Complexity Theory II	Course Instructor: V. Arvind

Lecture 03: Aug 09, 2006

Lecturer: V. Arvind | *Scribe: Shreevatsa R*

1 Overview

In the last lecture, we proved that a hard function can be used to derandomise BPP in SUBEXP:

Theorem 1. *If there is a function computable in* E *that has* n^c *hardness for every* $c > 0$*, then* $\mathsf{BPP} \subseteq \mathsf{SUBEXP} = \bigcup_{\epsilon>0} \mathsf{DTIME}(2^{n^\epsilon})$.

We also stated the Babai–Fortnow–Nisan–Wigderson theorem, and saw a sketch of the proof. In this lecture we actually prove it.

2 Initial remarks

Theorem 2 (BFNW). *If* $\mathsf{EXP} \not\subset \text{io-}\mathsf{P/poly}$*, then* $\mathsf{BPP} \subseteq \mathsf{SUBEXP}$.

As $\mathsf{EXP} \not\subset \text{io-}\mathsf{P/poly}$ is equivalent to $\mathsf{E} \not\subset \text{io-}\mathsf{P/poly}$, the hypothesis says that there exists some $f \in \mathsf{E}$, $f \notin \text{io-}\mathsf{P/poly}$, i.e., f is such that for every polysize circuit family $C = \{C_n\}$, $\Pr_{x\in\{0,1\}^n}[f(x) = C(x)] < 1$ for all but finitely many n. This is "worst-case hardness".

¿From this, we want to derive "average-case hardness", improving the "< 1" above to a negligible quantity. The proof will successively construct harder functions, starting with f, until we have one that satisfies the hypothesis of Theorem 1.

3 Constructing a harder function g

3.1 Arithmetise f and interpolate

$f_n : \{0,1\}^n \to \{0,1\}$. Pick a finite field F of size $n^{O(1)}$, say $F = \mathbb{F}_{2^k}$ where $k = O(\log n)$. We can assume $\{0,1\} \subset F$.

Find $g_n : F^n \to F$ such that g_n extends f_n, i.e., g_n coincides with f_n on all inputs from $\{0,1\}^n$. We can find this by interpolation: for each

$a \in \{0,1\}^n$, define $P_a(x_1, \ldots, x_n) = \prod_{i=1}^{n}(1 - a_i - x_i)$. P_a has degree n, takes the value 1 at a, and takes the value 0 for all other $b \in \{0,1\}^n$.

Define

$$g_n(x_1, \ldots, x_n) = \sum_{\{a \mid f_n(a)=1\}} P_a(x_1, \ldots, x_n) \ .$$

g_n has degree at most n, and $g = \{g_n\}$ agrees with f on $\{0,1\}^n$ for all n. g_n is a function $\{0,1\}^{nk} \to \{0,1\}^k$ where $k = O(\log n)$ ($2 \log n$, say). So $g = \{g_n\}$ is computable in $2^{O(n)}$ time.

3.2 Hardness claim

For every polysize circuit family $C' = \{C'_n\}$,

$$\Pr_{x \in F^n}[g_n(x) = C'_n(x)] < 1 - \frac{1}{3n}$$

for all but finitely many n.

Proof. Suppose not, i.e., suppose there exists C' which does better. Then we shall give a randomised polytime algorithm that uses C' as subroutine and computes g_n on all of F^n, and thus computes f.

Let x be any element of F^n. Pick r uniformly at random from F^n. Then $x + tr$, for $t \in F$, $t \neq 0$, is also a random variable with uniform distribution. Define the polynomial P as $P(t) = g_n(x + tr)$. As $\deg P \leq n$, it is sufficient to know its value at $n+1$ points to determine it completely.

Let $t_1, t_2, \ldots, t_{n+1}$ be $n+1$ distinct points in F^*. Compute $C'_n(x + t_i r)$ for each i, $1 \leq i \leq n+1$. This computation is wrong with probability at most $1/(3n)$, by our assumption about C'. As this is true for each i,

$$\Pr[\exists i : C'_n(x + t_i r) \neq g_n(x + t_i r)] \leq \frac{n+1}{3n} \leq \frac{2}{5} \ .$$

So we can find P (and hence $P(0) = g_n(x)$) with probability at least $3/5$. As this is a BPP algorithm, and $\mathsf{BPP} \subseteq \mathsf{P/poly}$, this contradicts the hardness of f. $\square$

Next, we *amplify* the hardness of g: we define a $\hat{g}$, also computable in E, such that

$$\Pr[C(x) = \hat{g}(x)] \leq \frac{1}{p(n)} \qquad (1)$$

for every polynomial p.

4 Constructing $\hat{g}$: The direct product lemma

The direct product lemma gives us a way of constructing harder functions from a given function. It states the following:

1. Suppose the function $f : \{0,1\}^n \to \{0,1\}^t$ is such that for all circuits C of size s, $\Pr[f(x) = C(x)] < \delta$. Then, for any $\epsilon > 0$, if $k \geq O(\frac{\log(1/\epsilon)}{1-\delta})$, the function $g : \{0,1\}^{nk} \to \{0,1\}^{tk}$ defined as $g(x_1, x_2, \ldots, x_k) = (f(x_1)f(x_2)\ldots f(x_k))$ satisfies the property that for all circuits C' of size $O(\frac{\epsilon}{\log(1/\epsilon)})$, $\Pr[g(x) = C'(x)] \leq \epsilon$.

