CS 294-114 – Fine-Grained Complexity and Algorithms	Sep 22, 2015

Lecture 8: Derandomization from Circuit Lowerbounds

Instructor: Russell Impagliazzo — *Scribe: Jingcheng Liu*

Recall from last class: Nisan-Wigderson's construction of PRG from an average-case hard function:

Theorem 1 (NW88) *If there is a* $(\text{poly}(m), 1/2 - 1/\text{poly}(m))$ *hard function on* n *bits, then exists PRG:* $G : \{0,1\}^{O(n^2)} \to \{0,1\}^m$*, such that G-RDP is* $1/3$*-hard for circuits of size at most* m.

To bridge the gap, we will see how to obtain an average-case hard function from a worst-case hard function. From the perspective of locally decodable (or locally list-decodable) error-correcting code, (the truth table of) a worst-case hard function is to be sent over a noisy channel, and one tries to recover (decode) the function from its noisy version. We require the decoding to be local, since recovering the whole truth table of a function would be too expensive and unnecessary, all that we need is to recover $f(\boldsymbol{x})$ for given input $\boldsymbol{x}$.
We will use a locally decodable ECC known as Reed-Muller code to encode a worst case function f_{wc}. Due to an inherent limitation of unique decoding property, one only gets a somewhat hard function f_{sh}. From then on we 'amplify' the hardness using Yao's XOR lemma. As a side note, this is usually not satisfactory enough to get full derandomization of BPP. Another approach that bypasses the XOR lemma is to use locally decodable ECC instead.

1 Reed-Muller code

Given $f_{wc} : \{0,1\}^n \to \{0,1\}$, let its Fourier expansion be $f_{wc}(x_1, \cdots, x_n) = \sum_S \alpha_S x^S$. For large enough p, consider $f_{sh} : \mathbb{Z}_p^n \to \mathbb{Z}_p$ by extending f_{wc} to $\mathbb{Z}_p$:

$$f_{sh}(\boldsymbol{x}) = f_{wc}(\boldsymbol{x}) = \sum_S \alpha_S x^S \mod p.$$

Claim 1 *If* C *computes* f_{sh} *on* $1 - \frac{1}{3(n+1)}$ *fraction of inputs, there exists* C' *of roughly the same size computing* f_{wc}.

We'll construct a probabilistic circuit C'' such that

$$\forall x, \Pr_{C''}[C''(x) = f_{sh}(x)] \geq 2/3,$$

and just let C' take the majority on C''. [1]
Now we construct C''. The idea is known as self-reducibility. Say we're interested in $f_{wc}(\boldsymbol{x})$, we will decode it from $f_{sh}(\boldsymbol{y})$, where $\boldsymbol{y}$ is chosen as follows: Start with a random direction $\boldsymbol{a}$, we evaluate f_{sh} restricted on the line $\boldsymbol{y} = \boldsymbol{x} + z\boldsymbol{a}$ parameterized by z:

$$fl(z) = \sum_S \alpha_S \prod_i (x_i + za_i)$$

[1] with the 'best' randomness over C'' hard-wired in, SEE ALSO the proof of BPP⊂P/poly in textbook of Sipser, or Barak and Arora.

Note that $fl(z)$ is a degree n univariate polynomial, and $fl(0) = f_{sh}(x)$, which is the noisy version of $f_{wc}(x)$. To 'decode' $f_{wc}(x)$, we evaluate $fl(z)$ for $z = 1, \cdots, n+1$, since $\boldsymbol{a}$ is uniformly random, $\boldsymbol{y}$ will be uniformly random too. Finally we interpolate $\widetilde{fl(z)}$, and output $\widetilde{fl(0)}$.
Now by union bound,

$$\Pr[\exists i : 1 \leq i \leq n+1, \widetilde{fl(i)} \neq fl(i)] \leq \frac{(n+1)}{3(n+1)} = 1/3.$$

Hence the (probabilistic) circuit that computes $\widetilde{fl}$ will be the C'' we need.
Note that here we omitted the steps of truncating the decoded function back to a boolean function. This can be done via concatenating another code that encodes the binary alphabet.

