CAS CS 535: Complexity Theory

Lecturer: Mark Bun Fall 2023

Lecture Notes 16: Error Reduction, $\mathbf{BPP} \subseteq \mathbf{P}_{/\mathbf{poly}}$, $\mathbf{BPP} \subseteq \mathbf{PH}$

Reading.

- Arora-Barak § 7.4-7.5

Last time: Probabilistic Classes, Concentration Inequalities

Definition 1. $L \in \mathbf{RP}$ if there exists a probabilistic TM running in time $\mathrm{poly}(n)$ such that

$$x \in L \implies \Pr[M(x) = 1] \geq 2/3$$
$$x \notin L \implies \Pr[M(x) = 1] = 0.$$

Definition 2. $L \in \mathbf{ZPP}$ if there exists a probabilistic TM running in expected time $\mathrm{poly}(n)$ such that

$$x \in L \implies \Pr[M(x) = 1] = 1$$
$$x \notin L \implies \Pr[M(x) = 1] = 0.$$

Markov's Inequality: For any random variable $X \geq 0$, and real number $t \geq 0$, we have

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.$$

Hoeffding's Inequality: Let $X_1, \dots, X_n$ be independent, $[0,1]$-valued random variables, and let $X = \sum_{i=1}^n X_i$. Then for all $t > 0$,

$$\Pr[|X - \mathbb{E}[X]| > t] \leq 2\exp\left(-\frac{2t^2}{n}\right).$$

1 Proof that $\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$

Proof that $\mathbf{ZPP} \subseteq \mathbf{RP} \cap \mathbf{coRP}$. Let $L \in \mathbf{ZPP}$ be decided by a zero-error PTM M running in expected time $p(n)$. We'll first show that $L \in \mathbf{RP}$. Here's the algorithm.
Algorithm $N(x)$
On input x:

1. Run M on x for $3p(|x|)$ steps.
2. If M accepted, accept. If M either rejected or did not yet halt, reject.

Runtime analysis: The algorithm always halts after $O(p(n))$ steps, so it has polynomial worst-case runtime.
Correctness analysis: First suppose $x \in L$. Then by our assumption on M, we have $\mathbb{E}[T_{M,x}] \leq p(|x|)$. By Markov, this implies

$$\Pr[T_{M,x} \geq 3p(|x|)] \leq \frac{1}{3}.$$

Hence, M halts on x with probability at least $2/3$, and if it halts, it correctly accepts by zero-error. Therefore, N accepts x with probability at least $2/3$.

On the other hand, if $x \notin L$, then with probability 1 we have that M either rejects (by zero error) or fails to halt within $3p(|x|)$ steps. In either case, N correctly rejects.

The proof for $\mathbf{ZPP} \subseteq \mathbf{coRP}$ is similar. The only change is that if M fails to halt, we *accept*.

Proof that $\mathbf{RP} \cap \mathbf{coRP} \subseteq \mathbf{ZPP}$. Let $L \in \mathbf{RP} \cap \mathbf{coRP}$ be decided by $p(n)$ time $\mathbf{RP}$ machine M_1 and $\mathbf{coRP}$ machine M_2. We'll argue that the following PTM decides L with zero error.

```
Algorithm $N(x)$
On input $x$
    Repeat the following indefinitely:
    1. Run $M_1$ on $x$. If it accepts, accept.
    2. Run $M_2$ on $x$. If it rejects, reject.
```

I'll leave the proof of correctness and polynomial expected runtime for your homework.

2 Error Reduction

The constant $2/3$ in the definitions of $\mathbf{RP}$, $\mathbf{coRP}$, and $\mathbf{BPP}$ may seem arbitrary – and that's because they are. These classes don't change when $2/3$ is replaced by any positive constant, in the case of $\mathbf{RP}$ and $\mathbf{coRP}$, or any constant $> 1/2$ in the case of $\mathbf{BPP}$.

In fact, something stronger is true. The definition of $\mathbf{BPP}$ doesn't change even when we replace the error threshold with $1/2 + 1/\text{poly}(n)$ or with $1 - 2^{-\text{poly}(n)}$.

