Average Case Complexity February 10, 2011

Hardness Amplification: Yao's XOR Lemma

Professor: Valentine Kabanets Scribe: Hoda Akbari

1 Hardness Amplification

Starting with a function that is somewhat hard on average, our aim is to get an even harder function.

Lemma 1 (**Yao's XOR Lemma [1]**– non-uniform version)**.**
Let function $f : \{0,1\}^n \to \{0,1\}$ be δ-hard for size s, and $k > 0$ be any integer. For $x_1, \cdots, x_k \in \{0,1\}^n$, define $f^{\oplus k}(x_1, \cdots, x_k) = f(x_1) \oplus \cdots \oplus f(x_k)$. Then for all $\varepsilon > 0$, $f^{\oplus k}$ is $\left(\frac{1}{2} - \frac{\varepsilon}{2} - \frac{(1-2\delta)^k}{2}\right)$-hard for size $s' = s \cdot poly(\varepsilon, \delta)$.

Intuition. Since f is δ-hard, there exists a hard core set H of size $|H| \geq 2\delta \cdot 2^n$ such that H is ε-hard core (for size s'), meaning that for any circuit $C, |C| \leq s'$:

$$\Pr_{x \sim H}[C(x) = f(x)] \leq \frac{1}{2} + \frac{\varepsilon}{2}$$

Let's ignore ε as it can be arbitrarily small. Observe that when choosing k samples, there is a good probability of falling into H:

$$\Pr_{x_1, \cdots, x_k}[\text{no } x_i \text{ falls in} H] \leq (1 - 2\delta)^k$$

Even if we know $f(x_j)$ for all $j \neq i$, when $f(x_i)$ is completely random and independent from other $f(x_j)$'s, it makes the whole $f^{\oplus k}(x_1, \cdots, x_k)$ completely unpredictable.

Definition 1 (Advantage)**.** *Advantage of an algorithm Alg to compute $f^{\oplus k}$ is defined as:*

$$\text{adv}_{Alg} = \Pr_{\bar{X}}[Alg(\bar{X}) \text{ is correct }] - \Pr_{\bar{X}}[Alg(\bar{X}) \text{ is incorrect }]$$

For each input $\bar{X}$, we define

$$\text{adv}_{Alg}(\bar{X}) = \Pr_{Alg}[Alg(\bar{X}) \text{ is correct }] - \Pr_{Alg}[Alg(\bar{X}) \text{ is incorrect }],$$

where the probability is over internal randomness of Alg (if any).

Before proceeding with the formal proof, we give an information theoretic reasoning why this result is correct at the intuitive level. Suppose $\chi(.)$ is a characteristic function which is 1 if its argument (an event) happens.

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\bar{X}}[\chi(Alg(\bar{X}) \text{ is correct}) - \chi(Alg(\bar{X}) \text{ is incorrect})] = \\
&\qquad \mathop{\mathbb{E}}_{\bar{X}}[\chi(Alg(\bar{X}) \text{ is correct}) - \chi(Alg(\bar{X}) \quad \text{is incorrect}) \quad | \exists i : x_i \in H] \cdot \Pr[\exists i : x_i \in H] + \\
&\qquad \mathop{\mathbb{E}}_{\bar{X}}[\chi(Alg(\bar{X}) \text{ is correct}) - \chi(Alg(\bar{X}) \quad \text{is incorrect}) \quad | \nexists i : x_i \in H] \cdot \Pr[\nexists i : x_i \in H] \\
&\qquad\qquad \leq \left(1.\frac{1}{2} - 1.\frac{1}{2}\right) + 1.(1 - 2\delta)^k \quad = (1 - 2\delta)^k \qquad (1)
\end{aligned}$$

Therefore,

$$P_{\bar{X}}[Alg(\bar{X}) \text{ is correct}] \leq \frac{1}{2} + \frac{\text{adv}_{Alg}(\bar{X})}{2} \leq \frac{1}{2} + \frac{(1-2\delta)^k}{2}$$

The only difference of $\varepsilon/2$ is to account for the fact that the function is not completely random, but there is $\varepsilon/2$ slack between our function and the actual random coin flip.

Proof. Using the hard core results as a black-box, the proof works by contradiction. Suppose there is some circuit C of size s' such that:

$$\mathop{\mathbb{E}}_{\bar{X}}[\text{adv}_C(\bar{X})] > \varepsilon + (1-2\delta)^k$$

We have:

$$\begin{aligned} \varepsilon + (1-2\delta)^k &< \mathop{\mathbb{E}}_{\bar{X}}[\text{adv}_C(\bar{X}) \mid \exists i : x_i \in H].\Pr[\exists i : x_i \in H] \\ &\quad + \mathop{\mathbb{E}}_{\bar{X}}[\text{adv}_C(\bar{X}) \mid \nexists i : x_i \in H].\Pr[\nexists i : x_i \in H] \\ &\leq \mathop{\mathbb{E}}_{\bar{X}}[\text{adv}_C(\bar{X}) \mid \exists i : x_i \in H].1 + 1.(1-2\delta)^k \end{aligned}$$

Hence,

$$\varepsilon < \mathop{\mathbb{E}}_{\bar{X}}[\text{adv}_C(\bar{X}) \mid \exists i : x_i \in H] \tag{2}$$

To get $\bar{X}$, suppose we do the sampling procedure in the following way:

1: Pick random $1 \leq \ell \leq k$ (ℓ has some distribution, other than uniform distribution)
2: Sample ℓ elements from H independently;
3: Sample $k - \ell$ elements from $\bar{H}$ independently;
4: Form a k-tuple by randomly ordering the elements.