2 Hardness amplification and the Hard-core lemma

There are 2 possibilities that a function is 'somewhat hard' to compute by small circuits (with probability better than $1 - \delta$):

- the hardness is 'spread' out over the boolean cube, different circuits fail on different places.
- there is a single subset of δ fraction of inputs such that the function is very hard on those inputs for every small circuits.

The hard-core lemma explains that the latter always happen.

Lemma 2 (Impagliazzo's Hard-core lemma) *Let f_{sh} be any (m, ε)-hard function, then $\exists H \subset \{0,1\}^n, |H| \geq \Omega(\varepsilon 2^n)$, such that $\forall C', |C'| \leq m \operatorname{poly}(\varepsilon, \delta)$,*

$$\Pr_{x \in H}[C'(x) \neq f(x)] \geq \frac{1}{2} - \delta.$$

This can be proved using min-max theorem:

Theorem 3 (von Neumann's min-max theorem) *For a zero-sum 2-player game, if we allow randomized strategies, the order of play doesn't change the outcome.*
Specifically let A be the payoff matrix, and x, y be distribution over $[n]$ strategies.

$$\min_x \max_y x^\intercal A y = \max_y \min_x x^\intercal A y.$$

Proof. Player A's strategy is to specify a circuit C that computes f_{sh}. Player B is to find $S \subset 0, 1^n$ such that $|S| \geq \varepsilon 2^n$. The payoff for C, S is

$$P_{C,S} = \Pr_{x \in S}[f(x) = C(x)] - \Pr_{x \in S}[f(x) \neq C(x)]$$

Suppose the opposite, namely A's payoff is at least δ, then by min-max theorem, A has a distribution over circuits so that $\forall S, |S| \geq \varepsilon 2^n$,

$$\Pr_{C, x \in S}[f(x) = C(x)] - \Pr_{C, x \in S}[f(x) \neq C(x)] \geq \delta$$

Let $\hat{S} = \left\{x : \Pr_C[f(x) = C(x)] \leq \frac{1+\delta}{2}\right\}$, since B didn't choose $\hat{S}$, it must be the case that $\left|\hat{S}\right| < \varepsilon 2^n$.

To this end, we will show a small circuit that is correct for all $x \notin \hat{S}$, this will contradict that f_{sh} is (m, ε)-hard, as $\hat{S} < \varepsilon 2^n$.
The idea is as before, we take roughly n/δ^2 independent copies of circuits from A's distribution, then we take the majority, and call this circuit C. By Chernoff bound, $\forall x \notin \hat{S}, \Pr[C(x) \neq f(x)] < 2^{-n}$. By a union bound, there exists such a C that's correct for every $x \notin \hat{S}$. Finally we just hard-wire the 'good randomness' for such C, we get the contradiction as disired.

Lemma 4 (Yao's XOR Lemma) *Let*

$$F(\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_k) = \sum_i f_{sh}(\boldsymbol{x}_i) \mod 2,$$

and f_{sh} is (m, ε)-hard, then F is $(m \operatorname{poly}(\varepsilon), 1/2 - 2(1-\varepsilon)^k)$-hard.

Proof. The key argument is that, as long as one of the $\boldsymbol{x}_i$ fall into the hard-core set, then even if one can compute every other $\boldsymbol{x}_j$, the XOR will make F as hard to compute as $f_s h$ on the hard-core set. So if we instantiate the hard-core lemma with $\delta = (1-\varepsilon)^k$, then for circuit C,

$$\begin{aligned}\Pr\left[C(\boldsymbol{X}) = F(\boldsymbol{X})\right] \leq & \Pr\left[\text{none of the } \boldsymbol{x}_i \text{ fall into the hard-core set}\right] + \\ & \Pr\left[\boldsymbol{x}_i \text{ fall into the hard-core set}, C(\boldsymbol{X}) \oplus \sum_{j \neq i} f_{sh}(\boldsymbol{x}_j) = f_{sh}(\boldsymbol{x}_i)\right] \\ \leq & (1-\varepsilon)^k + 1/2 + (1-\varepsilon)^k.\end{aligned}$$

To this end, we have obtained an average-case hard function from a worst case hard function.