80240233: Complexity of Computation
ITCS, Tsinghua Univesity, Fall 2007

Lecture 15
27 November 2007

Instructor: Andrej Bogdanov

Notes by: Yingchao Zhao and Yongxi Cheng

In lecture 9 we saw that if we could prove certain kinds of circuit lower bounds, we would also obtain derandomizations like BPP = P or AM = NP. Since the circuit lower bounds in question are believed to hold, this makes the corresponding derandomizations also plausible. But proving circuit lower bounds like the ones from lecture 9 is difficult. Can we deduce, say, BPP = P without relying on unproven and difficult to prove circuit lower bounds? This is not known. However, a certain kind of circuit lower bound is *necessary* to prove in order to show that NP = AM. We will discuss this "reverse connection" between circuit lower bounds and derandomization in this lecture.

We will sketch the proof of the following theorem.

Theorem 1 (Impagliazzo, Kabanets, Wigderson)**.** *If* NEXP $\subseteq$ P/poly*, then* NEXP = MA*.*

Here, MA is the class of decision problems computed by one-round interactive protocols (where the prover goes first): Formally, $L \in$ MA if there is a polynomial p and a randomized polynomial-time machine V such that

$$x \in L \Longrightarrow \exists y, |y| = p(|x|) : \Pr[V(x, y) \text{ accepts}] \geq 2/3$$
$$x \notin L \Longrightarrow \forall y, |y| = p(|x|) : \Pr[V(x, y) \text{ accepts}] \leq 1/3$$

It is straightforward that NP $\subseteq$ MA $\subseteq$ AM. (There seem to be no examples of decision problems known to be in MA but not in NP.)

What does this mean? Suppose that we cave proved that MA = NP. Since by the nondeterministic time hierarchy theorem NP $\neq$ NEXP, we have NEXP $\neq$ MA and so NEXP $\not\subseteq$ P/poly. This is a circuit lower bound for some function in NEXP – one that we do not know how to prove unconditionally.

The converse of Theorem 1 is also known to hold. However, to obtain nontrivial nondeterministic simulations of MA seems to require stronger hardness assumptions than NEXP $\not\subseteq$ P/poly. So the method we present here gives a result which does not seem "optimal" in the quantitative sense, but yields an interesting qualitative insight: In the setting of constant round interactive proofs, circuit lower bounds are both necessary and sufficient for derandomization.

1 Circuit lower bounds for EXP

To prove Theorem 1 we start with the following similar looking statement:

Theorem 2. *If* EXP $\subseteq$ P/poly*, then* EXP = MA*.*

Proof Sketch. Recall the proof that NEXP $\subseteq$ PCP from the last two lectures. Suppose $L \in$ NEXP and let N be the exponential time NTM that decides L. In the probabilistic checkable proof system, to check $x \in L$, the verifier V probabilistically verifies a proof π which consists of two parts:

1. A description of a function $A : \mathbb{F}^m \to \mathbb{F}$, which (in a correct proof) is a multilinear extension of some accepting computation of $N(x)$.
2. A description of the (honest) prover P in an interactive protocol which, given oracle access to A, allows the verifier to "compute" $\sum_{y \in \{0,1\}^m} p(y, \tilde{A}(y))$, where p is some explicit polynomial. Recall that the job of the prover in this protocol is to compute partial sums of the form

$$\sum_{y_i \in \{0,1\}, \ldots, y_m \in \{0,1\}} p(r_1, \ldots, r_{i-1}, y_i, \ldots, y_m, \tilde{A}(r_1, \ldots, r_{i-1}, y_i, \ldots, y_m)).$$

If $L \in$ EXP, then for every $x \in L$ there is a unique accepting computation tableau for N on input x, because N is then deterministic. Since the multilinear extension is unique, the function A is then also uniquely determined and can be computed in exponential time. The prover P above can also be implemented in exponential time. Thus, exists exponential time algorithm B such that $B(x, y)$ computes $\pi_x(y)$, which is the proof for $x \in L$ at location y.

Now let's assume that EXP $\subseteq$ P/poly. Then B can be computed by a circuit family C of polynomial size. The following is an MA protocol for L: On input x of length n,

Prover: Send description of C (on sufficiently large input length) to V;
Verifier: Simulate and output $V(x)$; when V queries y in the probabilistically checkable proof, use $C(x, y)$ as the answer.

