CSE 531: Computational Complexity I **Winter 2016**

Lecture 11: Randomization and Complexity

Feb 10, 2016

Lecturer: Paul Beame *Scribe: Paul Beame*

1 Randomized Computation

Though Turing machines capabilities are quite general, we have not included an important aspect of computation that may be practical and useful: the ability to use random choices in the design of algorithms.

Definition 1.1. *A* probabilistic Turing machine (PTM) M *is a TM with two transition functions in place of the usual one:* $\delta_0 : Q \times \Gamma^{k+1} \to Q \times (\Gamma \times L, S, R)^{k+1}$ *and* $\delta_1 : Q \times \Gamma^{k+1} \to Q \times (\Gamma \times L, S, R)^{k+1}$. *At each time step, independently, the algorithm chooses* b *from* $\{0, 1\}$ *with equal probability 1/2 and makes its move using transition function* δ_b. *The running time of* M *is the maximum number of steps before* M *halts over all random choices.*

But what does it mean for a probabilistic Turing machine to compute a Boolean function or decide a language A? We have several possibilities:

- 2-sided error: The probability the algorithm is correct is bounded away from 1/2, for example with probability $\geq 2/3$.
- 1-sided error: The algorithm succeeds with probability bounded away from 0, say probability $\geq 2/3$, when $x \in A$, but never makes an error when $x \notin A$. (Alternatively, we may swap the cases.)
- 0-sided error: The algorithm never makes an error. In this case we cannot assume a bounded running time, since any algorithm that is always correct and has a fixed bound on time can be replaced by a deterministic algorithm that only uses δ_0 (or only used δ_1). Instead, in this case we use expected running time.

For each of these notions of correctness and time bounds we define the corresponding complexity classes. For the first two notions, instead of using PTMs directly, we use an equivalent definition that lets us explicitly discuss the random choices made by the algorithm. The terminology as with the names of some other complexity classes is not especially consistent in style. For all of these

choices it is important to note that the complexity is the worst-case complexity over the choice of the input. The only random choices are made by the algorithm.

For convenience, for each language $A \subseteq \{0,1\}^*$, we also define an associated Boolean function on $\{0,1\}^*$, such that $A(x) = 1$ iff $x \in A$.

Definition 1.2. *Let $T : \mathbb{N} \to \mathbb{N}$ be a running time bound and $\varepsilon : \mathbb{N} \to [0,1]$ be an error bound.* $\mathsf{BPTIME}_\varepsilon(\mathsf{T}(\mathsf{n}))$ *is the set of languages A such that there is a TM M with running time $t(n)$ that is $O(T(n))$ such that*

$$\mathbb{P}_{r \in_R \{0,1\}^{t(|x|)}}[M(x,r) = A(x)] \geq 1 - \varepsilon(|x|)$$

The BP stands for "bounded-error probabilistic". We write $\mathsf{BPTIME}(\mathsf{T}(\mathsf{n})) = \mathsf{BPTIME}_{1/3}(\mathsf{T}(\mathsf{n}))$ and $\mathsf{BPP} = \bigcup_{\mathsf{k}} \mathsf{BPTIME}(\mathsf{n}^{\mathsf{k}})$.

Definition 1.3. *Let $T : \mathbb{N} \to \mathbb{N}$ be a running time bound and $\varepsilon : \mathbb{N} \to [0,1/2)$ be an error bound.* $\mathsf{RTIME}_\varepsilon(\mathsf{T}(\mathsf{n}))$ *is the set of languages A such that there is a TM M with running time $t(n)$ that is $O(T(n))$ such that*

$$x \in A \Rightarrow \mathbb{P}_{r \in_R \{0,1\}^{t(|x|)}}[M(x,r) = 1] \geq 1 - \varepsilon(|x|), \text{ and}$$
$$x \notin A \Rightarrow \mathbb{P}_{r \in_R \{0,1\}^{t(|x|)}}[M(x,r) = 0] = 0.$$

The R stands for "randomized". We write $\mathsf{RTIME}(\mathsf{T}(\mathsf{n})) = \mathsf{RTIME}_{1/3}(\mathsf{T}(\mathsf{n}))$ and $\mathsf{RP} = \bigcup_{\mathsf{k}} \mathsf{RTIME}(\mathsf{n}^{\mathsf{k}})$.

