CS 830: Randomness and Complexity 2/3/05

Lecture 6: Next-Bit Unpredictability and Pseudorandomness

instructor: Jin-Yi Cai Scribe: Bess Berg, Michael Kowalczyk

In this lecture, we clarify some facts on expanders and prove the equivalence between next-bit unpredictability and pseudorandomness.

1 Review of Expanders

Recall from previous lectures that given a weak one-way permutation f, we can recursively define a one-way permutation g that uses only $O(n \log n)$ public coins and has at most constant weakness. Furthermore, from this g we may define a strong one-way permutation h. Recall that h is defined over the vertices of an expander graph, where each vertex represents a possible input to g (including the public coins), and we think of h as alternating some polynomial number of times between an application of g and a step in a random walk on the expander. We would like to clarify that h is not exactly a random walk, rather it uses steps from a random walk in between applications of g.

2 Equivalence of Next-Bit Unpredictability and Pseudorandomness

First, we recall our definitions for pseudorandomness and next-bit unpredictability in the context of polynomial time.

Definition 2.1 (Pseudorandom generator) *For $n < \ell(n) \leq poly(n)$, let $g : 2^n \to 2^{\ell(n)}$ be a poly-time computable function. We say g is a pseudorandom generator if for all polynomially bounded adversaries A (uniform or non-uniform) we have $\delta(n) = \frac{1}{n^{\omega(1)}}$, where we define the distinguishing probability $\delta(n)$ to be*

$$\delta(n) = \left| \Pr_{x \in_u \{0,1\}^n} [A(g(x)) = 1] - \Pr_{y \in_u \{0,1\}^{\ell(n)}} [A(y) = 1] \right| .$$

Definition 2.2 (Next-Bit Unpredictability) *A poly-time computable function $g : 2^n \to 2^{\ell(n)}$ is next-bit unpredictable if for all polynomially bounded adversaries B (uniform or non-uniform) we have $\delta(n) = \frac{1}{n^{\omega(1)}}$, where we define $\delta(n)$ to be*

$$\delta(n) = \Pr_{\substack{I \in_u \{1,\ldots,\ell(n)\} \\ x \in_u \{0,1\}^n}} \left[B(g(x)_{\{1,\ldots,I-1\}}, I) = g(x)_I \right] - \frac{1}{2}.$$

Definition 2.3 (Next-Bit Predictability) *A poly-time computable function $g : 2^n \to 2^{\ell(n)}$ is next-bit predictable if g is not next-bit unpredictable.*

We want to show that next-bit unpredictability and pseudorandomness are equivalent in the sense that a function g is next-bit unpredictable if and only if it is a pseudorandom generator. We will prove this in the negation; that is, we will show that next-bit predictability for a given g implies that g fails to be pseudorandom and conversely.

THEOREM 2.4 *A poly-time computable function $g : 2^n \to 2^{\ell(n)}$ is next-bit unpredictable iff g is pseudorandom.*

Proof.

First we give an intuitively straightforward algorithm A for pseudorandomness detection on g given a next-bit predictor oracle B for g. We think of A as a *distinguisher*: the job of A is to attempt to distinguish the distribution generated by g from the uniform distribution.

A(Z - string of length $\ell(n)$)
  Uniformly pick $I \in_u \{1, \dots, \ell(n)\}$
  if $B(Z_{\{1,\dots,I-1\}}, I) = Z_I$ **then return** 1 (string generated by g)
        else return 0 (string from uniform distribution)

B runs in polytime, and so clearly A^B runs in polytime. By assumption we have

$$\Pr_{\substack{I \in_u \{1,\dots,\ell(n)\} \\ x \in_u \{0,1\}^n}} \left[B(Z_{\{1,\dots,I-1\}}, I) = Z_I\right] = \delta_B(n) + \frac{1}{2},$$

and by the definition of A we have

$$B(Z_{\{1,\dots,I-1\}}, I) = Z_I \iff A(Z) = 1.$$

Substituting into the definition of pseudorandomness we obtain

$$\delta_A(n) = \left| \delta_B(n) + \frac{1}{2} - \Pr_{y \in_u \{0,1\}^{\ell(n)}} [A(y) = 1] \right|.$$

Since y_I is selected uniformly at random, it follows that the second term is $\frac{1}{2}$, regardless of the value of $B(y_{1,\dots,I-1}, I)$. Thus,

$$\delta_A(n) = \left| \delta_B(n) + \frac{1}{2} - \frac{1}{2} \right| = \delta_B(n) \geq \frac{1}{poly(n)},$$

and the easy half of the proof is done.

