w is fixed, now, in order to implement A, we only have to compute $f_j(x,w)$ given x. However, for each j, $f_j(x,w)$ is a function that depends only on $\leq \log m$ bits of x, and so is computable by a circuits of size $O(m)$. Even composing $i-1 < m$ such circuit, we still have that the sequence $f_1(x,w), \ldots, f_{i-1}(x,w), c_i, c_{i+1}, \ldots, c_m$ can be computed, given x, by a circuit C of size $O(m^2)$. Finally, we notice that at this point $A(x,w,c)$ is doing the following: output the xor between c_i and the complement of $D'(C(x))$. Since c_i is fixed, either $A(x,w,c)$ always equals $D(C(x))$, or one is the complement of the other. In either case the Lemma follows. $\Box$

At this point it should not be too hard to derive Theorem 1.

References

[NW94] N. Nisan and A. Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994. Preliminary version in *Proc. of FOCS'88*.