Theorem 3 (Error Reduction for $\mathbf{BPP}$)**.** *Suppose M is a PTM running in time $T(n)$ such that*

$$x \in L \implies \Pr[M(x) = 1] \geq \frac{1}{2} + \varepsilon$$
$$x \notin L \implies \Pr[M(x) = 1] \leq \frac{1}{2} - \varepsilon.$$

We'll abuse notation a bit and write these conditions together as "For all x, we have $\Pr[M(x) = L(x)] \geq \frac{1}{2} + \varepsilon$."

Then there exists a PTM M' running in time $O(\frac{\log(1/\delta)}{\varepsilon^2} \cdot T(n))$ such that $\Pr[M'(x) = L(x)] \geq 1 - \delta$.

That is, one can start with a PTM that beats random guessing by with advantage ε, and boost its success probability to $1 - \delta$, all with a modest blowup in runtime.

Before proving this, let's see some examples of how to use it.

If I start with $\Pr[M(x) = L(x)] \geq$ _____	Then I get $\Pr[M(x) = L(x)] \geq$ _____	Using _____ repetitions of M
$\frac{1}{2} + \frac{1}{n^2}$	$\frac{2}{3}$	$O\left(\frac{\log 3}{(1/n^2)^2}\right) = O(n^4)$
$2/3$	$1 - \frac{1}{n^2}$	$O(\log n)$
$2/3$	$1 - 2^{-10n}$	$O(n)$
$\frac{1}{2} + \frac{1}{n^{100}}$	$1 - 2^{-n^{-100}}$	$O(n^{300})$
$\frac{1}{2} + 2^{-n}$	$1 - 2^{-n}$	$O(n \cdot 2^{2n})$

Proof. Define the algorithm $M'(x)$ as follows.
On input x:

1. Run $M(x)$ independently $k = \frac{\log(2/\delta)}{\varepsilon^2}$ times, producing outputs $b_1, \ldots, b_k$.
2. Out the majority vote of those outputs.

By construction, this has the stated runtime.
To analyze correctness, define the random variables

$$X_i = \begin{cases} 1 & \text{if } b_i = L(x) \\ 0 & \text{otherwise.} \end{cases}$$

Why are these r.v.'s useful? Note that

- Our algorithm outputs the majority vote of the b_i's, which agree with L iff the majority of the X_i's are 1. In other words, our algorithm makes an error iff $X < k/2$ where $X = \sum_{i=1}^{k} X_i$.
- Each X_i individually takes the value 1 with some advantage over random guessing, i.e., $\mathbb{E}[X_i] \geq 1/2 + \varepsilon$. So $\mathbb{E}[X] \geq k/2 + \varepsilon k$.

Putting these observations together and applying Hoeffding's inequality, we have

$$\begin{aligned} \Pr[M'(x) \neq L(x)] &= \Pr[X < k/2] \\ &\leq \Pr[|X - \mathbb{E}[X]| > \varepsilon k] \\ &\leq 2\exp\left(-\frac{2(\varepsilon k)^2}{k}\right) \leq \delta. \end{aligned}$$

□

Theorem 4 (Error Reduction for **RP**). *If M is a PTM deciding L with one-sided error, i.e.,*

$$\begin{aligned} x \in L &\implies \Pr[M(x) = 1] \geq \varepsilon \\ x \notin L &\implies \Pr[M(x) = 1] = 0, \end{aligned}$$

then the PTM M' obtained by repeating M independently $k = O(\log(1/\delta)/\varepsilon)$ times and accepting if at least one run accepts has one-sided error, and $x \in L \implies \Pr[M'(x) = 1] \geq 1 - \delta$.

3 $\mathbf{BPP} \subseteq \mathbf{P}_{/\mathbf{poly}}$

One of the major aims of complexity theory is to understand the relationship between computational resources. We'll prove a few results which shed light on how randomness can be traded for other resources, like advice and alternations.

The first result we'll prove shows that randomized algorithms can be derandomized using nonuniform advice.

