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

Lecture 12: Randomization and Complexity II

Feb 12, 2016

Lecturer: Paul Beame *Scribe: Paul Beame*

1 Polynomial Identity Testing continued

Last time we proved the following lemma:

Lemma 1.1 (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*

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

We now complete the proof the following theorem, which introduces additional algorithmic ideas that are broadly useful.

Theorem 1.2. $PIT_{\mathbb{Z}}$, $ZEROP_{\mathbb{Z}} \in$ coRP. *The same holds true for sufficiently large finite fields, for example the integers modulo a prime q.*

Proof. The general idea of the coRP algorithm is to randomly choose elements $a_1, \ldots, a_n$ from a succiently large S and evaluate the input circuit. If the circuit evaluates to a non-zero value then the output the $ZEROP$ algorithm should reject since the polynomial is surely nonzero. Conversely, if the polynomial is nonzero, then there is a good chance that it will not evaluate to 0 by the Schwartz-Zippel Lemma.

It remains to make this work: If the circuit size of C is at most m then C has at most m multiplication gates. Each such gate can at most double the degree of the output polynomial. (The $+$ and $-$ gates do not change the maximum degreee.) Therefore, the circuit has degree $d \leq 2^m$. Hence choosing S of size 2^{m+2} will be sufficient, e.g. $S = \{0, 1, \ldots, 2^{m+2} - 1\}$. If we are in the case of the finite field with prime q of size not much larger than 2^{m+2}, then we can compute each gate of the circuit and then take the value modulo q at each step, so the circuit evaluation is polynomial time and will have failure probability at most $1/4$.

However, over the integers $\mathbb{Z}$ it becomes much more difficult. Each a_i takes $m+2$ bits and a degee $d = 2^m$ monomial in the a_i will require more than $2^m(m + 2)$ bits, which is too many to write down. Even a polynomial in the a_i of degree 2^m will not need many more bits.

Fingerprinting To do this we need to use a technique called *fingerprinting* introduced by Rabin. The idea is to choose $4m$ random numbers $k_1, \ldots, k_{4m}$ of $2m$ bits each and evaluate $C(a_1, \ldots, a_n)$ by taking each gate modulo k_i for all i. Clearly this is a polynomial-time computation. The algorithm will accept unless some computation of C modulo k_i produces a non-zero value.

For correctness, observe that
$C(a_1, \ldots, a_n) \equiv 0 \bmod k_1, \cdots, C(a_1, \ldots, a_n) \equiv 0 \bmod k_{4m}$ if and only if

$$C(a_1, \ldots, a_n) \equiv 0 \bmod \mathrm{lcm}(k_1, \ldots, k_{4m}),$$

where lcm stands for "least common multiple".

Fact 1.3. $\mathbb{P}[\mathrm{lcm}(k_1, \ldots, k_{4m}) < 2^{2^{2m}}] \leq 1/16$

This follows by the Prime Number Theorem. (In fact, roughly $1/2m$ of the numbers k_i is in fact a prime.)

Hence the total probability of failure is the probability that the $a_1, \ldots, a_n$ chosen from S were a bad choice is at most $1/4$, plus the probability that the $k_1, \ldots, k_{4m}$ are bad, which is at most $1/16$ yielding failure at most $1/4 + 1/16 < 1/3$ as required. □

2 Reducing Errors

The following shows that the definitions of randomized classes are not especially sensitive to the error parameters.

Theorem 2.1. *Let* $T : \mathbb{N} \to \mathbb{N}$.