Going the other way (pseudorandom detector to next-bit predictor), it is helpful to first define some distributions that interpolate between strings Y chosen from the uniform distribution and strings $g(X)$ chosen from the distribution generated by g. In the definitions below, let $Y \in_u$

$\{0,1\}^{\ell(n)}$ and $X \in_u \{0,1\}^n$.

$$\begin{array}{rcl} \multicolumn{3}{l}{\textit{Let}} \\ \mathscr{D}_0 &=& Y \\ \mathscr{D}_1 &=& \left\langle g(X)_{\{1\}}, Y_{\{2,\ldots,\ell(n)\}} \right\rangle \\ \mathscr{D}_{i-1} &=& \left\langle g(X)_{\{1,\ldots,i-1\}}, Y_{\{i,\ldots,\ell(n)\}} \right\rangle \\ \mathscr{D}_i &=& \left\langle g(X)_{\{1,\ldots,i\}}, Y_{\{i+1,\ldots,\ell(n)\}} \right\rangle \\ \mathscr{D}_{\ell(n)} &=& g(X)_{\{1,\ldots,\ell(n)\}} \end{array}$$

We also define δ_i for $0 \le i \le \ell(n)$ as follows.

$$\delta_i = \Pr_{Z \sim \mathscr{D}_i}[A[Z] = 1]$$

Note that we have $\delta(n) = |\delta_{\ell(n)} - \delta_0|$, and furthermore that without loss of generality we can remove the absolute value signs: if $\delta_{\ell(n)} - \delta_0$ is negative at some length n, we can adjust A so that it flips its answers at this length (we can do this due to the fact we are allowing non-uniform circuit families). We now present the next-bit predictor algorithm B^A. The intuitive idea is to favor values of Y_I that "look like fake randomness".

B(Z - string of length $\ell(n)$)
  Uniformly pick $I \in_u \{1, \ldots, \ell(n)\}$
  Uniformly pick $Y_{\{I,\ldots,\ell(n)\}} \in_u \{0,1\}^{\ell - I + 1}$
  if $A(Z_{1,\ldots,I-1}; Y_{I,\ldots,\ell(n)}) = 1$ **then return** Y_I
                else return $1 - Y_I$

We want to determine the probability that B is correct given the input is generated by g ($Z = g(X)$). The first step is to rewrite this probability using the equivalence $\Pr[A] = \Pr[A|B]\Pr[B] + \Pr[A|\bar{B}]\Pr[\bar{B}]$.

$$\begin{aligned} \Pr[B \text{ correct}] &= \frac{1}{\ell(n)} \sum_{i=1}^{\ell(n)} \Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = g(X)_i] \\ &= \frac{1}{\ell(n)} \sum_{i=1}^{\ell(n)} \big(\Pr[Y_i = g(X)_i] \cdot \Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = g(X)_i \mid Y_i = g(X)_i] \\ &\quad + \Pr[Y_i \neq g(X)_i] \cdot \Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = g(X)_i \mid Y_i \neq g(X)_i] \big) \end{aligned}$$

The following strategy allows us to deal with the first difficult term by rewriting the term so that it is clear that there is no dependence on Y_i.

