6.841/18.405J: Advanced Complexity Theory	February 27, 2002
Lecture 7	
Lecturer: Madhu Sudan	*Scribe: Deniss Čebikins*

1. Review of definitions

Recall the definitions of BPP and RP.

A language L is in BPP if and only if there exists a polynomial time Turing machine M and a polynomial p such that

$$\Pr_y[M(x,y) \text{ accepts}] \geq c = \frac{2}{3} \text{ for all } x \in \text{L}$$

$$\Pr_y[M(x,y) \text{ accepts}] \leq s = \frac{1}{3} \text{ for all } x \notin L$$

where $|y| = p(|x|)$.

A language L is in RP if and only if there exists a polynomial time Turing machine M and a polynomial p such that

$$\Pr_y[M(x,y) \text{ accepts}] \geq c = \frac{2}{3} \text{ for all } x \in L$$

$$\Pr_y[M(x,y) \text{ accepts}] = 0 \text{ for all } x \notin L$$

where $|y| = p(|x|)$.

Our goal is to show that the above definitions do not really depend on the constants $\frac{1}{3}$ and $\frac{2}{3}$. In fact, in the definition of BPP these constants can be replaced by any pair of constants (c, s) provided that $0 < s < c < 1$. Furthermore, we will show that instead of constants we can use functions $c(n) = 1 - e^{-q(n)}$ and $s(n) = e^{-q(n)}$, where $n = |x|$, and q is a polynomial.

2. Example: amplification of RP

We start with an example of amplification that shows that a language with the following properties is in RP.

Let L be a language for which there exists a polynomial time Turing machine M and a polynomial p such that

$$\Pr_y[M(x,y) \text{ accepts}] \geq \frac{1}{n^2} \text{ for all strings } x \in L \text{ of length } n$$

$$\Pr_y[M(x,y) \text{ accepts}] = 0 \text{ for all } x \notin L$$

where $|y| = p(|x|)$. To show that $L \in$ RP we will construct another Turing machine M' that satisfies the conditions of the definition of RP stated above.

<u>Lemma.</u> There exists a polynomial $m(n)$ such that $\left(1 - \frac{1}{n^2}\right)^{m(n)} \leq \frac{1}{3}$.

Proof. Put $m(n) = 2n^2$. Then $\left(1 - \frac{1}{n^2}\right)^{m(n)} \approx e^{-\frac{m(n)}{n^2}} = e^{-2} = 0.1353... < \frac{1}{3}$.

Let $M'(\cdot, \cdot)$ be a Turing machine with the following algorithm:

- M' takes as input a string x of length n and a sequence $\bar{y} = \{y_1, y_2, \ldots, y_{m(n)}\}$ of strings, where $|y_i| = p(n)$ and $m(n)$ is a polynomial such that $\left(1 - \frac{1}{n^2}\right)^{m(n)} \leq \frac{1}{3}$.
- For each i between 1 and $m(n)$ machine M' simulates M on input (x, y_i).

- M' accepts if and only if $M(x, y_i)$ has accepted for some i.

Clearly, M' is a polynomial time Turing machine, and $\bar{y}$ has polynomial length. Let us analyze the probability of "success" of M'. First of all, it is obvious that

$$\Pr_{\bar{y}}[M'(x, \bar{y}) \text{ accepts}] = 0 \text{ for all } x \notin L$$

since $M(x, y_i)$ will never accept if $x \notin L$. If $x \in L$, and y_1, y_2, ... , are chosen independently at random, then for each i the probability that $M(x, y_i)$ rejects is at most $1 - \frac{1}{n^2}$, therefore the probability that $M(x, y_i)$ rejects for all i is at most $\left(1 - \frac{1}{n^2}\right)^{m(n)}$, hence

$$\Pr_{\bar{y}}[M'(x, \bar{y}) \text{ accepts}] \geq 1 - \left(1 - \frac{1}{n^2}\right)^{m(n)} \geq \frac{2}{3}$$

It follows that $L \in \mathrm{RP}$.

3. Amplification of BPP

In this section we will show that for every language L in BPP and every polynomial q there exists a polynomial time Turing machine $M'(\cdot, \cdot)$ and a polynomial u such that

$$\Pr_{\bar{y}}[M'(x, \bar{y}) \text{ accepts}] \geq 1 - e^{-q(n)} \text{ for all strings } x \in L \text{ of length } n$$

$$\Pr_{\bar{y}}[M'(x, \bar{y}) \text{ accepts}] \leq e^{-q(n)} \text{ for all strings } x \notin L \text{ of length } n$$

where $|\bar{y}| = u(n)$.

Since $L \in \mathrm{BPP}$, there exist a polynomial time Turing machine M and a polynomial p such that

$$\Pr_{y}[M(x, y) \text{ accepts}] \geq c \text{ for all } x \in L$$

$$\Pr_{y}[M(x, y) \text{ accepts}] \leq s \text{ for all } x \notin L$$

where $|y| = p(|x|)$. To make our result stronger we will assume that c and s are arbitrary constants satisfying $0 < s < c < 1$ rather than $\frac{2}{3}$ and $\frac{1}{3}$.

