CSC2401 - Introduction to Computational Complexity
Lecture #24, Thursday October 18 2001
Lecturer: Allan Borodin
Scribe: Denilson Barbosa

1 Randomized Complexity Classes

The complexity classes BPP, RP and ZPP are defined in terms of *probabilistic Turing Machines* (PTMs). We define a 2-tape PTM where the second tape is a *random-bit* tape, written by a fair random number generator ϕ. At each step the machine reads a new bit from that tape and uses this bit to (deterministically) decide which state to move next.

Definition 1 *A probabilistic 2-tape Turing Machine $\mathcal{M}$ is defined by $(Q, \Sigma, q_0, F, \delta)$, similarly to a DTM (recall Lecture 2), with the following extra conditions:*

- $\{0, 1\} \subseteq \Sigma$
- *δ is defined from $Q \times \Sigma \times \{0, 1\}$ to $Q \times \Sigma \times \{L, R\} \times \{R\}$ (i.e., the random-bit tape is read-only).*

In other words, a PTM is just a DTM that flips a fair coin at each step of the computation. Similarly to what we did for $DTMs$, we can define the notions of configurations; initial, accepting and rejecting configurations; and the next-configuration function $\vdash_M$ for a PTM M. Similarly, we also define (randomized) computational time; two kinds of time complexity are usually measured: the worst-case time complexity and the *expected* time complexity.

For simplicity, we use $M(x) = 1$ to denote that M accepts on input x, and $M(x) = 0$ to denote M rejects on input x. The *error* of a PTM M is defined as $Pr[M(x) \neq \mathcal{X}_{L(M)}(x)]$, where $L(M)$ is the set of all inputs accepted by M and $\mathcal{X}_{L(M)}(x)$ is 1 if $x \in L(M)$ and 0 if $x \notin L(M)$. In other words, the error of a PTM M is the chance it will give a wrong answer.

Now we are ready to define the randomized complexity classes BPP, RP and ZPP.

Definition 2 *BPP (bounded-error randomized poly-time) is the class of languages accepted by $PTMs$ that have the error probability bounded by a constant $c < 1/2$; i.e., $err_M(x) < 1/4$. In fact, it is sufficient to have $err_M(x) \leq \frac{1}{2} - \frac{1}{p(n)}$ for some polynomial p.*

Definition 3 *RP (randomized poly-time) is the class of languages A accepted by a polynomial-time PTM M such that $err_M(x) = 0$ for every $x \notin A$ and $err_M(x) < 1/2$ for $x \in A$.*

Definition 4 *ZPP (zero-error randomized expected poly-time) is $RP \cap coRP$.*

To improve the error bound, for deciding languages in BPP (respectively, RP), the algorithm can be executed $q(n)$ times, q a polynomial, taking a majority vote on the outcomes (respectively, accepting if any execution accepts and rejecting otherwise). There are $2^{q(n)}$ *trials*, or sequences of "coin tosses", each corresponding to a computation sequence of the PTM, and we want morre than $\frac{1}{2}2^{q(n)}$ of these trials to give the right answer.

Proposition 1 (Chernoff Bound) *Let $X_1, \ldots, X_m$ be independent 0-1 random variates such that $\Pr[X_i = 1] = p$. Define $X = \sum_{i=1}^{m} X_i$ and $\mu = \sum_{i=1}^{m} E[X_i] = mp$. Then*

$$\Pr[|X - \mu| \geq r] \leq 2 \cdot e^{-r^2/2m}$$

The Chernoff Bound gives us the following. Think of each X_i as the outcome of a trial each trial succedding with probability $\geq \frac{3}{4}$. The mean of the sum of these m trials is $\geq \frac{3}{4}m$. Then we get (if the sum is less than $\frac{1}{2}m$:

$$\Pr[|X - \mu| \geq m/4] \leq 2 \cdot e^{-(m^2/16)/(m/2)} \leq c^{-m}$$

In other words, the Chernoff Bound tells us that the error is exponentially small w.r.t. the number of trials.

2 Examples of natural problems

There are not many convincing examples of problems computable in randomized polynomial time not computable in deterministic polynomial time. Moreover, there are no known complete problems for these randomized classes.