1. *If* $\delta : \mathbb{N} \to [0, 1]$ *then*
   - *(a)* $\mathsf{RTIME}_{1-\delta}(\mathsf{T}(\mathsf{n})) \leq \mathsf{RTIME}_{1/3}(\mathsf{T}(\mathsf{n})/\delta)$.
   - *(b)* $\mathsf{BPTIME}_{1/2-\delta}(\mathsf{T}(\mathsf{n})) \leq \mathsf{BPTIME}_{1/3}(\mathsf{T}(\mathsf{n})/\delta^2)$.
2. *If* $K : \mathbb{N} \to \mathbb{N}$ *then*
   - *(a)* $\mathsf{RTIME}_{1/3}(\mathsf{T}(\mathsf{n})) \leq \mathsf{RTIME}_{2^{-\mathsf{K}}}(\mathsf{K} \cdot \mathsf{T}(\mathsf{n}))$.
   - *(b)* $\mathsf{BPTIME}_{1/3}(\mathsf{T}(\mathsf{n})) \leq \mathsf{BPTIME}_{2^{-\mathsf{K}}}(\mathsf{K} \cdot \mathsf{T}(\mathsf{n}))$.

Proof. For the 1(a) and 2(a), the algorithm is simply to run the RTIME algorithm with the larger error multiple times with independent random choices and accept iff any of the runs accepts. In that case, $2/\delta$ runs of an algorithm with error $1 - \delta$ will fail with probability at most $(1 - \delta)^{2/\delta} \leq$

$e^{-2} < 1/3$ since $(1-x)^x \leq e^{-1}$ for all real values x. K runs at error $1/3$ these will reduce the error 1/3 to error 3^{-K} which suffices.

For 1(b) and 2(b), since errors can occur in either direction, the algorithm is to run the BPTIME algorithm with the larger error multiples times with independent random choices and to output the majority of the answers computed. The bounds now follow by Chernoff bounds on the tails of binomial distributions (sums of independent Bernoulli trials). These give bounds that decay by the square of the difference between the threshold (1/2) to the probabilities and hence yield the larger dependence on δ in the bound. □

Corollary 2.2.
$\mathsf{BPP}_\varepsilon(\mathsf{n}) = \mathsf{BPP}$ *for* $2^{-n^{O(1)}} \leq \varepsilon(n) \leq 1/2 - n^{-O(1)}$.
$\mathsf{RP}_\varepsilon(\mathsf{n}) = \mathsf{RP}$ *for* $2^{-n^{O(1)}} \leq \varepsilon(n) \leq 1 - n^{-O(1)}$.

3 Randomization versus Non-uniformity

Theorem 3.1 (Adleman,Bennet-Gill)**.** $\mathsf{BPP} \subseteq \mathsf{P/poly}$.

Proof. Let $A \in \mathsf{BPP}$. By the corollary, $\mathsf{BPP} = \mathsf{BPP}_{2^{-2n}}$. Therefore there is a polytime TM M_A and a polynomial p such that

$$\forall n \geq 0\, \forall x \in \{0,1\}^n\, \mathbb{P}_{r \in \{0,1\}^{p(n)}}[M_A(x,r) \neq A(x)] \leq 2^{-2n}.$$

Fix n. Define the indicator predicate $\mathbb{1}_{M_A(x,r) \neq A(x)}$ on (x,r) to have value 1 if $M_A(x,r) \neq A(x)$ and 0 otherwise; i.e. this is the error predicate.

We can think of a matrix with rows indexed by elements $x \in \{0,1\}^n$ and columns indexed by elements $r \in \{0,1\}^{p(n)}$ and with (x,r) entry equal to the value of the predicate. The correctness of the algorithm implies that each row has at most a 2^{-2n} fraction of entries equal to 1, and hence the whole matrix has at most a 2^{-2n} fraction of 1 entries. This means that there must be some column with at most a 2^{-2n} fraction of 1 entries.