Since this is a one-sided notion of error, there is also the dual complexity class.
We write $\mathsf{coRTIME}(\mathsf{T}(\mathsf{n})) = \{\overline{\mathsf{A}} \mid \mathsf{A} \in \mathsf{RTIME}(\mathsf{T}(\mathsf{n}))\}$ and $\mathsf{coRP} = \{\overline{\mathsf{A}} \mid \mathsf{A} \in \mathsf{RP}\}$.
Clearly $\mathsf{RP}, \mathsf{coRP} \subseteq \mathsf{BPP}$ by definition.

By observing the correspondance of definitions we immediately also have $\mathsf{RTIME}(\mathsf{T}(\mathsf{n})) \subseteq \mathsf{NTIME}(\mathsf{T}(\mathsf{n}))$ for all $T(n)$. (Instead of just one witness, a substantial majority of strings are witnesses.) Hence $\mathsf{RP} \subseteq \mathsf{NP}$ and $\mathsf{coRP} \subseteq \mathsf{coNP}$.

Definition 1.4. *We now define* $\mathsf{ZPTIME}(\mathsf{T}(\mathsf{n}))$ *to be the set of languages A such that there is a polynomial-time PTM M such that for all $x \in \{0,1\}^*$,*

$$x \in A \Leftrightarrow M(x) \textit{ accepts}$$

and for every $x \in \{0,1\}^*$, the expected *running time of M on input x is $O(T(|x|))$.*

In this case, ZP stands for "zero-error".
We write $\mathsf{ZPP} = \bigcup_{\mathsf{k}} \mathsf{ZPTIME}(\mathsf{n}^{\mathsf{k}})$.

Error-free probabilistic algorithms became known as "Las Vegas" algorithms to distinguish them from probabilistic algorithms with error, which had been known as "Monte Carlo" algorithms since the 1940s.

Theorem 1.5. $\mathsf{ZPP} = \mathsf{RP} \cap \mathsf{coRP}$.
More generally, $\mathsf{ZPTIME(T(n))} = \mathsf{RTIME(T(n))} \cap \mathsf{coRTIME(T(n))}$ *for all* $T(n) \geq n$.

Proof. We prove the special case. The general case follows almost identically.
$\subseteq$: Let $A \in \mathsf{ZPP}$.
Then there is a PTM M_A with expected running time at most $T(n) \leq kn^k$ for integer k such that M_A computes $A(x)$ with certainty. By Markov's inequality, the probability that M_A takes time $\geq 3T(n)$ is at most $1/3$. The RP and coRP algorithms will both run M_A for $3kn^k$ steps. If M_A has halted in this many step then output the answer produced by M_A. In the case that M_A has not yet halted, the RP algorithm will reject, and the coRP algorithm will accept.
$\supseteq$: Let $A \in \mathsf{RP} \cap \mathsf{coRP}$.
Let M_A^1 be the RP algorithm for A, which is always correct when it outputs 1.
Let M_A^0 be the coRP algorithm for A, which is always correct when it outputs 0.
Let kn^k be an upper bound on the running time for each of M_A^1 and M_A^0. The zero-error algorithm is as follows:

Repeat forever:
.25in Choose a kn^k random bit string r and run both M_A^1 and M_A^0 on input x.
.25in If M_A^1 outputs 1 then halt and output 1; if M_A^0 outputs 0 then halt and output 0.

If $x \in A$ then M_A^1 will output 1 with probability $\geq 1/3$ per iteration. if $x \notin A$ then M_A^0 will output 0 with probability $\geq 1/3$ per iteration. The expected # of iterations required is 3 and hence the total running time is at most $6kn^k$. □

We have now discussed the basic definitions of randomized computation. Randomization is essential for subjects such as cryptography where one needs to randomization to have unpredictable secrets.

However, it is still open whether or not randomization helps for language recognition: Is $\mathsf{P} = \mathsf{BPP}$ or $\mathsf{P} = \mathsf{ZPP}$. When it was originally introduced for computation of languages, one of the first examples was the case of $PRIMES \in \mathsf{coRP}$ which gave a randomized algorithm for compositeness. However, in 2002, Agrawal, Kayal, and Saxena proved that $PRIMES \in \mathsf{P}$. Unlike the P versus NP question, where researchers overwhelming believe that nondeterminism increases computational power, the consensus is not so clear regarding the power randomization. However, more would likely bet on the ability to avoid its use. One of the other examples that still remains is that of polynomial identity testing, which is in some sense the key candidate problem in the area.