2. Suppose the function $g \in E$ satisfies the property that for a fixed polynomial $q(n)$, for every polysize circuit C,
$$\Pr[g(x) = C(x)] < 1 - \frac{1}{q(n)},$$
then letting $k = nq(n)$ in the above, we have a function $\hat{g} : \{0,1\}^n \to \{0,1\}^{t(n)}$ such that for every polysize circuit family C',
$$\Pr[\hat{g}(x) = C'(x)] < \frac{1}{p(n)}$$
for every polynomial p almost everywhere.

We now have a function $\hat{g}$ that has the hardness claimed in equation 1.

5 The Goldreich–Levin theorem

Let $v \in \{0,1\}^n$ be a "hidden vector". Suppose G is a randomised polytime algorithm such that
$$\Pr[G(r) = \langle v, r \rangle] \geq \frac{1}{2} + \epsilon,$$
the probability being taken over all choices of r from $\{0,1\}^n$ and over G's coin tosses. Then,

Theorem 3. *There is a poly$(n, 1/\epsilon)$ time algorithm that outputs v with probability at least $\frac{\epsilon^2}{2n}$.*

(Note: $v \mapsto [\langle v, 0^n \rangle, \langle v, 0^{n-1}1 \rangle, \langle v, 0^{n-2}10 \rangle, \langle v, 0^{n-2}11 \rangle, \ldots, \langle v, 1^n \rangle]$ is called the Hadamard code. We shall see later that the Goldreich–Levin theorem can be thought of as *list decoding* the Hadamard code.)

Proof. Let $e_1, e_2, \ldots, e_n$ be the standard basis of $\{0,1\}^n$. The naive idea would be to pick a random r from $\{0,1\}^n$, and find $G(r) \oplus G(r \oplus e_i)$. As $r \oplus e_i$ is also randomly distributed in $\{0,1\}^n$, with a probability better than half, this will be equal to $\langle v, r\rangle \oplus \langle v, (r \oplus e_i)\rangle = \langle v, e_i\rangle = v_i$.

The actual idea is to avoid make two calls to G. We guess the value of $\langle v, r\rangle$, and use G to compute only $\langle v, r\rangle \oplus G(r \oplus e_i)$.

Choose m=$poly(n, 1/\epsilon)$, and $l = \log(m+1)$. Pick $r_1, r_2, \ldots, r_l$ independently and uniformly at random from $\{0,1\}^n$. Define $r_J = \sum_{i \in J} r_i$, for each of the $m = 2^l - 1$ nonempty subsets J of $\{1, \ldots, l\}$. Similarly, guess σ_i, for each i, and define $\sigma_J = \sum_{i \in J} \sigma_i$. Clearly, as r_J is 0 or 1 with equal probability,

$$\Pr[\langle v, r_J\rangle \text{ is correct for each } J] = \frac{1}{2^l} = \frac{1}{m+1}$$

Our algorithm does the following: for each i, let

$$z_i = \operatorname*{maj}_J \sigma_J \oplus G(r_J \oplus e_i)\ .$$

Output $z = z_1 z_2 \ldots z_n$.

Claim 4. *If all the guesses σ_i are correct, then $z = v$ with probability more than half.*

Proof. We first prove the following subclaim: Assuming that all the guesses are correct,

$$\Pr\left[|\{J : \sigma_J \oplus G(r_J \oplus e_i) = v_i\}| \geq \frac{2^l - 1}{2}\right] \geq 1 - \frac{1}{2n}$$

Define, for each J, $X_J = 1$ if $\sigma_J \oplus G(r_J \oplus e_i) = v_i$ and 0 otherwise. From the hypothesis (of the Goldreich–Levin theorem) we know that

$$\mathbb{E}[X_J] \geq \frac{1}{2} + \epsilon$$
$$\mathbb{E}\left[\sum X_J\right] \geq \left(\frac{1}{2} + \epsilon\right) m$$

The probability of the "bad event" is

$$\begin{aligned}
\Pr\left[\sum X_J t < \frac{m}{2}\right] &\leq \Pr\left[\left|\sum X_J - \mathbb{E}\left[\sum X_J\right]\right| > m\epsilon\right] \\
&\leq \frac{\operatorname{Var}(\sum X_J)}{\epsilon^2 m^2} \qquad \text{(Chebyshev's inequality)} \\
&= \frac{\sum(\operatorname{Var} X_J)}{\epsilon^2 m^2} \\
&= \frac{m(\operatorname{Var} X_{\{1\}})}{\epsilon^2 m^2} \\
&= \frac{1}{\epsilon^2 m}\left(\mathbb{E}[X_{\{1\}}^2] - \mathbb{E}[X_{\{1\}}]^2\right) \\
&= \frac{\mathbb{E}[X_{\{1\}}](1 - \mathbb{E}[X_{\{1\}}])}{\epsilon^2 m} \\
&\leq \frac{1}{4\epsilon^2 m}
\end{aligned}$$

which is less than $\frac{1}{2n}$ when $m \geq \frac{n}{2\epsilon^2}$.

