18.405J/6.841J Advanced Complexity Theory	February 27, 2001

Lecture 6

Lecturer: Dan Spielman *Scribe: Edward Early*

1 Relativization, The Baker-Gill-Solovay Theorem

Theorem 1 (Baker-Gill-Solovay) $\exists$ *oracles A and B for which* $\mathrm{P}^A = \mathrm{NP}^A$ *and* $\mathrm{P}^B \neq \mathrm{NP}^B$.

Before giving a proof, we discuss relativization, i.e. stuff with oracles. We can relativize the proof in the previous lecture to get $\mathrm{NP}^A \subseteq \mathrm{P}^A/\mathrm{poly} \Rightarrow (\Sigma_2^{\mathrm{P}})^{\mathrm{A}} = (\Pi_2^{\mathrm{P}})^{\mathrm{A}}$. Of the four ATIME, ASPACE containment theorems that we proved earlier, three of them relativize.

Note that, in light of the Baker-Gill-Solovay theorem, any technique that relativizes won't resolve P vs. NP. In general, if we want to compare complexity classes C_1 and C_2, it can be helpful to ask if there exists A such that $C_1^A = C_2^A$, or if equality holds for most A. For example, Sipser proved that $\exists A$ s.t. $\forall k (\Sigma_k^P)^A \neq (\Pi_k^P)^A$, adding to the plausibility that the polynomial hierarchy does not collapse.

While it is known that IP=PSPACE, it was previously known that $\exists A$ s.t. $\mathrm{IP}^A \neq \mathrm{PSPACE}^A$ (moreover, the proof of equality appeared to relativize, prompting lots of debate). Hence the quantum result $\exists A$ s.t. $\mathrm{BQP}^A \neq \mathrm{NP}^A$ is nothing to get too excited about.

Proof (A) Let $A = TQBF$. Clearly, $\mathrm{P}^A \subseteq \mathrm{NP}^A$ for any oracle A. For the other direction, $\mathrm{NP}^{TQBF} \subseteq \mathrm{NPSPACE}^{TQBF} \subseteq \mathrm{NPSPACE}$ since we can compute instances of $TQBF$ in polynomial space rather than asking the oracle. From Savitch's theorem, NPSPACE $\subseteq$ PSPACE. And finally, since $TQBF$ is PSPACE-complete, we have $\mathrm{PSPACE} \subseteq \mathrm{P}^{TQBF}$. Hence $\mathrm{P}^{TQBF} = \mathrm{NP}^{TQBF}$.

(B) The idea is that nondeterminism gives us more powerful access to the oracle, allowing us to ask more questions than a deterministic TM can.

We need $L \in \mathrm{NP}^B$, but $L \notin \mathrm{P}^B$. We define $L(B)$ as follows:

$$L(B) = \{w | \exists x \in B : |x| = |w|\}$$

In other words, $L(B)$ is the language of words which are the same length of another word in B. We see that $L(B) \in \mathrm{NP}^B$ because we can

nondeterministically guess a word x such that $|x| = |w|$ and then check if it is in B.

We want to show that there is a B for which $L(B) \notin \mathrm{P}^B$, i.e. for every PTIME OTM $M^?$, $\exists x$ s.t. $M^B(x)$ rejects but $x \in L(B)$ or $M^B(x)$ accepts but $x \notin L(B)$. Our proof will be by diagonalization.

As a warm-up, we first show how to construct an oracle to defeat one specific OTM $M^?$ (i.e. $L(M^?) \neq L(B)$). Let $p(n)$ be a polynomial upper bound on the running time of $M^?$ on input 0^n. Then we can find n such that on input 0^n, $M^?$ runs in fewer than 2^n steps. Simulate $M^?$ on input 0^n, and answer "no" to all

queries (i.e. take $? = \emptyset$). If $M^?$ ends up accepting 0^n, simply let B be the empty language. That means $L(B)$ is also empty, but the language of $M^?$ contains 0^n.
On the other hand, if $M^?$ rejects 0^n, then let x be a word of length n for which $M^?$ did not query the oracle. Such a word must exist since $M^?$ ran in $p(n) < 2^n$ steps. So set $B = \{x\}$. That makes $0^n \in L(B)$, so once again $L(M^?) \neq L(B)$.

Now we need to show that there is a B such that for all polynomial time OTMs $M^?$, $L(M^B) \neq L(B)$. First we let M_1, M_2, etc. be an enumeration of polynomial time oracle TMs. We construct a sequence $B_0 \subseteq B_1 \subseteq \cdots \subseteq B$, with lengths $n_1 < n_2 < \cdots$, such that

1. $M_i^{B_i}(0^{n_i}) \neq (0^{n_i} \in L(B_i))$.

2. $M_i^{B_i}(0^{n_i}) = M_i^B(0^{n_i})$.