(Deriving this in formulas we have,

$$\begin{aligned}
\forall x \in \{0,1\}^n\, \mathbb{E}_{r \in \{0,1\}^{p(n)}} \mathbb{1}_{M_A(x,r) \neq A(x)} &\leq 2^{-2n} \\
\text{So } \mathbb{E}_{x \in \{0,1\}^n} \mathbb{E}_{r \in \{0,1\}^{p(n)}} \mathbb{1}_{M_A(x,r) \neq A(x)} &\leq 2^{-2n} \\
\text{and } \mathbb{E}_{r \in \{0,1\}^{p(n)}} \mathbb{E}_{x \in \{0,1\}^n} \mathbb{1}_{M_A(x,r) \neq A(x)} &\leq 2^{-2n} \\
\text{So } \exists r_0 \in \{0,1\}^{p(n)}.\, \mathbb{E}_{x \in \{0,1\}^n} \mathbb{1}_{M_A(x,r_0) \neq A(x)} &\leq 2^{-2n} \\
\text{and } \exists r_0 \in \{0,1\}^{p(n)}.\, \mathbb{P}_{x \in \{0,1\}^n}[M_A(x,r_0) \neq A(x)] &\leq 2^{-2n}
\end{aligned}$$

Since the matrix has only 2^n elements in any column, this column must only have 0 entries. If we fix the r_0 associated with the column and give these $p(n)$ bits as advice for inputs of length n, then by construction the algorithm will always be correct on any input of length n. □

4 Randomization versus PH

Theorem 4.1 (Sipser-Gacs-Lauteman)**.** $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}}$.

Proof. Since BPP is closed under complement, it suffices to prove that $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}}$. We use that $\mathsf{BPP} = \mathsf{BPP}_{2^{-n}}$. Therefore there is a polytime TM M using $m = p(n)$ random bits such that $\mathbb{P}_{r\in\{0,1\}^m}[M(x,r) \neq A(x)] \leq 2^{-n}$.

For each $x \in \{0,1\}^n$ let $S_x \subseteq \{0,1\}^m$ be the set of r such that $M(x,r) = 1$.

Observe that
if $x \in A$ then $|S_x|/2^m \geq (1 - 2^{-n})$ but
if $x \notin A$ then $|S_x|/2^m \leq 2^{-n}$.

That is, S_x is either tiny or it is almost all of $\{0,1\}^m$. Determining whether x is in A is equivalent to determining of whether S_x is tiny or huge; moreover, by definition we have a polynomial-time test for membership of r in S_x, namely run M on input (x,r).

The key is to look at many "shifts" of S_x. For a set $S \subseteq \{0,1\}^m$ and a vector $u \in \{0,1\}^m$, define $S \oplus u = \{r \oplus u \mid r \in S\}$.

The following lemma was proved by Lautemann. (Sipser and Gacs used a different argument.)

Lemma 4.2. *Let $S \subseteq \{0,1\}^m$.*

1. *Let $k = \lceil \frac{m}{n} + 1 \rceil$. If $|S|/2^m \geq 1 - 2^{-n}$ then $\exists u_1, \ldots, u_k \in \{0,1\}^m$ such that $\bigcup_{i=1}^k (S \oplus u_i) = \{0,1\}^m$.*
2. *If $k < 2^n$ and $|S|/2^m \leq 2^{-n}$ then $\forall u_1, \ldots, u_k \in \{0,1\}^m$ such $\bigcup_{i=1}^k (S \oplus u_i) \neq \{0,1\}^m$.*

We first see how the statement follows from the lemma. By definition, for $k = \lceil \frac{m}{n} + 1 \rceil$ we have

$$\begin{aligned} x \in A \Leftrightarrow\ & \exists u_1, \ldots, u_k \in \{0,1\}^m\ \forall r \in \{0,1\}^m\ r \in \bigcup_{i=1}^{k} (S_x \oplus u_i) \\ \Leftrightarrow\ & \exists u_1, \ldots, u_k \in \{0,1\}^m\ \forall r \in \{0,1\}^m\ \bigvee_{i=1}^{k} (r \oplus u_i \in S_x) \\ \Leftrightarrow\ & \exists u_1, \ldots, u_k \in \{0,1\}^m\ \forall r \in \{0,1\}^m\ \bigvee_{i=1}^{k} (M(x, r \oplus u_i) = 1) \end{aligned}$$

which shows that $A \in \Sigma_2^{\mathsf{P}}$ as required. □

We finish by giving the proof of the lemma in the next class.