Theorem 5. $\mathbf{BPP} \subseteq \mathbf{P}_{/\mathbf{poly}}$.

To prove this, let $L \in \mathbf{BPP}$. First, we'll state the following useful characterization of $\mathbf{BPP}$ languages in terms of deterministic TMs operating with an additional random input.

Claim 6. *A language $L \in \mathbf{BPP}$ iff there exists a deterministic 2-input TM M running in polynomial time, and a polynomial $p(n)$, such that*

$$x \in L \iff \Pr_{r \leftarrow \{0,1\}^{p(|x|)}}[M(x,r) = L(x)] \geq 2/3.$$

The proof of this claim is similar to our proof that $\mathbf{NP}$ is characterized by polynomial-time certificate verifiers. The difference is that this time, we interpret the (random) string r as encoding the sequence of random transitions taken by a PTM.

The first thing we'll do is error reduction, to improve the success probability to be exponentially close to 1. That is, there exists a poly-time TM M' such that

$$x \in L \iff \Pr_r[M'(x,r) \neq L(x)] \leq 2^{-2n}.$$

Now we'll use the fact that we can convert poly-time deterministic TMs into poly-size circuits to obtain

$$x \in L \iff \Pr_{r \leftarrow \{0,1\}^m}[C(x,r) \neq L(x)] \leq 2^{-2n}$$

for some $m = \text{poly}(n)$ and some circuit family where $|C| = \text{poly}(n)$.

Now we'll get rid of the randomness in the circuit family with the following claim.

Claim 7. *For every n there exists an $r^* \in \{0,1\}^m$ such that $C(x, r^*) = L(x)$ for $\underline{\text{every}}$ $x \in \{0,1\}^m$*

Our final circuit family is obtained by "hardwiring" this good choice of r^* into C, i.e., by defining $C^*(x) = C(x, r^*)$.

Proof of Claim. We prove this by the "probabilistic method": We'll show that a random choice of r works positive probability, which in particular, shows that such an r exists.

Letting $r \leftarrow \{0,1\}^m$ be uniformly random, define $B_{x,r}$ to be the "bad" event that $C(x,r) \neq L(x)$. Then by definition, for all $x \in \{0,1\}^n$, we have $\Pr_r[B_{x,r}] \leq 2^{-2n}$.

Thus, the probability that r satisfies the claim is

$$\begin{aligned}
\Pr_r[\forall x C(x,r) = L(x)] &= 1 - \Pr_r[\exists x C(x,r) \neq L(x)] \\
&= 1 - \Pr_r\left[\bigcup_{x\in\{0,1\}^n} B_{x,r}\right] \\
&\geq 1 - \sum_{x\in\{0,1\}^n} \Pr_r[B_{x,r}] \qquad \text{(union bound)} \\
&\geq 1 - 2^n \cdot 2^{-2n} \\
&= 1 - 2^{-2n} > 0.
\end{aligned}$$

□

This result has some interesting consequences for understanding the relationship between $\mathbf{NP}$ and $\mathbf{BPP}$. Observe that if SAT had a fast randomized algorithm, i.e., SAT $\in \mathbf{BPP}$, then we would have SAT $\in \mathbf{P_{/poly}}$, which by the Karp-Lipton Theorem, would imply that $\mathbf{PH}$ collapses. So this gives us evidence that SAT does not, in fact, have a fast randomized algorithm.

4 BPP and the Polynomial Hierarchy

It seems unlikely that $\mathbf{NP} \subseteq \mathbf{BPP}$, but what about the other direction? We know that $\mathbf{RP} \subseteq \mathbf{NP}$, but the power of two-sided error doesn't immediately appear compatible with just nondeterministic guessing. We don't yet know whether $\mathbf{BPP}$ is contained in $\mathbf{NP}$, but we can at least place it in the polynomial hierarchy.

Theorem 8 (Sipser-Gács-Lautemann)**.** $\mathbf{BPP} \subseteq \mathbf{\Sigma}_2^p \cap \mathbf{\Pi}_2^p$.