Since, by assumption, $C(x, y) = \pi_x(y)$, where π is the honest proof for $x \in L$ whenever this is the case, it follows that when $x \in L$ then the verifier in the above protocol accepts with probability one, while when $x \notin L$ then no matter what C is the verifier rejects with probability 1/3. □

2 Some derandomization results

Theorem 2 will be used in the proof of Theorem 1. Apart from this, we will also need to use a derandomization result under uses a hardness hypothesis. The derandomization result we need is in some ways similar to the ones we saw in Lecture 9. However, we do not have all the necessary ingredients to give a formal proof here. Instead, we will try to explain it by analogy to other results we have already seen.

First some notation: Let E and NE be the classes of all decision problems that can be decided in deterministic and nondeterministic time $2^{O(n)}$, respectively. (In contrast, EXP and NEXP allow time $2^{\text{poly}(n)}$.)

We now start with the derandomization result of Impagliazzo, Nisan, and Wigderson and show how it can be modified in several steps to obtain something which will be useful for proving our theorem.

Theorem 3. *If there exists* $f \in \mathrm{E}$ *which is hard for any circuit family of size* $2^{\delta n}$ *for some* $\delta > 0$*, then* $\mathrm{BPP} = \mathrm{P}$*.*

Here, and elsewhere in this lecture, by "hard" we mean worst-case hard: Every circuit family of the given size fails on some sufficiently large input.

What happens when instead of BPP = P we are interested in proving AM = NP? This can be done under a different assumption:

Theorem 4. *If there exists* $f \in \mathrm{NE} \cap \mathrm{coNE}$ *which is hard for every* **nondeterministic** *circuit family of size* $2^{\delta n}$ *for some* $\delta > 0$*, then* $\mathrm{AM} = \mathrm{NP}$*.*

Now suppose we do not about complete derandomization of AM, but only want a "weak" derandomization. Trivially, we know that $\mathrm{AM} \subseteq \mathrm{NEXP}$. Theorem 4 tells us $\mathrm{AM} \subseteq \mathrm{NP}$. We can obtain intermediate results – for instance $\mathrm{AM} \subseteq \mathrm{NE}$ – by using assumptions weaker than in Theorem 4:

Theorem 5. *Suppose that for every* $c > 0$ *there exists* $f \in \mathrm{NE} \cap \mathrm{coNE}$ *which is hard for every nondeterministic circuit family of size* n^c*. Then* $\mathrm{AM} \subseteq \mathrm{NE}$*.*

The next step is technical but (unfortunately) necessary: It considers what happens when we introduce some nonuniformity in both the circuit lower bound and the derandomization result. Let us recall the definition of machines with advice. These are machines that in addition to the input x obtain a canonical string a that depends only on the input length of x.

Here we will be interested in the complexity class NE/n (NE with n bits of advice): This is the class of all decision problems L for which there exists a nondeterministic Turing Machine N running in time $2^{O(n)}$ and taking two inputs x and a and a sequence of advice strings $a_1, \ldots, |a_n| = n$ such that $x \in L$ if and only if $N(x, a_{|x|})$ accepts.

Theorem 6. *Suppose that for every* $c > 0$ *there exists* $f \in \mathrm{NE}/O(n)$ *which is hard for every nondeterministic circuit family of size* n^c*. Then* $\mathrm{AM} \subseteq \mathrm{NE}/n$*.*

Notice that in Theorem 5 part of the assumption was $f \in \mathrm{NE} \cap \mathrm{coNE}$, while in Theorem 6 it became $f \in \mathrm{NE}/O(n)$. What happened to $\mathrm{coNE}/O(n)$? It turns out that $\mathrm{NE}/O(n) = \mathrm{coNE}/O(n)$.

The next step is also somewhat technical. When we say "f is hard for a class of circuits C", what we have meant so far is that for every sufficiently large input length n, there exists some $x \in \{0,1\}^n$ such that $f(x) \neq C(x)$. Here we will need a weaker notion of hardness: We won't require that f and C differ on all sufficiently large input lengths, but merely on infinitely many inputs. (We were never explicit about this minor point before, but it makes a difference here.)