3. $0^{n_i} \in L(B_i) \Leftrightarrow 0^{n_i} \in L(B)$.

The construction algorithm is as follows:

1. First let $B_0 = \emptyset$.

2. For $i = 1, 2, 3, \ldots$ do

   (a) Choose n_i such that $\forall j < i$, $M_j^{B_{i-1}}$ does not ask about any strings of length n_i on input 0^{n_j}. In addition, $M_i^{B_{i-1}}$ should run for fewer than 2^{n_i} steps on input 0^{n_i}.

   (b) Simulate M_i with oracle B_{i-1} on input 0^{n_i}. (At this point, B_i has no strings of length n_i.) This will answer no to all queries of length n_i.

   (c) If $M_i^{B_{i-1}}$ accepts, set $B_i = B_{i-1}$. (So $M_i^{B_i}$ has accepted 0^{n_i}, but $0^{n_i} \notin L_{B_i}$).
   If $M_i^{B_{i-1}}$ rejects, then find x such that $|x| = n_i$ and $M_i^{B_{i-1}}$ on input 0^{n_i} did not ask x. Then set
   $$B_i = B_{i-1} \cup \{x\}.$$
   (So we have $M_i^{B_{i-1}}$ rejecting 0^{n_i}, but $0^{n_i} \in L(B)$, just as in the warm-up.)

Set $B = \bigcup_{i=0}^{\infty} B_i$. ■

One can similarly prove that there exists an oracle B such that $\mathrm{NP}^B \neq \mathrm{coNP}^B$.

2 Randomized Computation

Definition 2 (RP) $L \in \mathrm{RP}$ *(randomized polynomial time) if* $\exists$ *a randomized (can "toss coins" along the way) polynomial time TM* M *s.t.* $\forall x :$ $\begin{array}{l} x \in L \Rightarrow \Pr[M(x) \text{ accepts}] \geq \frac{2}{3} \\ x \notin L \Rightarrow \Pr[M(x) \text{ accepts}] = 0 \end{array}$.

For example, it is easy to show that Composites $\in$ RP via elementary number theory (Euler's theorem). It is also true, but nontrivial, that Primes $\in$ RP. Clearly RP $\subseteq$ NP (existentially guess every random choice).

Definition 3 (BPP) $L \in \mathrm{BPP}$ *(bounded (error) probabilistic polynomial time) if* $\exists$ *a randomized poly time TM* M *and a polynomial* $p(\cdot)$ *s.t.* $\forall x :$ $\begin{array}{l} x \in L \Rightarrow \Pr_{r \in \{0,1\}^{p(|x|)}}[M(r,x) \text{ accepts}] \geq \frac{2}{3} \\ x \notin L \Rightarrow \Pr_{r \in \{0,1\}^{p(|x|)}}[M(r,x) \text{ accepts}] \leq \frac{1}{3} \end{array}$.

The choice of $\frac{2}{3}$ and $\frac{1}{3}$ is somewhat arbitrary. See the amplification lemma in the next lecture. In the proof of the following proposition, we will show that the bounds can be replaced by $1 - 2^{-n-1}$ and 2^{-n-1}.

Proposition 4 RP $\subseteq$ BPP $\subseteq$ P/poly.

Proof The first containment is obvious. For the second, the idea is to use values of r as advice strings. Turn error into 2^{-n-1} for strings of length n.

Say we have M s.t. $\forall_{|x|=n} \Pr_{r \in \{0,1\}^{p(|x|)}}[M(r,x) \text{ is wrong}] < 2^{-n-1}$. There are 2^n x of length n, therefore

$\Pr_r[\exists x : |x| = n \text{ and } M(r,x) \text{ is wrong}] < 2^n \cdot 2^{-n-1} = \frac{1}{2} < 1 \Rightarrow \exists r$ s.t. $\forall |x| = n$, $x \in L \Leftrightarrow M(r,x)$ accepts (see probability handout). Build this r in as the advice string.

The only detail left is that we can get the error down to 2^{-n-1}. For this, we run the machine

kn times (k to be specified later), and accept if it accepts a majority of the time. The probability of getting the wrong answer is the probability of being wrong at least half of the time, or

$$\sum_{i=0}^{kn/2} \binom{kn}{i} \left(\frac{2}{3}\right)^i \left(\frac{1}{3}\right)^{kn-i} = \left(\frac{1}{3}\right)^{kn} \sum_{i=0}^{kn/2} \binom{kn}{i} 2^i.$$

Since the sum over all i of $\binom{kn}{i}$ is 2^{kn}, and each 2^i term is at most $2^{kn/2}$, we can easily bound this probability from above by $\left(\frac{1}{3}\right)^{kn} 2^{kn} 2^{kn/2} = \left(\frac{2\sqrt{2}}{3}\right)^{kn}$. Since $2\sqrt{2} < 3$, we can find k large enough so that this probability is less than 2^{-n-1}, as desired. ■