Proof. Since $\mathbf{BPP}$ is closed under complement, it suffices to show that $\mathbf{BPP} \subseteq \mathbf{\Sigma}_2^p$. Using error reduction, if $L \in \mathbf{BPP}$, then there is a poly-time TM M such that

$$x \in L \iff \Pr_{r\in\{0,1\}^m}[M(x,r) = L(x)] \geq 1 - 2^{-n}$$

for some $m = \mathrm{poly}(n)$.

For each $x \in \{0,1\}^n$, define the set

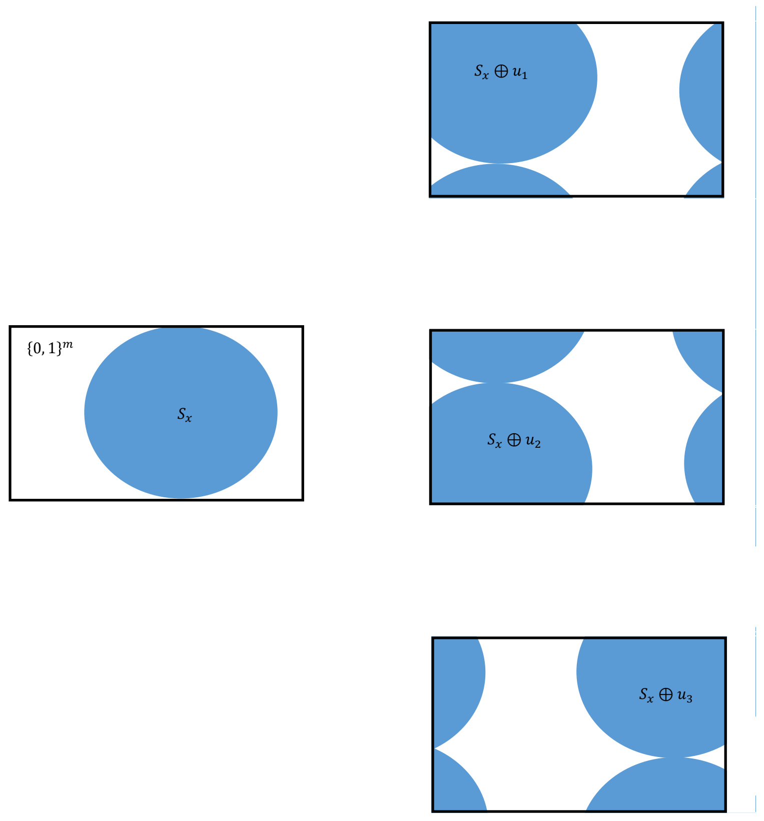
$$S_x = \{r \mid M(x,r) = 1\}$$

to be the set of coin tosses that cause M to accept input x. Then we have

$$\begin{aligned}
x \in L &\implies \frac{|S_x|}{2^m} \geq (1 - 2^{-n}) \\
x \notin L &\implies \frac{|S_x|}{2^m} \leq 2^{-n}
\end{aligned}$$

Our goal will be to use two quantifiers to distinguish between these cases, i.e., to check whether S_x is huge (almost everything) or tiny (almost nothing).

The idea is as follows. If S_x, then there will *exist* a small number of "additive shifts" of S_x whose union cover *all* of the space $\{0,1\}^m$.



Meanwhile, if S_x is tiny, then every small number of shifts will still fail to cover the whole space. Here are the details. First, let us define what we mean by additive shifts.

Definition 9. For $S \subseteq \{0,1\}^m, u \in \{0,1\}^m$, let $S \oplus u = \{z \oplus u \mid z \in S\}$, where $\oplus$ denotes the bitwise XOR.

Let $k = 2m/n = \text{poly}(n)$. To distinguish our two cases, our goal is to prove the following two lemmas.

Lemma 10. *If $|S_x|/2^m \geq 1 - 2^{-n}$, then there exist shifts $u_1, \ldots, u_k \in \{0,1\}^m$ such that $\bigcup_{i=1}^k (S_x \oplus u_i) = \{0,1\}^m$.*

Lemma 11. *If $|S_x|/2^m \leq 2^{-n}$, then for all shifts $u_1, \ldots, u_k \in \{0,1\}^m$, we have $\bigcup_{i=1}^k (S_x \oplus u_i) \subsetneq \{0,1\}^m$.*

Assuming these lemmas, let us now see how we can distinguish our two cases using two quantifiers.

