Review: Covered definitions of RP, coRP, ZPP, BPP last class.

From last class:

**Definition 1** BPP is the class of languages $L$ such that $\exists$ PPT $M$ s.t.

- $x \in L \implies \Pr M(x) = 1 \geq \frac{2}{3}$ (or $\frac{1}{2} + \frac{1}{p(n)}$ or $1 - \frac{1}{2^n}$)
- $x \notin L \implies \Pr M(x) = 0 \geq \frac{2}{3}$ (or $\frac{1}{2} + \frac{1}{p(n)}$ or $1 - \frac{1}{2^n}$)

**Theorem 1** $\text{BPP} \subseteq \text{P/poly}$ [Adleman]

This implies that randomness does not provide much extra power, but we do not know if $\text{BPP} \subseteq \text{NP}$.

**Proof.** Consider $L \in \text{BPP}$. Let $M$ be a machine deciding $L$. Assume WLOG that $M$ makes an error with probability $\leq \frac{1}{2^n}$. For each $x$ of length $n$, only 1 in $2^{2n}$ random tapes are bad. The total fraction of tapes that are “bad” for at least one $x$ is $\leq 2^n \cdot 2^{2n} = 2^{-n}$. Therefore, there are a lot of strings which work for every $x$ of a given length, so that string can be the advice for a P/poly machine.

**Theorem 2** $\text{BPP} \subseteq \Sigma_2(\cap \Pi_2)$

**Proof.** Reminder: $\Sigma_2$ is the set of languages where $x \in L$ if $\exists y_1 \forall y_2 \ R(x, y_1, y_2)$.

Consider $L \in \text{BPP}$ and let $M$ be a machine deciding $L$ that on input of length $n$ uses $m(n)$ random bits where $m$ is a polynomial. Assume WLOG that $M$ makes an error with probability $< \frac{1}{m(n)}$. Note that we can get $< \frac{1}{2^n}$ error with polynomially many random bits so the probability bound is not a problem.

Idea: using different random bits, all of $x \in L$ will get covered by some random bits for which $M(x) = 1$, but not all of $x \notin L$ will be covered by some random bits for which $M(x) = 1$ because too few of them are erroneously marked as in $L$ by each selection of random bits.

Given input $x$, let $S_x$ denote the set of all random tapes $r$ for which $M_r(x) = 1$.

- If $x \in L$ \quad $|S_x| \geq (1 - \frac{1}{m})2^m$. 

10-1
• If $x \notin \mathcal{L}$ then $|S_x| < \frac{1}{m}2^m$.

Use $\text{xor}$ ($\oplus$) for the permutation. Recall that $|S_x \oplus z| = |S_x|$ for any $z \in \{0, 1\}^{m(n)}$.

**Proposition 1** If $|S_x| < \frac{1}{m(n)}2^{m(n)}$, $\forall z_1, \ldots, z_{m(n)} \in \{0, 1\}^{m(n)}$

$$
\bigcup_{i=1}^{m(n)} S_x \oplus z_i \subseteq \{0, 1\}^{m(n)}
$$

**Proof.**

$$
\left| \bigcup_{i=1}^{m(n)} S_x \oplus z_i \right| \leq \sum_{i=1}^{m(n)} |S_x \oplus z_i| \leq \sum_{i=1}^{m(n)} |S_x| = m(n)|S_x| < m(n)\frac{1}{m(n)}2^{m(n)} = 2^{m(n)}
$$

**Proposition 2** If $|S_x| \geq (1 - \frac{1}{m(n)})2^{m(n)}$, $\exists z_1, \ldots, z_{m(n)} \in \{0, 1\}^{m(n)}$

$$
\bigcup_{z_1, \ldots, z_{m(n)}} S_x \oplus z_i = \{0, 1\}^{m(n)}
$$

**Proof.** Consider $y \in \{0, 1\}^{m(n)}$.

$$
\Pr_{z_1, \ldots, z_m} \left[ y \notin \bigcup_{i=1}^{m(n)} S_x \oplus z_i \right] \leq \prod_{i=1}^{m(n)} \Pr_{z_i \in \{0, 1\}^{m(n)}} [y \notin S \oplus z_i] = \prod_{i=1}^{m(n)} \Pr_{z_i \in \{0, 1\}^{m(n)}} [z_i \notin S \oplus y] < \left(\frac{1}{m(n)}\right)^{m(n)}
$$

By the union bound, $\Pr_{z_1, \ldots, z_m} \left[ \exists y \notin \bigcup_{i=1}^{m(n)} S_x \oplus z_i \right] \leq 2^{m(n)}\left(\frac{1}{m(n)}\right)^{m(n)}$ because $y \in \{0, 1\}^{m(n)}$.

Taking the complement gives $\Pr_{z_1, \ldots, z_m} \left[ \bigcup_{i=1}^{m(n)} S_x \oplus z_i = \{0, 1\}^{m(n)} \right] \geq 1 - \frac{2^{m(n)}}{m(n)^{m(n)}} > 0$.

The probability that such a set of permutations $z_1, \ldots, z_{m(n)}$ exists is positive, so such a set exists.

Therefore, the problem can be written as a $\Sigma_2$ problem with the relation $R$ being the BPP machine $\mathcal{M}$ run with a specific random tape:

$x \in \mathcal{L}$ iff $\exists z_1, \ldots, z_m \bigcup_{i=1}^{m(n)} S_x \oplus z_i = \{0, 1\}^m$ \iff $x \in \mathcal{L}$ iff $\exists z_1, \ldots, z_m \forall y \in \{0, 1\}^{m(n)} \bigvee_{i=1}^{m(n)} \text{M}(x, z_i \oplus y)$

Recall that BPP is closed under complement so it is also in $\Pi_2$.

Open problem: $\text{BPTIME}(n^{10}) \leq \text{BPTIME}(n)$. No one knows if more polynomial time gives more power for BPP.

10-2
**Definition 2** A promise problem is a pair of languages $L_{yes}, L_{no}$ such that $L_{yes} \cap L_{no} = \emptyset$ and $M$ decides $(L_{yes}, L_{no})$ if when

- $x \in L_{yes}$  $M(x) = 1$
- $x \in L_{no}$  $M(x) = 0$

Open question: $NP \cap coNP = P \implies NP = P$

The promise version:

**Theorem 3** Promise $(NP \cap coNP) = P \implies NP = P$

**Proof.** There exists a complete promise problem:

$L_{yes} = \{f, g| f \in SAT, g \not\in SAT\}$, $L_{no} = \{f, g| f \not\in SAT, g \in SAT\}$

Assume $M$ decides $(L_{yes}, L_{no})$:

$M'(\Phi) : M(\Phi_{x_0=0}, \Phi_{x_0=1}) = 1$ then let $a_0 = 0$; otherwise $a_0 = 1$. Repeat for every variable. Finally output $\Phi(a_0, \ldots, a_n)$. This works because if both are satisfiable or both are not satisfiable, then the choice of the value of $a_i$ does not matter so the fact that the return value of $M$ is not defined does not matter. If $M$ is defined, then the choice of value for $a_0$ is the one for which $\Phi$ is satisfiable. Therefore, $M'(\Phi)$ solves $SAT$ in polynomial time, so $NP \subseteq P \implies NP = P$.  

10-3