18.405J/6.841J Advanced Complexity Theory	April 6/8 1999
Lecture 15/16	
Lecturer: Daniel A. Spielman	*Scribe: Matthias Ruhl and Rudi Seitz*

Derandomizing BPP

In the previous lectures we saw how to derandomize randomized logspace computations. Here we want to do the same thing for the class **BPP**. This time, we will need to base our result on some complexity theoretic assumptions instead of proving it from first principles.

The first results on derandomizing **BPP** relied on cryptographic assumptions, such as the hardness of RSA, which are usually stronger than $\mathbf{P} \neq \mathbf{NP}$. In this lecture we will present a result of Nisan and Wigderson, which uses a much weaker and more 'reasonable' assumption. This result serves as the foundation for most of the subsequent work on derandomizing BPP.

We begin by stating Nisan and Wigderson's main result:

Theorem 1 (Nisan, Wigderson) *If there exists a function $f \in \mathbf{EXP}$ that cannot be approximated by polynomial size circuits, then*

$$\mathbf{BPP} \subseteq \bigcap_{\varepsilon > 0} \mathit{TIME}(2^{n^\varepsilon}). \quad \square$$

We used the following definition in stating the theorem:

Definition 2 (Non-approximability by circuits) *We say a function f* cannot be approximated by polynomial size circuits, *if for all polynomials $p(n)$, and all sequences of circuits $\{C_n\}_{n=1}^\infty$ such that $|C_n| = p(n)$, the following holds:*

$$\exists k > 2 \text{ s.t. } \forall \text{ sufficiently large } n \ : \ \left| \Pr_{x \in \{0,1\}^n} [C_n(x) = f(x)] - \frac{1}{2} \right| < n^{-k} \quad \square$$

In this lecture we will show a slightly weaker result than Nisan and Wigderson's main theorem. We will employ the following definition:

Definition 3 (Hardness) *A function $f : \Sigma^* \to \{0,1\}$ has hardness h (where $h : \mathbb{N} \to \mathbb{N}$) if for all n, and for all circuits C of size $h(n)$:*

$$\left| \Pr_{x \in \{0,1\}^n} [C(x) = f(x)] - \frac{1}{2} \right| \leq \frac{1}{h(n)} \quad \square$$

Theorem 4 *If there exists a function $f \in \mathbf{EXP}$ with hardness $n^{\log n}$, then*

$$\mathbf{BPP} \subseteq \bigcap_{\varepsilon > 0} \mathit{TIME}(2^{n^\varepsilon}). \quad \square$$

Proof of Theorem

Let $f \in \mathbf{EXP}$ have hardness $n^{\log n}$. Fix $\varepsilon > 0$. We will show that $\mathbf{BPP} \in \mathrm{TIME}(2^{O(n^{c\varepsilon})})$, where 2^{n^c} is the time complexity of f. To do this, we will construct a pseudorandom number generator G that outputs some desired polynomial number of pseudorandom bits given only n^ε truly random bits as input. To derandomize a $\mathbf{BPP}$algorithm we will simply run the algorithm on all possible outputs of the generator for seed length n^ε, and take the majority vote.

So let A be some fixed $\mathbf{BPP}$ algorithm, which uses $p(n)$ random bits during its execution. To construct G, let the seed consist of the bits $x_1, \ldots, x_\ell$, where $\ell = n^\varepsilon$. The generator will apply the function f to several subsets S_i of the bits in the seed. These subsets are chosen according to the following restrictions:

$$S_1, \ldots, S_{p(n)} \subseteq \{1, 2, \ldots, \ell\}, \quad |S_i| = \sqrt{\ell}, \quad \forall i \neq j : |S_i \cap S_j| \leq \log p(n)$$

We claim without proof that it is possible (and in fact fairly easy) to construct such set systems, which are known as $(\sqrt{l}, \log p(n))$-designs.

Write x_S as an abbreviation for the bit-string $(x_i)_{i \in S}$. The generator then is just:

$$G(x_1 \ldots x_\ell) := f(x_{S_1}) \circ f(x_{S_2}) \circ \cdots \circ f(x_{S_{p(n)}})$$

First note that this generator can actually be computed in $\mathrm{TIME}(2^{O(n^{c\varepsilon})})$: since f has running time 2^{n^c}, it follows that G has running time $p(n) \cdot 2^{n^{c\varepsilon}}$. Again, derandomization can be accomplished by simply running G on all possible inputs, and giving those inputs to A as random strings (this takes total time $poly(n) \cdot 2^{n^{c\varepsilon}}$).