Starting with the YES case, we have by Lemma 10 that

$$\begin{aligned}
x \in L &\implies |S_x|/2^m \geq 1 - 2^{-n} \\
&\implies \exists u_1, \ldots, u_k \in \{0,1\}^m \text{ s.t. } \bigcup_{i=1}^{k} (S_x \oplus u_i) = \{0,1\}^m \\
&\implies \exists u_1, \ldots, u_k \in \{0,1\}^m \forall r \in \{0,1\}^m \qquad r \in \bigcup_{i=1}^{k} (S_x \oplus u_i) \\
&\implies \exists u_1, \ldots, u_k \in \{0,1\}^m \forall r \in \{0,1\}^m \quad (\exists i \in [k]\ r \oplus u_i \in S_x) \\
&\implies \exists u_1, \ldots, u_k \in \{0,1\}^m \forall r \in \{0,1\}^m (\exists i \in [k]\ M(x, r \oplus u_i) = 1).
\end{aligned}$$

Similarly, in the NO case, we have by Lemma 11 that

$$x \notin L \implies \neg \exists u_1, \ldots, u_k \in \{0,1\}^m \forall r \in \{0,1\}^m (\exists i \in [k]\ M(x, r \oplus u_i) = 1).$$

Thus, we can distinguish the two cases using a poly-length round of existential guessing, followed by a poly-length round of universal guessing, and evaluation of the poly-time computable predicate "$\exists i \in [k]\ M(x, r \oplus u_i) = 1$." □

Now let's prove the lemmas.

Proof of Lemma 11. It suffices to show that if $|S|/2^m \leq 2^{-n}$, then for all sequences of k bitstrings $u_1, \ldots, u_k \in \{0,1\}^m$, where $k = 2m/n$, we have that

$$\left| \bigcup_{i=1}^{k} S \oplus u_i \right| < 2^m.$$

to do this, we use the union bound to upper bound the left hand side via

$$\sum_{i=1}^{k} |S \oplus u_i| = k|S| = \frac{2m}{n} \cdot 2^{-n} \cdot 2^m < 2^m$$

for sufficiently large n, since $m = \text{poly}(n)$. □

Proof of Lemma 10. Now we want to show that if $|S|/2^m \geq 1 - 2^{-n}$, there exist $u_1, \ldots, u_k$ such that $\bigcup_{i=1}^{k} S \oplus u_i = \{0,1\}^m$. We'll do this via the probabilistic method, showing that a random choice of the u's works with positive probability.

Let us choose $u_1, \ldots, u_k \in \{0,1\}^m$ uniformly at random. It now suffices to show that

$$\Pr_u \left[\bigcup_{i=1}^{k} S \oplus u_i = \{0,1\}^m \right] > 0$$

or equivalently, that

$$\Pr \left[\exists r \in \{0,1\}^m \text{ s.t. } r \notin \bigcup_{i=1}^{k} S \oplus u_i \right] < 1.$$

We bound the left hand side using a union bound by

$$\begin{aligned}
\sum_{r\in\{0,1\}^m} \Pr\left[r \notin \bigcup_{i=1}^{k} S \oplus u_i\right] &= \sum_{r\in\{0,1\}^m} \Pr[r \notin S \oplus u_1] \cdot \Pr[r \notin S \oplus u_2] \cdot \ldots \cdot \Pr[r \notin S \oplus u_k] \quad \text{by independence} \\
&= \sum_{r\in\{0,1\}^m} \prod_{i=1}^{k} \Pr[r \notin S \oplus u_i] \\
&\leq \sum_{r\in\{0,1\}^m} (2^{-n})^k \\
&\leq 2^m \cdot 2^{-n\cdot 2m/n} \\
&= 2^{-m} < 1.
\end{aligned}$$

□