Lecture 13: Fourier-based bounds for codes

Error-Correcting Codes (Spring 2016)
Rutgers University
Swastik Kopparty
Scribes: Meng-Tsung Tsai

1 Overview

This lecture comprises an application of error-correcting code to complexity theory and a Fourier-based upper bound for rate versus relative distance. We will see:

- If PSPACE does not have polynomial size circuits, then PSPACE contains a problem hard to compute in the average case for polynomial size circuits.
- To attain the relative distance $\delta = 1/2 - \varepsilon$, no code can have rate $R = \omega(\varepsilon^2 \log(1/\varepsilon))$.

2 Worst-case Hardness to Average-case Hardness Reduction

Definition 1. *Let $f : \{0,1\}^n \to \{0,1\}$. We say that a circuit c computes f in the worst case if $c(x) = f(x)$ for all $x \in \{0,1\}^n$, and a circuit $\tilde{c}$ computes f in the average case if*

$$\Pr_{x \in \{0,1\}^n}[\tilde{c}(x) = f(x)] \geq 0.9.$$

2.1 Two big question in CS

Q1. find a function that is not computable by any $\mathsf{poly}(n)$-size circuit in the worst case,

Q2. find a function that is not computable by any $\mathsf{poly}(n)$-size circuit in the average case.

The hardness in the average case is important because it has applications to crytography (e.g. pseudo-random number generators).

2.2 Transformation that coverts solution to Q1 to solution to Q2

Suppose there exists a function $f : \{0,1\}^n \to \{0,1\}$ that every $\mathsf{poly}(n)$-size circuit c has an x with $c(x) \neq f(x)$, then one can construct a function $\tilde{f} : \{0,1\}^N \to \{0,1\}$ so that every $\mathsf{poly}(N)$-size circuit $\tilde{c}$ has > 0.1 fraction of disagreement with $\tilde{f}$, i.e.

$$\Pr_{y \in \{0,1\}^N}\left[\tilde{c}(y) \neq \tilde{f}(y)\right] > 0.1.$$

Here is a construction of $\tilde{f}$. We write the truth table of f as a bit-string of length 2^n, and let $\tilde{f} = E(f)$ where $E : \{0,1\}^n \to \{0,1\}^N$ is an encoding mapping of some error correcting code L and L can recover from 0.1 fraction of errors. Since $\tilde{f}$ is a codeword of L, any circuit $\tilde{c}$ that computes $\tilde{f}$ in the average case can be viewed as a received word of code L. In other words, the truth table of $\tilde{f}$ can be recovered from the truth table of circuit $\tilde{c}$. If each bit of the truth table f can be decoded efficiently given the oracle access of each bit of the truth table of small circuit $\tilde{c}$, then it would contradict the non-existence of small circuit c that computes f.

To make the contradiction happened, implying that no small circuit $\tilde{c}$ can compute $\tilde{f}$ in the average case, we need to show that $f(x)$ is computable in $\mathsf{poly}(n)$ time for any $x \in \{0,1\}^n$, given that $\tilde{c}(y)$ is computable in $\mathsf{poly}(N)$ time for any $y \in \{0,1\}^N$.

This is exactly the local decoding problem. Let $N = 2n$ and hence we can take a code that maps k $(= 2^n)$ message bits to k^2 $(= 2^N)$ codeword bits. We want the query complexity for local decoding to be $\mathsf{poly}(n) = \mathsf{poly}(N) = \mathsf{poly}\log(k)$.

Reed-Muller codes are such locally decodable codes. We complete the proof of Theorem 2 by picking L as a RM code with F_q^m and d-degree polynomials where $q = \log k$, $m = 2\log k / \log\log k$ and $d = \sqrt{q}$.

We also have that Reed-Muller codes can be encoded in logarithmic space. This gives us an upper bound on the complexity of the hard function $\tilde{f}$.

Putting everything together, we get the following.

Theorem 2. *If* PSPACE *does not have polynomial size circuits,* PSPACE *contains a problem hard to compute in the average case for polynomial size circuits.*

2.3 Extreme hardness in the average case

One can replace the threshold 0.9 with $1/2 + \varepsilon$ for any constant $\varepsilon > 0$ in the definition of hardness in the average case (see Definition 1). A similar theorem also holds for this form of average case hardness, but this requires by means of local list decoding. See the classic paper of Sudan-Trevisan-Vadhan (also Arora-Sudan) for a definition, and for an algorithm for local list decoding Reed-Muller codes.