2.1 Symbolic Determinant Problem

Definition 5 *Given a matrix $A_{n\times n}$ whose entries are multivariate polynomials in $x_1, \ldots, x_{r(n)}$ of degree $\leq q(n)$ where q is a polynomial, is $det(A)$ is not identically 0?*

That is, SDP $= \{A : det(A) \not\equiv 0\}$.

Claim: SDP $\in RP$

Proof: One well known (and polytime) method for computing determinants of *numeric* matrices is Guassian elimination. We use this fact to decide SDP as follows. Substitute the variables in A by random numbers, let A_N be the *numeric* matrix resulting from the substitutions. Compute $det(A_N)$ using Guassian elimination, then if we get a non-zero result, then we *know* $A \in$ SDP (thus, we *accept*); if we get zero, then either the symbolic determinant was identically zero or we randomly chose a root of the polynomial corresponding to the symbolic determinant of A. In either case we *reject*. The following result helps giving an estimate on the error of the PTM sketched above.

Theorem 1 (Schwartz-Zippel Lemma) *Let* $Q = \sum_{k=1}^{n} c_k \cdot x_1^{i_1} x_2^{i_2} \cdots x_n^{i_n}$ *be a polynomial on* n *variables, not identically zero, of degree* $d = \max \sum i_j$ *over a field* $\mathbb{F}$*. Let* $a_1, \ldots, a_n$ *be* n *numbers independently and uniformly chosen from* $S \subseteq \mathbb{F}$*. Then, assuming that* Q *is not idnetically zero,*

$$\Pr[Q(a_1, \ldots, a_n) = 0] \leq \frac{d}{|S|}$$

The proof of the Schwartz-Zippel Lemma is by induction on n, the number of variables (see [1, pp244]). Now, we know that the degree of $det(A)$ is $n \cdot q(n)$ (recall the definition above), thus all we need to do is choose a big enough S so as to reduce the error. This completes the proof. □

2.2 Primes

Theorem 2 PRIMES $\in ZPP$.

Comments: This is a hard result. It is relatively easy to show that COMPOSITES $\in RP$, but hard to show that PRIMES $\in RP$ (see the textbook).

There is a fallacious proof of COMPOSITES $\in RP$, based on the following:

Theorem 3 (Fermat's Theorem) *If* p *is prime, then* $\forall a, 1 \leq a \leq p - 1,\ a^{p-1} = 1 \bmod p$.

□

Fermat's test: choose a at random, if $a^{p-1} \neq 1 mod\ p$ say *composite*, else, say *prime*. This test does not work because of Carmichael Numbers (the *false primes*):

$$a^{n-1} = 1 \bmod n \quad \forall a \ \gcd(a, n) = 1$$

As far as I know, there are no convincing examples of problems in $BPP - (RP \bigcup coRP)$.

3 Relationships among complexity classes

$$P \subseteq ZPP \subseteq RP \begin{array}{c} \subseteq BPP \\ \\ \subseteq NP \end{array} \subseteq PSPACE$$

4 Two basic results about BPP

Theorem 4 $BPP \subseteq POLYSIZE$.

This result is rather easy.

Theorem 5 *$BPP \subseteq \Sigma_2^P$ (Sipser, Lautemann, Kabanets).*

Proof sketch: We want to characterize BPP languages as Σ_2^P languages. Let $L \in BPP$ and let M be a PTM that decides L. Let $y = r_1, \ldots, r_m$ be the random bit sequecne used for a randomized computation of M on input x. Now define

$$S_x = \{y | M \text{ accepts } x \text{ with random computation sequence } y\} \subseteq \{0,1\}^m.$$

We know that

- if $x \in L$, then $|S_x|$ is large (say, $\geq 1 - \frac{1}{2^{p(n)}}$).
- if $x \notin L$, then $|S_x|$ is small (say, $\leq \frac{1}{2^{p(n)}}$).

Now, we define

$$\mathcal{H}(S) \equiv \exists u_1, \ldots u_m \text{ s.t. } \forall v[(u_1 \oplus v \in S) \vee (u_2 \oplus v \in S) \vee \ldots \vee (u_m \oplus v \in S)]$$

Note that $\mathcal{H}$ is a typical Σ_2^P language. All we need to show is:

- if $|S|$ is large, then $\mathcal{H}(S)$ is true;
- if $|S|$ is small, then $\mathcal{H}(S)$ is false.

We will continue this proof in next lecture. □

References

[1] Christos Papadimitriou. *Computational Complexity.* Addison Wesley, 1995.