2 Polynomial Identity Testing

Definition 2.1. *An* arithmetic (algebraic) circuit C *over* $\mathbb{Z}$ *or over a finite field* $\mathbb{F}$ *is a circuit with binar fan-in gates labelled by* $+$, $-$, $\times$, *and variable names* $x_1, \ldots, x_n$, *as well as constants* 0 *and* 1. *It computes a multivariate polynomial in the input variables* $x_1, \ldots, x_n$.

Definition 2.2. *The* polynomial identity testing *problem* $PIT_{\mathbb{F}} = \{[C], [C'] \mid C,\ C'$ *are arithmetic circuits over* $\mathbb{F}$ *By adding a single* $-$ *to join* C *and* C' *at the output, it is clearly equivalent to the following problem which is more convenient:* $ZEROP_{\mathbb{F}} = \{[C] \mid C$ *is an arithmetic circuit over* $\mathbb{F}$ *computing the 0 polynomial*$\}$. *The analogous definitions hold if we replace* $\mathbb{F}$ *by* $\mathbb{Z}$.

Theorem 2.3. $PIT_{\mathbb{Z}},\ ZEROP_{\mathbb{Z}} \in$ coRP. *The same holds true for sufficiently large finite fields* $\mathbb{F}$.

The proof of the theorem will be derived from the following lemma.

Lemma 2.4 (Schwartz,Zippel)**.** *Let* $p(x_1, \ldots, x_n)$ *be a non-zero polynomial over* $\mathbb{Z}$ *(respectively* $\mathbb{F}$*) of (total) degree* $\leq d$. *Let* $S \subseteq \mathbb{Z}$ *(respectively* $\mathbb{F}$*) be a finite set. If* $a_1, \ldots, a_n$ *are independently and uniformly chosen from* S *then*

$$\mathbb{P}_{a_1,\ldots,a_n}[p(a_1, \ldots, a_n) = 0] \leq \frac{d}{|S|}.$$

Proof. We prove this by induction. The case of $n = 0$ is trivial, since a non-zero constant polynomial is never 0.

We prove a base case for $n = 1$ also. In this case, $p \neq 0$ is an ordinary univariate polynomial of degree at most d. By the Fundamental Theorem of Arithmetic it has at most d roots in $\mathbb{Z}$ or in in $\mathbb{F}$. At most d of these roots are in S so the probability that the single variable a_1 is chosen to be one of them is $d/|S|$ as required.

Now suppose that it is true for 1 and n. We prove it for $n + 1$ We expand p using the powers of x_{n+1}. Let i^* be the largest power of x_{n+1} in p. Then

$$p(x_1, \ldots, x_{n+1}) = \sum_{i=0}^{i^*} p_i(x_1, \ldots, x_n) x_{n+1}^i$$

where each p_i has total degree at most $d-i$. In particular, observe that if $p_{i^*}(a_1, \ldots, a_n)$ is non-zero then $q(x_{n+1}) = p(a_1, \ldots, a_n, x_{n+1})$ is a non-zero polynomial in x_{n+1} of degree i^*. Therefore

$$\begin{aligned}
&\mathbb{P}_{a_1,\ldots,a_{n+1}}[p(a_1, \ldots, a_{n+1}) = 0] \\
\leq &\mathbb{P}_{a_1,\ldots,a_n}[p_{i^*}(a_1, \ldots, a_n) = 0] \\
&+ \mathbb{P}_{a_1,\ldots,a_{n+1}}[p(a_1, \ldots, a_{n+1}) = 0 \mid p_{i^*}(a_1, \ldots, a_n) \neq 0] \\
\leq &\frac{d - i^*}{|S|} + \frac{i^*}{|S|} = \frac{d}{|S|}.
\end{aligned}$$

where the first bound follows by induction from the fact that the degree of p_{i^*} is at most $d - i^*$ and it has at most n variables and the second bound follows from the univariate case.

The extension to $\mathbb{F}$ is immediate. $\square$