RANDOMIZED COMPUTATION, CONTINUED

Last time

Probabilistic Turing machine (PTM)

Two transition functions δ_0, δ_1

At each step, pick transition δ_i, $i \in \{0,1\}$ with probability $1/2$

$L \in$ **BPP** if exists probabilistic Turing machine that

1. Always runs in polynomial time for all x (regardless of random choices)
2. $$\Pr_{\text{random choices}}\left[M(x) = L(x) \right] \geq 2/3 \quad \text{for all } x$$

Can also think of this as standard deterministic Turing machine with pre-filled random tape

Important: If $L \in BPP$ as shown by PTM M, then for all x either

a) $2/3 \leq \Pr[M(x)=1] \leq 1$, if $x \in L$

b) $0 \leq \Pr[M(x)=1] \leq 1/3$, if $x \notin L$

Never $1/3 < \Pr[M(x)=1] < 2/3$

Widely believed that $P = BPP$

But there are problems for which we have polynomial-time algorithms with randomness but not without (as of yet)

POLYNOMIAL IDENTITY TESTING (PIT)

Input polynomial represented as algebraic circuit

Output Is the polynomial identically zero?

DEMILLO-LIPTON-SCHWARZ-ZIPPEL LEMMA

$p(x_1, \ldots, x_n)$ non-zero multivariate polynomial of degree $\leq d$

$S \subseteq \mathbb{Z}$ finite set of integers

Then for uniformly chosen $(a_1, \ldots, a_n) \in S^n$ it holds that

$$\Pr_{\vec{a} \in S^n}\left[p(a_1, \ldots, a_n) \neq 0 \right] \geq 1 - \frac{d}{|S|}$$

RANDOMIZED PIT ALGORITHM

Let m = size of input circuit C

Let $S = \{1, 2, \ldots, 10 \cdot 2^m\}$

Sample uniformly $(a_1, \ldots, a_n) \in_R S^n$

Evaluate $C(a_1, \ldots, a_n) = val$

If $val \neq 0$ answer "polynomial not zero"
else answer "polynomial identically zero"

By DLSZ lemma, answer when $val = 0$ correct with 90% probability

answer when $val \neq 0$ is correct with 100% probability

DONE ☺ Except: Can circuit evaluation be carried out in polynomial time?

NOT NECESSARILY ...

Problem If degree $\approx 2^m$, then intermediate numbers can grow as large as $(10 \cdot 2^m)^{2^m} \Rightarrow$ exponentially many bits

Cannot compute with such numbers in polynomial time...

Solution "FINGERPRINTING"

Compute modulo some $k \in [2^{2m}]$

After each operation, divide by k and take remainder

Suppose polynomial computed by circuit $C(x_1, \ldots, x_n)$

Suppose $A = C(a_1, \ldots, a_n)$

If $A = 0$, then clearly $A \equiv 0 \pmod{k}$

CLAIM 5 If $A \neq 0$, then for a randomly chosen $k \in [2^{2m}]$ it holds that $k \nmid A$ with probability at least $\frac{1}{8m}$

Given this claim run test $K \cdot 8m$ times for suitably large constant K with different moduli k.

Suppose $A \neq 0$. Then test fails only if $k \mid A$

$$\Pr[k \mid A : A \neq 0] \leq 1 - \frac{1}{8m}$$

$$1 - x \leq e^{-x}$$

$$\Pr[\text{all } K \cdot 8m \text{ tests fail}] \leq$$

$$\left(1 - \frac{1}{8m}\right)^{K \cdot 8m} \leq$$

$$\left(e^{-\frac{1}{8m}}\right)^{K \cdot 8m} = e^{-K}$$

$$\Pr[\text{randomized PIT fails}] \leq$$

$$\Pr[A = 0] + \Pr[A \equiv 0 \bmod k_i \text{ for all } k_i]$$

$$\leq \frac{1}{10} + e^{-K} \leq \frac{1}{3}$$

for K chosen large enough. ∎

Proof of Claim 5

Assume $A \neq 0$ We know $A \leq (10 \cdot 2^m)^{2^m}$

Let B = prime factors of A

Sufficient to show that with probability $\geq \delta = \frac{1}{8m}$ k is a prime not in B

A has at most $\log A \leq 2^m \log(10 \cdot 2^m) \leq$

$\leq 5m \cdot 2^m$ prime factors

By Prime Number Theorem

$$\#\text{primes} \leq N \sim N / \ln N$$

Correct constant is actually 1, but see Thm A.23 in Arora-Barak for simpler version that is also sufficient

$$\#\text{primes} \le 2^{2m} \sim \frac{2^{2m}}{2m} > \frac{2^{2m}}{4m}$$

for large enough m

$$5m \cdot 2^m = o\left(\frac{2^{2m}}{2m}\right) < \frac{2^{2m}}{8m}$$

for large enough m.