3 R v.s. δ tradeoff

For $\delta = 1/2 - \varepsilon$, until now we have

- there exists codes with $R = \Omega(\varepsilon^2)$ (Gilbert-Varshamov bound),
- no code can have rate $R = \Omega(\varepsilon)$ (Plotkin bound),
- there exists explicit codes with $R = \Omega(\varepsilon^3)$ (Concatenation of RS outer code and brute-forcely search inner code).

In what follows, we will show that no linear code can have rate $R = \omega(\varepsilon^2 \log(1/\varepsilon))$ using Fourier analysis. The proof is a translation into Fourier language of a very nice proof due to Alon, and an improvment due to Schechtman and Shraibman. A more sophisticated version of this argument, due to Friedman-Tillich and Navon-Samorodnitsky, yields the full "linear-programming bound" (proved first by McEliece, Rodemich, Rumsey and Welch using the theory of orthogonal polynomials), which is the best known bound on the R vs δ tradeoff (and is 40 years old!).

We begin with some basics of Fourier analysis and then derive an upper bound of rate R.

3.1 Fourier bases

For every $\alpha \in \mathbb{F}_2^n$, define $\chi_\alpha : \mathbb{F}_2^n \to \mathbb{R}$ as $\chi_\alpha(x) = (-1)^{\langle \alpha, x \rangle} = (-1)^{\sum_i \alpha_i x_i}$.

Observations

- $\chi_0(x) = 1$ for all x,
- $\sum_x \chi_\alpha(x) = \sum_x (-1)^{\sum_i \alpha_i x_i} = \prod_{i=1}^n (1 + (-1)^{\alpha_i}) = 0$ if $\alpha \neq 0$,
- $\chi_\alpha(x+y) = (-1)^{\langle \alpha, x+y \rangle} = (-1)^{\langle \alpha, x \rangle}(-1)^{\langle \alpha, y \rangle} = \chi_\alpha(x)\chi_\alpha(y)$,
- $\chi_{\alpha+\beta}(x) = \chi_\alpha(x)\chi_\beta(x)$,
- χ_α's form an orthogonal basis

$$\begin{aligned}
\langle \chi_\alpha, \chi_\beta \rangle &= \sum_x \chi_\alpha(x)\chi_\beta(x) \\
&= \sum_x \chi_{\alpha+\beta}(x) \\
&= \begin{cases} 2^n & \text{if } \alpha + \beta = 0 \text{ (i.e. } \alpha = \beta) \\ 0 & \text{if } \alpha + \beta \neq 0 \text{ (i.e. } \alpha \neq \beta) \end{cases}
\end{aligned}$$

3.2 Fourier coefficients

Given $f : \mathbb{F}_2^n \to R$, define $\widehat{f}(\alpha) = \langle f, \chi_\alpha \rangle$

Observations

$$f(x) = \frac{1}{2^n}\sum_{\alpha}\widehat{f}(\alpha)\chi_\alpha(x) \tag{1}$$

$$\begin{aligned}\langle f, g\rangle &= \frac{1}{2^n}\frac{1}{2^n}\sum_{\alpha,\beta}\widehat{f}(\alpha)\widehat{g}(\beta)\langle\chi_\alpha,\chi_\beta\rangle \\ &= \frac{1}{2^n}\sum\widehat{f}(\alpha)\widehat{g}(\beta) \qquad \text{(Parseval's identity)}\end{aligned}$$

$$\langle f, f\rangle = \frac{1}{2^n}\sum\left(\widehat{f}(\alpha)\right)^2 \tag{2}$$

3.3 Convolution of functions

Given $f, g : \mathbb{F}_2^n \to \mathbb{R}$, define $f * g : \mathbb{F}_2^n \to \mathbb{R}$ as $f * g(x) = \sum_{y+z=x} f(y)g(z)$.

Observations

$$\begin{aligned}\widehat{f * g}(\alpha) &= \langle f * g, \chi_\alpha\rangle \\ &= \sum_x f * g(x)\chi_\alpha(x) \\ &= \sum_x \sum_{y+z=x} f(y)g(z)\chi_\alpha(x) \\ &= \sum_x \sum_{y+z=x} f(y)g(z)\chi_\alpha(y+z) \\ &= \sum_y\sum_z f(y)g(z)\chi_\alpha(y+z) \\ &= \sum_y\sum_z f(y)g(z)\chi_\alpha(y)\chi_\alpha(z) \\ &= \sum_y f(y)\chi_\alpha(y)\sum_z g(z)\chi_\alpha(z) \\ &= \widehat{f}(\alpha)\widehat{g}(\alpha)\end{aligned} \tag{3}$$

3.4 ε-biased codes

In this section, we prove that any ε-biased code has rate $R = \mathcal{O}(\varepsilon^2\log(1/\varepsilon))$ and leave the discussion of the general case (i.e. $\delta = 1/2 - \varepsilon$) to Section 3.5.

