1 Recap

We defined $\text{RP}$ as the class of languages accepted by PPT machine with one-sided error bounded below 1/3, $\text{BPP}$ with two-sided error with gap 1/3. $\text{RP}$ was shown to be robust in the following sense.

Define $\text{RP}_e$ such that $L \in \text{RP}_e$ if for some poly-time TM $M$ and random bits $y$,

\[
x \in L \Rightarrow \Pr[M(x, y) \text{ rejects}] \leq e(|x|)
\]
\[
x \notin L \Rightarrow \Pr[M(x, y) \text{ accepts}] = 0
\]

Then $\text{RP}_{1/poly(n)} = \text{RP} = \text{RP}_{1/2^{o(poly(n)}}$ (the two poly’s may be different polynomials), yet $\text{RP}_{1/2^n} = \text{NP}$.

We will see that $\text{BPP}$ is robust in the similar sense. Define $\text{BPP}_{c,s}$ such that $L \in \text{BPP}_{c,s}$ if for some poly-time TM $M$ and random bits $y$,

\[
x \in L \Rightarrow \Pr[M(x, y) \text{ accepts}] \geq c(|x|)
\]
\[
x \notin L \Rightarrow \Pr[M(x, y) \text{ accepts}] \leq s(|x|)
\]

Let us assume that, as often necessary, that $s$ is “nice”, ie fully time constructible.

(Quick note: If $c \leq s$ then $\text{BPP}_{c,s}$ would contain every language. While it is not required that $c(n) \geq 0.5$ and $s(n) \leq 0.5$, one can shift the probability by proper amount so that $c, s$ do straddle 0.5.)

2 Amplification for BPP

Using Chernoff bound we will see that $\text{BPP}_{f(n)+1/poly(n), f(n)-1/poly(n)} = \text{BPP} = \text{BPP}_{1-2^{-poly(n)}, 2^{-poly(n)}}$.

**Theorem 1** (Chernoff bound) Let $X_1, \ldots, X_n \in [0,1]$ be independent random variables and $X = \sum_i X_i/n$. Then $\Pr[|X - E[X]| \geq \epsilon] \leq e^{-(\epsilon^2 n)/2}$.

Suppose some poly-time TM $M$ places $L$ in $\text{BPP}_{f(n)+1/p(n), f(n)-1/p(n)}$ where $p$ is a polynomial, and $f$ a “nice” function. Intuitively if one runs $M$ for $k$ times (with different random bits) and output according to whether the average of $k$ answers exceeds $f(n)$, the error probability should decrease.

By how much? Let random variable $X_i$ denote the output of $i$th run. For $x \in L$, error occurs if $\sum_i X_i/k < f(n)$ i.e at least $p(n)$ off expectation, thus with probability $O(e^{-k/2(p(n))^2})$ by Chernoff bound. With $k$ polynomial in $n$, this can be as small as $2^{-\Theta(n)}$ for any polynomial $q$. Likewise for $x \notin L$.

Here we have used polynomially many more random bits to reduce error. Can we do with fewer? The state-of-art, using ideas from pseudorandomness (ie expanders), is that $O(k)$ extra random bits can reduce error from 1/3 to $2^{-k}$.

3 $\text{BPP} \subseteq \text{P/poly}$

In advice (ie non-uniform) classes, one piece of (short) advice is expected to help all $2^n$ computations on length $n$ input. This might seem weak at first, but often times randomization is not more powerful than non-uniformity. In particular Adleman showed that $\text{BPP} \subseteq \text{P/poly}$.

Suppose machine $M$ places $L$ in $\text{BPP}$ with error probability below $2^{-p(n)}$, $p(n) > n$ (okay due to amplification). Is there a random string $y$ good for all $2^n$ inputs of length $n$, ie $M(x, y) = L(x)$ for each $x \in \{0,1\}^n$? Indeed, for each $x$ only a $2^{-p(n)}$ fraction of all random strings are bad; summing over all $2^n$ possible $x$ this fraction is still below 1! Thus some advice works for all inputs as random tape.
4 \( \text{BPP} \subseteq \Sigma_2^p \cap \Pi_2^p \)

How about some uniform class upper bounding \( \text{BPP} \)? It is clear that \( \text{BPP} \subseteq \text{PSPACE} \); it is unclear how \( \text{BPP} \) is related to \( \text{NP} \). Nevertheless we can show something intermediate: \( \text{BPP} \subseteq \Sigma_2^p \). (Which implies \( \text{BPP} \subseteq \Sigma_2^p \cap \Pi_2^p \) as \( \text{BPP} \) is closed under complementation.)

As before, suppose some machine \( M \) places \( L \) in \( \text{BPP} \) with error probability below \( 2^{-n} \). Let \( x \) be a length \( n \) input, and \( M \) uses \( m \) random bits on \( x \).

(Note that letting \( \exists \)-player show a set of polynomially many strings good for \( x \), as evidence, is not enough. To decide \( L \) by a 2-round debate one must ensure some kind of “fairness”, eg say one might let \( \exists \)-player to produce first half bits of \( y \), \( \forall \)-player second half, and see if \( M(x, y) \) accepts. This is still too crude to work, but illustrates the point.)

The idea is we do let \( \exists \)-player show a set of polynomially many strings good for \( x \), and the \( \forall \)-player tries to find some bijection mapping all of them to strings bad for \( x \). The bijections allowed are \( \oplus \) for any \( y' \). Intuitively, for \( x \in L \) it is hard for \( \forall \)-player to come up with such bijection that works on all good strings, and for \( x \notin L \) it is easy (and “easy” in a stronger sense than it is hard in the \( x \in L \) case).

Formally, one claims

\[
L = \{ x : \exists y_1, \ldots, y_m \forall y' \left[ \bigvee_{1 \leq i \leq m} M(x, y_i \oplus y') = 1 \right] \}
\]

Proof. Suppose \( x \in L \). Imagine one picks \( y_1, \ldots, y_m \) at random. Probability that \( \bigwedge_i M(x, y_i \oplus y') = 0 \), for each \( y' \), is below \( 2^{-mn} \); union bound over all possible \( y' \) the probability is still below 1, ie some \( y_1, \ldots, y_m \) make this false for all \( y' \).

Now suppose \( x \notin L \). Imagine one picks \( y' \) at random. Probability that \( \bigvee_i M(x, y_i \oplus y') = 1 \), for each \( y_1, \ldots, y_m \), is at most \( m2^{-n} < 1 \), ie some \( y' \) makes this false for all \( y_1, \ldots, y_m \).

This very idea can also be used to show \( \text{promiseBPP} \subseteq \text{promiseRP} \) (ie if \( P = \text{promiseRP} \) then \( P = \text{promiseBPP} \)).

5 Next time

We will talk about promise problems, which arise naturally eg when \( \text{BPP} \) has no known complete problem (as we don’t know how to enumerate error-bounded PPTs, ie to verify error-bounded-ness) yet \( \text{promiseBPP} \) has complete problems (eg given input \((M, x)\), promised that \( M \) is indeed error bounded, does \( M(x) = 1 \)?)

We will talk about the complexity of \( \text{UNIQUE} – \text{SAT} \), ie \( \text{SAT} \) with the promise that the satisfying assignment is either unique or non-existing. Is \( \text{UNIQUE} – \text{SAT} \) hard? We shall see that \( \text{NP} \neq \text{RP} \Rightarrow \text{UNIQUE} – \text{SAT} \) hard. (This problem arises in cryptography, where we want a mapping easy to compute one-way, but hard to revert. The mapping certainly should be one-to-one, so \( \text{UNIQUE} – \text{SAT} \) may be a good candidate.)