So $\Pr[\text{k prime not in } B] \ge \dfrac{2^{2m}/8m}{2^{2m}}$

$$= \frac{1}{8m}$$ ▨

Many natural randomized algorithms have **ONE-SIDED ERROR**

Might make mistake when $x \in L$ but never when $x \notin L$, or other way round

(We just saw an algorithm that is always correct when $x \in L$, though not always when $x \notin L$)

DEF 6 **RTIME($T(n)$)** contains every language L for which exists probabilistic TM M running in time $O(T(n))$ such that

$$x \in L \Rightarrow \Pr[M(x)=1] \ge 2/3$$
$$x \notin L \Rightarrow \Pr[M(x)=0] = 1$$

RP $= \bigcup_{c \in \mathbb{N}^+} \text{RTIME}(n^c)$

Never false positives Positive answers Right

OBS 7 $RP \subseteq NP$

Proof Every accepting computation is a certificate, since if $x \in L$ we know for sure that $M(x) = 0$. So just specify sequence of random choices that make M accept and check that this indeed yields accepting path ∎

We do NOT know if $BPP \subseteq NP$
(but suspect so, since we believe $P = BPP$)

$$\boxed{coRP = \{ L \mid \overline{L} \in RP \}}$$

"Never false negatives"

Our randomized algorithm for polynomial identity testing showed that $ZEROP \in coRP$

So far, we demanded that our probabilistic Turing machines should always terminate within given time bound

Given general probabilistic Turing machine M, can define random variable

$$\boxed{T_M(x)} = \text{running time of } M \text{ on } x$$

Could be integer or $+\infty$

We can then take the EXPECTATION

$$\mathcal{E}\left[T_M(x)\right] = \sum_{t=0}^{\infty} t \cdot \Pr\left[T_M(x) = t\right]$$

of $T_M(x)$, if this expectation is finite

Note that expectation can be finite even if M could run forever

Say that M has EXPECTED RUNNING TIME $T(n)$ if for all $x \in \Sigma^*$

$$\mathcal{E}\left[T_M(x)\right] \leq T(|x|)$$

DEF 8 $\mathcal{ZTIME}(T(n))$ contains all languages L for which there exists a probabilistic Turing machine M that

(a) M runs in expected time $O(T(n))$

(b) $\Pr\left[M(x) = L(x) \mid M \text{ halts}\right] = 1$

$\mathcal{ZPP} = \bigcup_{c \in \mathbb{N}^+} \mathcal{ZTIME}(n^c)$

"Zero-sided error"

$\mathcal{ZPP}$ zero-error probabilistic polynomial time

"LAS VEGAS ALGORITHMS"

$\mathcal{BPP}$ "MONTE CARLO ALGORITHMS"

THM 9 $\mathcal{ZPP} = \mathcal{RP} \cap co\mathcal{RP}$

Proof Very useful exercise

We should also mention that it is immediately clear from the definitions that

$$\boxed{\begin{aligned} RP &\subseteq BPP \\ co\,RP &\subseteq BPP \end{aligned}}$$

ROBUSTNESS OF DEFINITION OF BPP

(a) Error probability: constant 2/3 is arbitrary
(b) Could use expected running time instead of worst-case
(c) Biased coins would be possible
(d) Could even use imperfect random sources (so-called "weak random sources")

We will show (a) — see Section 7.4 in Arora-Barak for the rest

THM 10 For any function $f: \mathbb{N} \to [0,1]$, let $\boxed{BPP_{f(n)}}$ denote the class of languages L for which there exists a polynomial-time probabilistic Turing machine M such that for all $x \in \Sigma^*$ it holds that

$$\boxed{P_r\left[M(x) = L(x)\right] \geq f(n)}$$

Then for all $c, d \in \mathbb{R}^+$ it holds that

$$\boxed{BPP_{\frac{1}{2}+n^{-c}} = BPP = BPP_{1-2^{-n^d}}}$$

This is referred to as $\boxed{\text{ERROR REDUCTION}}$ for BPP

Proof clearly $BPP_{1-2^{-n^d}} \subseteq BPP \subseteq BPP_{\frac{1}{2}+n^{-c}}$

We just need to show that

$$\boxed{BPP_{\frac{1}{2}+n^{-c}} \subseteq BPP_{1-2^{-n^d}}}$$

That is, given probabilistic Turing machine M with success probability $\geq \frac{1}{2} + |x|^{-c}$

for any input x, build PTM M' with success probability $\geq 1 - 2^{-|x|^d}$

M' will simply run M $k = 8|x|^{2c+d}$ times collect the answers, and go with the majority vote

How confident can we be that this is correct? Use material from Appendices A.2.1 and A.2.4 in Arora–Barak