Definition 3. *A linear code is ε-biased if all non-zero codewords have weight in $[1/2 - \varepsilon, 1/2 + \varepsilon]$.*

Let G be the generator matrix of code C and let $S \subseteq \mathbb{F}_2^k$ be the columns of G. Note that G is a k by n matrix. Let $\mu : \mathbb{F}_2^k \to \mathbb{R}$ and

$$\mu(x) = \begin{cases} 1 & \text{if } x \in S, \\ 0 & \text{otherwise.}\end{cases}$$

Observations

- For $\alpha \neq 0$, $\widehat{\mu}(\alpha) = \sum_{x \in \mathbb{F}_2^k} \mu(x)\chi_\alpha(x)$
$$= \frac{1}{|S|} \sum_{x \in S} (-1)^{\langle x, \alpha \rangle}$$
$$= \frac{1}{|S|} (\#\text{ of 0-coordinate in } \alpha G - \#\text{ of 1-coordinate in } \alpha G)$$
$$= \begin{cases} \in [-\varepsilon, \varepsilon] & \text{in } \varepsilon\text{-biased case} \\ \in (-1, \varepsilon) & \text{in distance } \delta = 1/2 - \varepsilon \text{ case} \end{cases} \tag{4}$$

- For $\alpha = 0$,
$$\widehat{\mu}(\alpha) = \sum_x \mu(x) = 1. \tag{5}$$

Using Parseval's identity, we get

$$\frac{1}{|S|} = \sum_x \mu^2(x) = \frac{1}{2^k} \sum_\alpha \widehat{\mu}^2(\alpha)$$
$$\leq \frac{1}{2^k}(1 + \varepsilon^2(2^k - 1)). \qquad \text{(By (5), (4))}$$

Rearranging yields

$$n = |S| = \frac{1}{\sum_x \mu^2(x)} \geq \frac{2^k}{1 + \varepsilon^2(2^k - 1)}.$$

If $\varepsilon = \mathcal{O}(2^{-k/2})$, then $n = \Omega(2^k)$ and $R = k/n = \mathcal{O}\left(\log(1/\varepsilon)/\varepsilon^2\right)$. However, it is useless for constant ε. For larger ε, we plan to reduce the bias by convolution and then apply the previous argument.

Let $\nu = \mu^{(*t)}$. Then $\widehat{\nu}(\alpha) = (\widehat{\mu}(\alpha))^t \leq \varepsilon^t$. By Parseval's identity

$$\sum_x \nu^2(x) = \frac{1}{2^k} \sum_\alpha \widehat{\nu}^2(\alpha) \leq \frac{1}{2^k} \left(1 + \varepsilon^{2t}(2^k - 1)\right), \tag{6}$$

and by Cauchy-Schwarz inequality

$$\sum_x \nu^2(x) \geq \frac{\left(\sum_x \nu(x)\right)^2}{\mathsf{support}(\nu)} = \frac{1}{\mathsf{support}(\nu)} \geq \frac{1}{\binom{|S|=n}{t}}. \tag{7}$$

Combining the upper bound (6) and the lower bound (7) of $\sum_x \nu^2(x)$, we have

$$\frac{2^k}{(1 + (2^k - 1)\varepsilon^{2t})} \leq \binom{n}{t} \leq \left(\frac{en}{t}\right)^t. \tag{8}$$

Rearranging (8), we get

$$n \geq \frac{t}{e}\left(\frac{2^k}{1+(2^k-1)\varepsilon^{2t}}\right)^{1/t}.$$

Choose t so that $\varepsilon^{2t} = 1/2^k$, then $\left(2^k\right)^{1/t} = 1/\varepsilon^2$ or equivalently $t = \frac{k}{2\log(1/\varepsilon)}$. As a result,

$$n = \Omega\left(\frac{k}{\varepsilon^2 \log(1/\varepsilon)}\right) \text{ and } R = k/n = \mathcal{O}\left(\varepsilon^2 \log(1/\varepsilon)\right).$$

3.5 Codes with $\delta = 1/2 - \varepsilon$

From (4), we know $\widehat{\mu}(\alpha) \in (-1, \varepsilon)$ for all $\alpha \neq 0$. Because now $|\widehat{\mu}^2(\alpha)|$ can be 1 rather than bounded by ε^2, we cannot reuse the analysis of ε-biased code.