Reformulating equation 2, we get:

$$\mathop{\mathbb{E}}_{\substack{\ell; x_1, \cdots, x_\ell \in H \\ y_1, \cdots, y_{k-\ell} \notin H}}[\text{adv}_C(\bar{X}) \mid \exists i : x_i \in H] > \varepsilon$$

We fix randomness for everything except x_1. By averaging, there exists a good choice of ℓ for which there exists good choice for $x_2, \cdots, x_\ell \in H, y_1, \cdots, y_{k-l} \notin H$ such that the expectation is at least preserved:

$$\mathop{\mathbb{E}}_{x \sim H}[\text{adv}_C(\bar{X}) \mid \exists i : x_i \in H] \geq \varepsilon$$

The circuit C works as follows:

" Given $x \sim H$:

Run $C(x, x_2, \cdots, x_\ell, y_1, \cdots, y_{k-\ell}) \oplus f(x_2) \oplus \cdots \oplus f(y_{k-\ell})$"

Note that "$\oplus f(x_2) \oplus \cdots \oplus f(y_{k-\ell})$"is fixed and can be hard-wired, thus C is of size the same as s' except a constant extra circuit elements required for the hard-wired part. As a result, $\text{adv}_C(\bar{X}) > \varepsilon$ on H, contradicting the fact that H is a hard core. □

Remark. What we discussed so far was the non-uniform version of XOR lemma. The XOR lemma has a *Uniform* version as well:

Lemma 2 (The Uniform Version of the XOR Lemma [2]). *Suppose f is δ-hard, and let $k > 0$ be any integer . Then $f^{\oplus k}$ is γ-hard for $\gamma = \frac{1}{2} - \frac{\varepsilon}{2}$ where $\varepsilon \leq \exp(-\delta k)$. Moreover, there is an efficient randomized algorithm such that, given a circuit C that computes $f^{\oplus k}$ on more than $(1-\gamma)$-fraction of inputs, the algorithm outputs a list of $O(1/\varepsilon^2)$ circuits so that, with high probability, one of the circuits on the list computes f on more than $1-\delta$ fraction of inputs.*

Remark. In the context of error-correcting codes, this lemma gives rise to *approximately list-decodable* codes[2]. That is, given a corrupted codeword corresponding to a message with at most a $\frac{1}{2} - \varepsilon$-fraction of corrupted bits, it is possible to construct a small list of words that contains at least one word that agrees with the original message in all but δ-fraction of bits. Given circuit C computing $f^{\oplus k}$ with adv $\geq \varepsilon$, the algorithm *Reconstruct* outputs $t = O(\frac{1}{\varepsilon^2})$ circuits $C_1, \cdots, C_t$ one of which computes f on more than $1 - \delta$ fraction of inputs.

Hardness amplification for NP

Having f as a δ-hard NP function, we want to get a function within NP but much harder. One possible approach is to make use of the parity function $f^{\oplus k}(x_1, \cdots, x_k) = \oplus_{1 \leq i \leq k} f(x_i)$. As we've shown, the new function is certainly very hard on average. But, the new function may not be in NP (unless NP = CoNP). The reason is that PARITY is not monotone!

PARITY is a good function to use for hardness amplification because it has high sensitivity: changing any one bit has an effect on the value of the PARITY function. We would like to have a monotone Boolean function with similar properties. (Actually, the notion we need is *noise stability*, which we won't define here for lack of time.)

Instead, we use some *good* monotone function $g : \{0,1\}^k \to \{0,1\}$ where

$$f^{gk} = g(f(x_1), \cdots, f(x_k))$$

It turns out that there exist good monotone functions that have noise stability close to that of PARITY, albeit only polynomially good rather than exponentially good. One such good monotone function is the iterated Majority function, which is a composition of a certain number of the Maj_3 (majority of three) functions.

References

[1] A. C. Yao. "Theory and applications of trapdoor functions". In *Proc. 23th IEEE Symposium on Foundations of Computer Science*, pp. 80–91, 1982.

[2] R. Impagliazzo, R. Jaiswal, V. Kabanets. and A. Wigderson. "Uniform Direct-Product Theorems: Simplified, Optimized, and Derandomized". In *SIAM Journal on Computing (SICOMP) 39(4)*, pp. 1637–1665, 2010.