This proves the subclaim, and hence the claim. $\square$

When all the guesses are correct, the algorithm outputs v with probability at least half. Thus, the probability that the complete algorithm outputs the correct v is at least $\frac{1}{2(m+1)} \geq \frac{\epsilon^2}{4n}$.

$\square$

6 Constructing a hard $\tilde{g}$

As we saw at the end of section 4, we have a function $\hat{g}$ for which

$$\Pr[C(x) = \hat{g}(x)] \leq \frac{1}{p(n)}$$

for every polynomial p. We define a new function $\tilde{g}$ as $\{\tilde{g}_n\}$, where

$$\tilde{g}_n : \{0,1\}^n \times \{0,1\}^{t(n)} \to \{0,1\}$$

is defined as

$$\tilde{g}_n(x, r) = \langle \hat{g}_n(x), r \rangle \pmod 2.$$

Once we prove that $\tilde{g}_n$ has hardness $p(n)$ for every polynomial p, we will have proved the BFNW theorem, for this $\tilde{g}$ satisifes the hypothesis of theorem 1. Thus it only remains to prove the hardness of $\tilde{g}$.

We prove this by contradiction. Suppose there exists a polysize circuit family $\tilde{C}$ and a polynomial n^c such that

$$\Pr_{x,r}[\tilde{g}_n(x,r) = \tilde{C}(x,r)] \geq \frac{1}{2} + \frac{1}{n^c} \tag{2}$$

for infinitely many n.

Define the random variable $X(x)$ to be $\Pr_r[\tilde{g}_n(x,r) = \tilde{C}(x,r)]$. We have assumed that

$$\mathop{\mathbb{E}}_{x\in\{0,1\}^n}[X(x)] \geq \frac{1}{2} + \frac{1}{n^c}$$

for infinitely many n.

That is,

$$\frac{1}{2} + \frac{1}{n^c} \leq \sum_{a\in\{0,1\}^n} X(a)p_a \text{ , where } p_a = \frac{1}{2^n}$$

We can split the right hand side above as the sum of

$$\sum_{\left\{a | X(a) > \frac{1}{2} + \frac{1}{2n^c}\right\}} X(a)p_a \leq \Pr_{a\in\{0,1\}^n}\left[X(a) > \frac{1}{2} + \frac{1}{2n^c}\right]$$

(using the fact that $X(a) \leq 1$) and

$$\sum_{\left\{a | X(a) \leq \frac{1}{2} + \frac{1}{2n^c}\right\}} X(a)p_a \leq \frac{1}{2} + \frac{1}{2n^c}$$

(using the fact that $\sum p_a \leq 1$). Thus, we have

$$\Pr_{a\in\{0,1\}^n}\left[X(a) > \frac{1}{2} + \frac{1}{2n^c}\right] \geq \frac{1}{2n^c}$$

which gives a lower bound on the size of the set $S = \left\{a \mid X(a) \geq \frac{1}{2} + \frac{1}{2n^c}\right\}$:

$$|S| \geq \frac{2^n}{2n^c} \tag{3}$$

Now notice that the Goldreich–Levin theorem applies in this setting: for any fixed $a \in S$, we have a polytime algorithm $\tilde{C}$ such that

$$\Pr_r\left[\tilde{C}(a,r) = \langle \hat{g}_n(x), r\rangle\right] > \frac{1}{2n^c}$$

By the theorem, there exists a randomised polysize circuit family $\left\{\tilde{\tilde{C}}\right\}$ such that (for every $a \in S$)

$$\Pr\left[\tilde{\tilde{C}}(a,r) = \hat{g}_n(a)\right] \geq \frac{1}{q(n)}$$

where the probability is taken over choices of r from $\{0,1\}^n$ and over $\tilde{\tilde{C}}$'s internal coin tosses, and $\frac{1}{q(n)}$ is $\frac{\epsilon^2}{2n} = \frac{1}{8n^{2c+1}}$.

In other words, for each a, at least $\frac{1}{q(n)}$ of the random choices work. Thus there must exist a *fixed* choice which works for at least $\frac{1}{q(n)}$ of the as in S. That is, we can fix the random choices of r and the internal choices in the computation of $\tilde{\tilde{C}}$ to get a polysize circuit C, so that there exists a set S' of size at least $\frac{1}{q(n)}$ the size of S satisifying: for every $a \in S'$, $C(a) = \hat{g}(a)$.

Using equation 3, we see that

$$\Pr\left[C(x) = \hat{g}(x)\right] \geq \frac{|S'|}{2^n} \geq \frac{1}{2n^c}\frac{1}{8n^{2c+1}},$$

which contradicts equation 1. This proves that our assumption in equation 2 must be wrong, and hence concludes the proof of the BFNW theorem.