We consider the new fact $\mu \geq 0 \Rightarrow \mu^{*3}(0) \geq 0$ and therefore

$$\begin{aligned} 0 \leq \mu^{*3}(0) &= \sum_{\alpha} \left(\widehat{\mu}(\alpha)\right)^3 \chi_\alpha(0) \\ &= \sum_{\alpha} \left(\widehat{\mu}(\alpha)\right)^3 \\ &= 1 + \sum_{\alpha:\ \widehat{\mu}(\alpha) < -\varepsilon} \left(\widehat{\mu}(\alpha)\right)^3 + \sum_{\alpha:\ |\widehat{\mu}(\alpha)| \leq \varepsilon} \left(\widehat{\mu}(\alpha)\right)^3 \end{aligned} \tag{9}$$

Rearranging (9), we get

$$\begin{aligned} \sum_{\alpha:\ \widehat{\mu}(\alpha) < -\varepsilon} \left(\widehat{\mu}(\alpha)\right)^3 &\geq -1 - \sum_{\alpha:\ |\widehat{\mu}(\alpha)| \leq \varepsilon} \left(\widehat{\mu}(\alpha)\right)^3 \\ &\geq -1 - \varepsilon^2(2^k - 1) \end{aligned} \tag{10}$$

To obtain a bound of $\sum_\alpha \left(\widehat{\mu}(\alpha)\right)^2$ from $\sum_\alpha \left(\widehat{\mu}(\alpha)\right)^3$, we consider

$$-\varepsilon \left(\sum_{\alpha:\ \widehat{\mu}(\alpha) < -\varepsilon} \left(\widehat{\mu}(\alpha)\right)^2 \right) \geq \sum_{\alpha:\ \widehat{\mu}(\alpha) < -\varepsilon} \left(\widehat{\mu}(\alpha)\right)^3,$$

plug-in (10), and thus

$$\sum_{\alpha:\ \widehat{\mu}(\alpha) < -\varepsilon} \left(\widehat{\mu}(\alpha)\right)^2 \leq \frac{1}{\varepsilon} + \varepsilon^2(2^k - 1).$$

Define $\nu = \mu^{(*t)}$ for some t large and odd. Since $\nu \geq 0$,

$$\sum_{\alpha:\ \widehat{\nu}(\alpha) < -\varepsilon^t} \left(\widehat{\nu}(\alpha)\right)^2 \leq \frac{1}{\varepsilon^t} + \varepsilon^{2t}(2^k - 1). \tag{11}$$

Consider $\sum_\alpha \left(\widehat{\nu}(\alpha)\right)^2$ for the case of $\alpha:\ \widehat{\nu}(\alpha) < -\varepsilon^t$ (i.e. (11)), $\alpha = 0$, and the rest of α's, we get

$$\begin{aligned}\frac{1}{\mathsf{support}(\nu)} &\leq \sum_x \nu^2(x) \\ &= \frac{1}{2^k}\sum_\alpha \widehat{\nu}^2(\alpha) \\ &\leq \frac{1}{2^k}\left(1 + \frac{1}{\varepsilon^t} + \varepsilon^{2t}(2^k-1) + \varepsilon^{2t}(2^k-1)\right) \qquad (12)\end{aligned}$$

Combining $\mathsf{support}(\nu) \leq \binom{n}{t}$, we rewrite (12) as

$$\binom{n}{t} \geq \frac{2^k}{1 + \frac{1}{\varepsilon^t} + 2\varepsilon^{2t}(2^k-1)}.$$

Choose t so that $1/\varepsilon^{3t} = 2^k$, $t = \frac{k}{3\log(1/\varepsilon)}$, $\left(2^k\right)^{1/t} = 1/\varepsilon^2$. As a result,

$$\begin{aligned}n &\geq \frac{t}{e}\left(\frac{2^k}{1 + \frac{1}{\varepsilon^t} + 2\varepsilon^{2t}(2^k-1)}\right)^{1/t} \\ &\geq \frac{t}{e}\left(\frac{2^k}{1 + 2^{k/3} + 2\cdot 2^{k/3}}\right)^{1/t} \\ &\geq \frac{t}{e}\left(\frac{2^{2k/3}}{3}\right)^{1/t} \\ &= \Omega\left(t\left(\frac{1}{\varepsilon^3}\right)^{2/3}\right) \\ &= \Omega\left(\frac{k}{\varepsilon^2\log(1/\varepsilon)}\right)\end{aligned}$$

and therefore $R = k/n = \mathcal{O}\left(\varepsilon^2\log(1/\varepsilon)\right)$.