As in the previous section, the input to the new machine M' will be a string x of length n and a sequence $\bar{y} = \{y_1, y_2, \ldots, y_{m(n)}\}$ of independently selected random strings such that $|y_i| = p(n)$ for all i. We will see how to choose the polynomial $m(n)$ later.

The algorithm of M' will be simple: for each i, simulate M on input (x, y_i) and accept if the fraction of i's in $\{1, 2, \ldots, m(n)\}$ for which $M(x, y_i)$ has accepted is at least $\frac{c+s}{2}$. (Thus, for example, if $s = 0.86$ and $c = 0.88$ then M' will accept if $M(x, y_i)$ accepts for at least 87% of indices i in the set $\{1, 2, \ldots, m(n)\}$.)

In our analysis of the probability of correctness of this algorithm we will use the following lemma.

Lemma (Chernoff bound). Let D be a distribution on $\{0, 1\}$. Suppose that $x_1, x_2, \ldots, x_N$ are chosen independently from D. Let $\mu = \mathrm{E}_{x \in D}[x]$. Then for any λ the following inequality holds:

$$\Pr\left[\left|\frac{\sum_{i=1}^{N} x_i}{N} - \mu\right| \geq \lambda\right] \leq e^{-\lambda^2 \cdot \frac{N}{2}}$$

We apply the lemma in the following way. For $1 \leq i \leq m(n)$, define

$$X_i = \begin{cases} 1 & \text{if } M(x, y_i) \text{ accepts} \\ 0 & \text{if } M(x, y_i) \text{ rejects} \end{cases}$$

First let us consider the case $x \in L$. Then $\mu = \frac{\mathrm{E}[X_i]}{m(n)} \geq c$. Put $\lambda = \frac{c-s}{2}$. Then

$$\Pr\left[\text{ the fraction of } i \text{ in } \{1, 2, \ldots, m(n)\} \text{ for which } M(x, y_i) \text{ accepts is smaller than } \tfrac{c+s}{2} \right] \leq$$

$$\leq \Pr\left[X_1 + X_2 + \cdots + X_{m(n)} \leq \left(\frac{c+s}{2}\right) \cdot m(n)\right]$$

$$\leq \Pr\left[\left|\frac{\sum_{i=1}^{m(n)} X_i}{m(n)} - \mu\right| \geq \frac{c-s}{2}\right] \leq e^{-\left(\frac{c-s}{2}\right)^2 \cdot \frac{m(n)}{2}}$$

since the distance from μ to $\left(\frac{c+s}{2}\right)$ is at least $\left(\frac{c-s}{2}\right)$. Therefore

$$\Pr_{\bar{y}}[M'(x, \bar{y}) \text{ accepts}] \geq 1 - e^{-\left(\frac{c-s}{2}\right)^2 \cdot \frac{m(n)}{2}}$$

Similarly one can show that if $x \notin L$ then

$$\Pr_{\bar{y}}[M'(x, \bar{y}) \text{ accepts}] \leq e^{-\left(\frac{c-s}{2}\right)^2 \cdot \frac{m(n)}{2}}$$

In order to finish the argument it remains to set $m(n) = \frac{2q(n)}{(c-s)^2}$.

Here is an application of this result:

<u>Proposition (Adelman).</u> BPP $\subseteq$ P/poly.

Proof. Suppose that L is in BPP. Then there exists a polynomial time randomized Turing machine M' such that

$$x \in L \implies M' \text{ accepts } x \text{ with probability } 1 - 2^{-(n+1)} \text{ or more}$$

$$x \notin L \implies M' \text{ accepts } x \text{ with probability } 2^{-(n+1)} \text{ or less}$$

We claim that given n, there exists a string y such that $M(x, y)$ accepts if and only if $x \in L$ for all $x \in \{0,1\}^n$. (Hence y is the advice to M corresponding to inputs of length n.)

Let us call a string y "bad" for $x \in \{0,1\}^n$ if x is in L and $M(x, y)$ rejects, or else if x is not in L and $M(x, y)$ accepts. We will also say that y is "good" for x if it is not "bad" for x. For any fixed x we have

$$\Pr_y[\ y \text{ is "bad" for } x\ ] \leq 2^{-(n+1)}$$

therefore

$$\Pr_y[\exists x \in \{0,1\}^n \mid y \text{ is "bad" for } x] \leq \sum_{x \in \{0,1\}^n} \Pr_y[\ y \text{ is "bad" for } x\ ] \leq 2^n \cdot 2^{-(n+1)} = \frac{1}{2}$$

so $\Pr_y[\ y$ is "good" for all $x\ ] \geq \frac{1}{2}$. The claim and hence the proposition follow.