$$
\begin{aligned}
\Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = g(X)_i \mid Y_i = g(X)_i] &= \Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = Y_i \mid Y_i = g(X)_i] \\
&= \Pr[A(g(X)_{\{1,\ldots,i-1\}}, Y_{\{i,\ldots,\ell(n)\}}) = 1 \mid Y_i = g(X)_i] \\
&= \Pr[A(g(X)_{\{1,\ldots,i\}}, Y_{\{i+1,\ldots,\ell(n)\}}) = 1 \mid Y_i = g(X)_i] \\
&= \Pr[A(g(X)_{\{1,\ldots,i\}}, Y_{\{i+1,\ldots,\ell(n)\}}) = 1] \\
&= \delta_i
\end{aligned}
$$

The other tricky term can be handled in a similar manner.

$$
\begin{aligned}
\Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = g(X)_i \mid Y_i \neq g(X)_i] &= \Pr[B(g(X)_{\{1,\ldots,i-1\}}, i) = 1 - Y_i \mid Y_i \neq g(X)_i] \\
&= \Pr[A(g(X)_{\{1,\ldots,i-1\}}, Y_{\{i,\ldots,\ell(n)\}}) = 0 \mid Y_i \neq g(X)_i] \\
&= \Pr[A(g(X)_{\{1,\ldots,i-1\}}, \overline{g(X)_i}, Y_{\{i+1,\ldots,\ell(n)\}}) = 0 \mid Y_i \neq g(X)_i] \\
&= \Pr[A(g(X)_{\{1,\ldots,i-1\}}, \overline{g(X)_i}, Y_{\{i+1,\ldots,\ell(n)\}}) = 0] \\
&= 1 - \Pr[A(g(X)_{\{1,\ldots,i-1\}}, \overline{g(X)_i}, Y_{\{i+1,\ldots,\ell(n)\}}) = 1] \\
&= 1 - \bar{\delta}_i,
\end{aligned}
$$

where we let

$$\bar{\delta}_i = \Pr[A(g(X)_{\{1,\ldots,i-1\}}, \overline{g(X)_i}, Y_{\{i+1,\ldots,\ell(n)\}}) = 1].$$

Using these equalities with the observation that $\delta_{i-1} = \frac{1}{2}(\delta_i + \bar{\delta}_i)$, we get

$$
\begin{aligned}
\Pr[B \text{ correct}] &= \frac{1}{\ell(n)} \sum_{i=1}^{\ell(n)} \big(\Pr[Y_i = g(X)_i] \cdot \delta_i + \Pr[Y_i \neq g(X)_i] \cdot (1 - \bar{\delta}_i)\big) \\
&= \frac{1}{\ell(n)} \sum_{i=1}^{\ell(n)} \left(\frac{1}{2}\delta_i + \frac{1}{2}(1 - \bar{\delta}_i)\right) \\
&= \frac{1}{\ell(n)} \sum_{i=1}^{\ell(n)} \left(\frac{1}{2}\delta_i + \frac{1}{2} - (\delta_{i-1} - \frac{1}{2}\delta_i)\right) \\
&= \frac{1}{\ell(n)} \sum_{i=1}^{\ell(n)} (\delta_i - \delta_{i-1}) + \frac{1}{2} \\
&= \frac{1}{\ell(n)} \big(\delta_{\ell(n)} - \delta_0\big) + \frac{1}{2} \\
&= \frac{\delta(n)}{\ell(n)} + \frac{1}{2} \\
&\geq \frac{1}{poly(n) \cdot \ell(n)} + \frac{1}{2} \\
&= \frac{1}{poly(n)} + \frac{1}{2}.
\end{aligned}
$$

♣

We note an alternate construction of B which may be more intuitive than the above construction and leave the proof of its correctness as an exercise. On input Z (a string of length $\ell(n)$), B will pick $i \in_u \{1, \ldots, \ell(n)\}$ and $Y \in_u \{0,1\}^n$ (we really only need the last $\ell(n) - i$ bits of Y) and run A twice on the input

$$Z_1 \ldots Z_{i-1}, b, Y_{i+1}, \ldots, Y_{\ell(n)},$$

once setting $b = 0$ and once setting $b = 1$. If A favors one of these runs, i.e. A outputs 1 for some setting of b but not for the other, then B outputs the value of b on which A accepted. Otherwise, if A outputs the same thing on both runs, B flips a random coin to decide whether to output 0 or 1.