We will say that f is *infinitely often (i.o.) hard for* C if $f(x) \neq C(x)$ for infinitely many $x \in \{0,1\}^*$. For two complexity classes $\mathbf{C}$ and $\mathbf{D}$, we will say $\mathbf{C}$ is *infinitely often contained* in $\mathbf{D}$, or $\mathbf{C} \subseteq_{\mathrm{i.o.}} \mathbf{D}$, if for every $L \in \mathbf{C}$ there exists some $L' \in \mathbf{D}$ such that $L \cap \{0,1\}^n = L' \cap \{0,1\}^n$ for infinitely many n.

Since hardness versus randomness proofs look at every input length separately, the results carry over to the infinitely often setting:

Theorem 7. *Suppose that for every $c > 0$ there exists $f \in \mathrm{NE}/O(n)$ which is i.o. hard for every nondeterministic circuit family of size n^c. Then* $\mathrm{AM} \subseteq_{\mathrm{i.o.}} \mathrm{NE}/n$.

We are now close to stating the result needed for the proof of Theorem 1. To explain this step, let us go back to Theorem 3. This theorem assumes the existence of a hard function in E. Recall that a function is in E if for every x of length n, $f(x)$ can be computed in time $2^{O(n)}$. Notice that we could equivalently require that the table of *all* values $f(x)$, where $x \in \{0,1\}^n$ be computed in time $2^{O(n)}$. This table – called the *truth table* of f on length n and denoted by $\langle f_n \rangle$ – can be represented by a string of length 2^n: At position x, the string contains the bit $f(x)$.

While for deterministic algorithms running in time $2^{O(n)}$, computing a function and computing its truth table are essentially the same task, an interesting issues arises when non-deterministic algorithms come into play. Computing the truth table of a function is not even a decision problem, so what does it mean to say that the truth table of f is computable in nondeterministic exponential time? One reasonable possibility is to require that on input 1^n, all accepting computation paths of the machine halt with the string $\langle f_n \rangle$ on the output tape.

However, for the purpose of derandomization, a more relaxed notion is possible: Since all we care is that the truth table of the function output by the machine is sufficiently hard, it is even permissible to have different accepting computation paths of the NE machine output different functions – as long as all these functions are hard, and there is at least one accepting path.

Now we do not even have a single hard function f, but a family of hard functions $\mathcal{F}$, one corresponding to each accepting output of the NE machine. We will say that a family of boolean functions $\mathcal{F}$ is *i.o. hard* for nondeterministic circuits of size S if there are infinitely many n such that *every* $f_n : \{0,1\}^n \to \{0,1\}$ in $\mathcal{F}$ cannot be computed by a nondeterministic circuit of size $S(n)$. We will call $\widetilde{\mathrm{NE}}/O(n)$ the class of all function families $\mathcal{F}$ if there exists a nondeterministic machine N running in time $2^{O(n)}$ taking $O(n)$ bits of advice such that on every computation path, $N(1^n)$ outputs the truth table of some function $f_n : \{0,1\}^n \to \{0,1\}$ and the truth tables of functions output on the accepting paths are exactly those $f_n \in \mathcal{F}$.

Theorem 8. *Suppose that for every $c > 0$ there exists an infinite family of boolean functions $\mathcal{F} \in \widetilde{\mathrm{NE}}/O(n)$ that is i.o. hard for nondeterministic circuits of size n^c. Then* $\mathrm{AM} \subseteq_{\mathrm{i.o.}} \mathrm{NE}/n$.

2.1 Proof Sketch of Theorem 1

We sketch the proof of Theorem 1. If NEXP = EXP, then Theorem 1 follows from Theorem 2, so we will assume that NEXP $\neq$ EXP. We can then use the following claim.

Claim 9. *If* NEXP $\neq$ EXP*, then for every $c > 0$ there exists a function family $\mathcal{F} \in \widetilde{\mathrm{NE}}/(n+1)$ such that $\mathcal{F}$ is i.o. hard for nondeterministic circuits of size n^c.*

Using this claim and Theorem 8, we can deduce that

$$\mathrm{NEXP} \neq \mathrm{EXP} \implies \mathrm{AM} \subseteq_{\mathrm{i.o.}} \mathrm{NE}/n. \tag{1}$$

Now suppose that NEXP $\subseteq$ P/poly; we will show that this is impossible (unless NEXP = EXP). We will need the following simple claim:

Claim 10. *If* NEXP $\subseteq$ P/poly, *then* NE/$n \subseteq$ SIZE($O(n^d)$) *for some* $d > 0$.