The proof that G is actually a pseudorandom generator proceeds by contradiction. Assume that there are inputs w_i with $|w_i| = i$, such that for all sufficiently large n we have

$$\left| \Pr_{r \in \{0,1\}^{p(n)}} [A(w_n, r) \text{ accepts }] - \Pr_{x \in \{0,1\}^{n^\varepsilon}} [A(w_n, G(x)) \text{ accepts }] \right| \geq \frac{1}{p(n)} \quad (*)$$

We will now derive a contradiction to the assumption that f has hardness $n^{\log n}$.

By "hard-wiring" the inputs w_n into A, we obtain the following statement equivalent to $(*)$: There exists a family $(A_n)_{n \in \mathbb{N}}$ of circuits of size $p(n)$, such that

$$\Pr_{r \in \{0,1\}^{p(n)}} [A_n(r) = 1] - \Pr_{x \in \{0,1\}^{n^\varepsilon}} [A_n(G(x)) = 1] \geq \frac{1}{p(n)}$$

(We omitted the absolute-value sign to simplify the presentation; it is easy to see that no generality is lost.)

Let $(z^j)_{j=0}^{p(n)}$ be random variables defined as follows:

$$z_i^j := \begin{cases} G(x)_i, & \text{for } i \leq j \\ r_i, & \text{for } i > j \end{cases}$$

where r is chosen uniformly from $\{0,1\}^{p(n)}$ and x is chosen uniformly from $\{0,1\}^{n^\varepsilon}$).

Then $z^0 = r$, and $z^{p(n)} = G(x)$. By a straightforward hybrid argument we can conclude that for every n there is some fixed j, such that

$$\Pr\left[A_n(z^j) = 1\right] - \Pr\left[A_n(z^{j-1}) = 1\right] > \frac{1}{p(n)^2} \quad (**)$$

Now, if we look at the strings z^{j-1} and z^j, we see that they differ at only one position:

$$\begin{aligned} z^{j-1} &= f(x_{S_1}) \circ \cdots \circ f(x_{S_{j-1}}) \circ r_j \circ r_{j+1} \circ \ldots r_{p(n)} \\ z^j &= f(x_{S_1}) \circ \cdots \circ f(x_{S_{j-1}}) \circ f(x_{S_j}) \circ r_{j+1} \circ \ldots r_{p(n)} \end{aligned}$$

We will define a circuit D that takes as input $f(x_{S_1}), \ldots, f(x_{S_{j-1}})$ and attempts to compute $f(x_{S_j})$. The circuit D flips $n-j+1$ random bits $r_j, \ldots, r_n$ and computes $A_n(f(x_{S_1}), \ldots, f(x_{S_{j-1}}), r_j, \ldots, r_n)$. If this evaluates to 1, then D outputs r_j, otherwise it outputs $1 - r_j$. It is possible to show, using an argument due to Yao, that

$$Pr[D(f(x_{S_1}), \ldots, f(x_{S_{j-1}})) = f(x_{S_j})] - 1/2 > 1/n^2.$$

where the probability is taken over x as well as $D's$ internal random bits $r_j, \ldots, r_n$.

By linearity of expectation, there must be a way to fix the values of $r_j, \ldots, r_{p(n)}$ and x_i for $i \notin S_j$, such that this inequality still holds. From here on, assume these values have been so fixed.

We will now construct a circit C that attempts to compute $f(x_{S_j})$ given x_{S_j} as input. Without loss of generality, assume that $S_j = \{1, \ldots m\}$. Notice that each value $f(S_k)$ where $k \neq j$ depends on at most $log\ p(n)$ of the bits in x_{S_j}. Therefore each of these values (considered as a function of x_{S_j}) can be computed by a circuit of size $p(n)$. Our circuit C will simply take x_{S_j} as input, compute $f(x_{S_1}), \ldots, f(x_{S_{j-1}})$, and output $D(f(x_{S_1}), \ldots, f(x_{S_{j-1}}))$. (Remember that D's internal random bits have been fixed.)

It is easy to see that C has polynomial size in n and computes $f(x_{S_j})$ for the $\log p(n)$ bit-string x_{S_j} with non-negligible advantage. Thus, f is not $n^{\log n}$-hard. ■