4. BPP and the Polynomial Hierarchy

We will show in this section that BPP $\subseteq$ PH. In fact, we will prove that BPP $\subseteq \Sigma_2^{\mathrm{P}}$.

Let L be a language in BPP. To show that $L \in \Sigma_2^{\mathrm{P}}$ we can represent the process of deciding whether x is in L as a two round debate, in which Player 1 tries to prove that $x \in L$, and Player 2 tries to prove that $x \notin L$. Player 1 passes some information to Player 2, Player 2 then replies to Player 1, and after the discussion an independent "judge" decides the winner.

Since $L \in$ BPP, there exist a polynomial time Turing machine M and a polynomial p such that M takes as input a string x of length n and a random string $y \in \{0,1\}^{p(n)}$, and

$$x \in L \quad \Longrightarrow \quad M \text{ accepts with probability of more than } \tfrac{1}{2}$$

$$x \notin L \quad \Longrightarrow \quad M \text{ accepts with probability of less than } \tfrac{1}{2p(n)}$$

Let us fix $x \in \{0,1\}^n$. In the two round debate Player 1 will try to find a bijection $\pi : \{0,1\}^{p(n)} \to \{0,1\}^{p(n)}$ such that for every $y \in \{0,1\}^{p(n)}$ at least one of y, $\pi(y)$ is "good" for x (recall that y is "good" for x if $x \in L \Leftrightarrow M(x,y)$ accepts). Player 2 will attempt to prove by counterexample that the bijection specified by Player 1 does not satisfy the condition. In other words, Player 2 will try to find a string y such that both y and $\pi(y)$ are "bad" for x.

Notice that if $x \notin L$ then the fraction of "good" strings for x is too small for a satisfactory bijection π to exist.

Define $\pi_r(x) = x \oplus r$. (Here "$\oplus$" means XOR: for example, $01101 \oplus 10001 = 11100$.)

Since the description of a bijection between $\{0,1\}^{p(n)}$ and $\{0,1\}^{p(n)}$ is too long to be transmitted in polynomial time, let us consider the following debate scheme. Player 1 chooses $p(n)$ strings $r_1, r_2, \ldots, r_{p(n)}$ of length $p(n)$, Player 2 chooses a string $y \in \{0,1\}^{p(n)}$, and the "judge" decides the winner as follows: Player 1 wins (i.e. $x \in L$) if at least one of $M(x, \pi_y(r_i))$ accepts, otherwise Player 2 wins (i.e. $x \notin L$).

First let us show that if $x \in L$ then Player 1 can always choose $r_1, r_2, \ldots, r_{p(n)}$ such that the "judge" concludes that $x \in L$ no matter what y is produced by Player 2. We write

$$\Pr_{r_1,\ldots,r_{p(n)}}[M(x,\pi_y(r_i)) \text{ rejects}] < \frac{1}{2}$$

$$\Longrightarrow \quad \Pr_{r_1,\ldots,r_{p(n)}}[M(x,\pi_y(r_i)) \text{ rejects for all } 1 \le i \le p(n)] < 2^{-p(n)}$$

$$\Longrightarrow \quad \Pr_{r_1,\ldots,r_{p(n)}}[\exists y \in \{0,1\}^{p(n)} \text{ such that } \forall i \; M(x,\pi_y(r_i)) \text{ rejects}] < 1$$

$$\Longrightarrow \quad \Pr_{r_1,\ldots,r_{p(n)}}[\forall y \in \{0,1\}^{p(n)} \; \exists i \; M(x,\pi_y(r_i)) \text{ accepts}] > 0$$

The last inequality means that there exists a sequence $r_1, r_2, \ldots, r_{p(n)}$ such that for any string y the "judge" algorithm will conclude that $x \in L$.

Now suppose that $x \notin L$. In this case for any sequence $r_1, r_2, \ldots, r_{p(n)}$ generated by Player 1 there must exist a string y such that the "judge" algorithm concludes that $x \notin L$. We have

$$\Pr_y[M(x,\pi_y(r_i)) \text{ accepts}] \le \frac{1}{2p(n)} \qquad \Longrightarrow \qquad \Pr_y[\exists i \; M(x,\pi_y(r_i)) \text{ accepts}] \le p(n) \cdot \frac{1}{p(n)} = \frac{1}{2}$$

hence for any sequence $r_1, r_2, \ldots, r_{p(n)}$ there exists $y \in \{0,1\}^{p(n)}$ such that $M(x, \pi_y(r_i))$ rejects for all i, as desired.

The following algorithm with two alternations decides L: first nondeterministically select $r_1, r_2, \ldots, r_{p(n)}$, then verify that for all $y \in \{0,1\}^{p(n)}$ the "judge" algorithm determines that $x \in L$. Since it takes polynomial time to run the "judge" algorithm for a particular choice of $r_1, r_2, \ldots, r_{p(n)}$ and y, it follows that $L \in \Sigma_2^{\mathrm{P}}$.