Let

$$x_i = \begin{cases} 1 & \text{if } i\text{th run of } M \text{ yields correct answer} \\ 0 & \text{otherwise} \end{cases}$$

$$Pr[x_i = 1] = p \quad \text{for} \quad p \geq \frac{1}{2} + |x|^{-c}$$

In what follows, suppose $p = \frac{1}{2} + |x|^{-c}$ for simplicity in our calculations

$$\mathbb{E}[x_i] = p$$

By linearity of expectation

$$\mathbb{E}\left[\sum_{i=1}^{k} x_i\right] = kp$$

$$kp = \underbrace{\frac{8|x|^{2c+d}}{2}}_{\text{half of runs}} + \underbrace{8|x|^{c+d}}_{\text{expected extra margin}}$$

If you repeat independent trials sufficiently many times, then you will get very close to the expected value with very high probability

Example Fair coinflips

# coin flips	outcome
10	less than 4 heads or more than 6 would not be outrageous
100	Expect between 40 & 60 heads
1000	Will definitely see between 400 and 600 heads
10 000	Bet your house on between 4000 and 6000 heads

More formally:

deviation from expected value

LEMMA 11 (CHERNOFF BOUND)

For any constant $\delta > 0$, and for independent random variables $X_i \in \{0,1\}$ with $Pr[X_i = 1] = p$

$$Pr\left[\left|\sum_{i=1}^{k} X_i - pk\right| > \delta pk\right] < \exp\left(-\frac{\delta^2}{4} pk\right)$$

Plug in $p = \frac{1}{2} + |x|^{-c}$

$$\delta = \frac{|x|^{-c}}{2}$$

The majority vote will be correct unless

$$\sum_{i=1}^{k} x_i < pk - \delta pk$$

The probability of this happening is bounded by

$$\exp\left(-\frac{1}{4|x|^{2c}} \cdot \frac{8|x|^{2c+d}}{2}\right) =$$

$$\exp\left(-|x|^{d}\right) < 2^{-|x|^{d}}$$ ▨

How is BPP related to other complexity classes we have seen?

THM 12 $\boxed{BPP \subseteq P/poly}$

THM 13 $\boxed{BPP \subseteq \Sigma_2^P \cap \Pi_2^P}$ $(\subseteq PH)$

Both proofs use error reduction as in Thm 10 plus some other ideas.

Proof of Thm 13 is extremely neat, but we don't have time...

Try to sketch proof of Thm 12

Proof sketch for Thm 12

Suppose $\Sigma = \{0,1\}$ for simplicity

If $L \in BPP$, then by Thm 10 (and Prop 3) in last lecture there exists a probabilistic Turing machine M that on input size n

- uses m random bits
- runs in polynomial time
- gets the answer right except with probability $2^{-(n+1)}$

which can be converted to a deterministic TM M' with a random tape of m bits and the same guarantees

For a string of bits $r \in \{0,1\}^m$, say that r is BAD for input x if

$$M'(x,r) \neq L(x)$$

For every x, M' succeeds with probability $\geq 1 - 2^{-(n+1)}$

$\Rightarrow$ out of 2^m random strings, $\leq \frac{2^m}{2^{n+1}}$ strings are bad for x

Count the total # bad strings

$$\left|\{r \mid r \text{ bad for some } x\}\right| \leq \sum_{x \in \{0,1\}^n} \left|\{r \mid r \text{ bad for } x\}\right|$$

$$\leq 2^n \cdot \frac{2^m}{2^{n+1}} = \frac{2^m}{2}$$

But this means that there is at least one random string $r^* \in \{0,1\}^m$ (in fact, at least half of the strings) that are good for all inputs x of length n

Run M' with string r^* hardwired as advice for inputs of length n.

Since all Turing machines involved have polynomial running time, we have

$$m = \text{poly}(n)$$

What about complete problems for BPP?

A typical complete problem looks like:

"There exists a Turing machine of correct type running within resource bound such-and-such"

"Correct type" deterministic TM - simple syntactic criterion

nondeterministic TM - ditto

BPP-type: For every x, M accepts x with probability $p \geq 2/3$ or $p \leq 1/3$ but never $1/3 < p < 2/3$

Undecidable to check this condition

What about **hierarchy theorems**?
Fail for similar reasons.

A final useful notion: **RANDOMIZED REDUCTIONS**

DEF 14 A language B reduces to another language C under RANDOMIZED REDUCTIONS, denoted $B \leq_r C$, if there exists a polynomial-time probabilistic Turing machine M such that for all $x \in \Sigma^*$

$$\Pr\left[C(M(x)) = B(x) \right] \geq \frac{2}{3}$$

Not necessarily a transitive relation.

But if $C \in BPP$ and $B \leq_r C$, then $B \in BPP$

We could have defined NP in terms of randomized reductions instead, if we would have considered BPP a better formalization of "efficient computation"

$$NP = \{ L \mid L \leq_p 3\text{-SAT} \}$$

DEF 15 $$BP \cdot NP = \{ L \mid L \leq_r 3\text{-SAT} \}$$