We now can do the following sequence of derivations:

$$\begin{aligned}
\text{NEXP} \subseteq \text{P/poly} &\implies \text{EXP} \subseteq \text{P/poly} \\
&\implies \text{EXP} = \text{MA} && \text{by Theorem 2} \\
&\implies \Sigma_4 \subseteq \text{AM} \\
&\implies \Sigma_4 \subseteq_{\text{i.o.}} \text{NE}/n && \text{by (1)} \\
&\implies \Sigma_4 \subseteq_{\text{i.o.}} \text{SIZE}(O(n^d)).
\end{aligned}$$

Recall that in Homework 1, we saw that $\Sigma_4 \not\subseteq \text{SIZE}(O(n^d))$. In fact, the same proof shows that $\Sigma_4 \not\subseteq_{\text{i.o.}} \text{SIZE}(O(n^d))$, so the assumption NEXP $\subseteq$ P/poly must be false.

It remains to prove Claim 9 and Claim 10.

Proof of Claim 9. Assume there exists a decision problem $L \in$ NEXP such that $L \notin$ EXP. We will show how to obtain the desired function family $\mathcal{F}$ from L. Since $L \in$ NEXP, there exists a relation $R(x, y)$ computable in time $2^{|x|^d}$, and such that

$$x \in L \iff R(x, y) = 1 \text{ for some } y \text{ of length } 2^{|x|^d}.$$

Here $d > 0$ is some constant. Without loss of generality, we may assume that $d = 1$ (by padding).

Since y is a string of length 2^n, we can think of it as representing the truth table of some function $f : \{0,1\}^n \to \{0,1\}$. So we can rephrase the above condition as

$$x \in L \iff R(x, \langle f \rangle) = 1 \text{ for some } f : \{0,1\}^{|x|} \to \{0,1\}.$$

Now instead of looking at *all* functions f, let us restrict attention to those f computable in time n^c. We claim that there are infinitely many $x \in \{0,1\}^*$ such that $L(x) = 1$, but $R(x, \langle f \rangle) = 0$ for all f computable by circuits of size $|x|^c$. If not, the following is an EXP algorithm for L: On input x, accept iff $R(x, \langle C \rangle) = 1$ for some (function computed by a) nondeterministic circuit C of size $|x|^c$.

Let $a_1, \ldots$ be an infinite list of all x such that $L(x) = 1$ but $R(x, \langle f \rangle) = 0$ for all f computable by circuits of size $|x|^c$. If we "knew" this list, the following $\widetilde{\text{NE}}$ algorithm A outputs the desired list of functions $\mathcal{F}$: On input 1^n,

1. If there exists no a_i of length n, reject. Otherwise, choose an arbitrary a_i of length n.
2. Nondeterministically guess a function $f : \{0,1\}^n \to \{0,1\}$.
3. If $R(a_i, \langle f \rangle) = 1$, output $\langle f \rangle$ and accept. Otherwise reject.

By construction, any f output by the algorithm in an accepting state is hard for nondeterministic circuits of size n^c. By what we proved about the a_is, the list of functions output by the algorithm (over the various choices of n, a_i, and the nondeterminism) is infinite.

We don't know the actual list $a_1, \ldots, a_n$, but we can give it as advice to the algorithm A: For an input length n, if a_i in step 1 exists use advice $1a_i$, and if it doesn't exist use advice 0^{n+1}. So we obtain a nondeterministic algorithm with $n+1$ bits of advice that produces the truth tables of an infinite list of functions, all of them hard for nondeterministic circuits of size n^c. □

Proof of Claim 10. Assume NEXP $\subseteq$ P/poly. Let EBH be the NEXP-complete problem

$(N, x, 1^n)$: N is a nondeterministic machine that accepts x within 2^n steps.

Since EBH $\in$ NEXP, it can be computed by some family of polynomial-size circuits, say of size $O(n^d)$.

Now let $L \in$ NE/n, so there exists a nondeterministic machine N that runs in time $2^{O(n)}$ and a sequence $a_1, \ldots,$ where $|a_n| = n$ such that

$$x \in L \iff N(a_n, x) \text{ accepts, where } |x| = n.$$

Now $N(a_n, x)$ accepts iff $(N, (a_n, x), 1^{O(n)}) \in$ EBH, and this can be decided by a circuit of size $O(n^d)$. By hardwiring the advice strings, we obtain a family of circuits of size $O(n^d)$ for L. □