CMPUT 675: Approximation Algorithms **Fall 2014**

Lecture 32 (Dec 1 & 3): Håstad's Max-2Lin(3) Hardness

Lecturer: Zachary Friggstad *Scribe: Zachary Friggstad*

32.1 Hardness of Max-2Lin(3)

Our final lectures cover the hardness of Max-2Lin(3), in which we are given linear constraints over integers modulo 2 where each constraint only involves three variables. Naturally, the goal is to satisfy as many constraints as possible.

New Notation
We will be talking a lot about bits and boolean functions. In our discussion, we will adopt the following (seeming unusual) convention. Bits are represented by the real numbers -1 or $+1$ with the notion that $-1 \equiv$ True and $1 \equiv$ False. An n-bit boolean function is a mapping $f : \{-1,+1\}^n \to \{-1,+1\}$.

Definition 1 (redefinition) *In* Max-2Lin(3)*, we are given variables over the real values* $\{-1,+1\}$ *and a number of constraints of the form* $x_i \cdot x_j \cdot x_k = 1$ *or* $x_i \cdot x_j \cdot x_k = -1$*. Furthermore, each constraint* C *comes with a weight* $w_C \geq 0$ *such that* $\sum_C w_C = 1$*. The goal is to assign a* $\{-1,+1\}$ *value to each variable to maximize the total weight of satisfied constraints.*

Theorem 1 *Håstad [H01] For any constant* $\epsilon > 0$ *and any language* $L \in \mathbf{NP}$*, there is a reduction from a instance* x *of the decision problem* $x \in L$*? to* Max-2Lin(3) *such that:*

- **Completeness***: If* $x \in L$*, then at least a* $(1-\epsilon)$*-weight of the constraints can be satisfied.*
- **Soundness***: If* $x \notin L$*, then at most a* $(1/2+\epsilon)$*-weight of the constraints can be satisfied.*

This is the first time we will see a hardness reduction from the PCP Theorem to a constraint satisfaction problem that does not have perfect completeness. This is necessary: using Gaussian elimination we can determine if **all** constraints can be satisfied in polynomial time by solving the system of linear equations when viewing the problem under the original "integers mod 2" perspective. So, any hardness reduction must have imperfect completeness. Given this, Theorem 1 is essentially the best possible because a random assignment satisfies half of the total weight of the constraints in expectation.

32.1.1 Preliminaries

Definition 2 *A boolean function* $f : \{-1,+1\}^n \to \{-1,+1\}$ *is* folded *if* $f(-\mathbf{z}) = -f(\mathbf{z})$ *for each* $\mathbf{z} \in \{-1,+1\}^n$*.*

Note that a folded boolean function f on n bits can be completely described using a 2^{n-1} bit string in the following way. Say the bit string records the value of $f(\mathbf{z})$ explicitly for all $\mathbf{z} \in \{-1,+1\}^n$ that have $z_1 = 1$.

Then we can compute any $f(\mathbf{z})$ value by querying this bit string once, either with $\mathbf{z}$ if $z_1 = 1$ or by computing $-f(-\mathbf{z})$ with one query to the bit string if $z_1 = -1$.

Call this string the **compact representation** of the folded function f.

32.1.2 The Reduction

For an appropriate constant $\delta > 0$ that we will describe later, we first reduce an instance x of the decision problem $x \in L$? to Set Cover with soundness $\delta \cdot |E|$ (so use $\ell = \log_2 1/\delta$). This gives us a regular graph $G = (V; E)$ with sides V_L, V_R, a set of labels Σ, and constraints $\pi_e, e \in E$. Since δ will be chosen to be a constant, then the reduction takes polynomial time and Σ has constant size.

There are $|V| \cdot 2^{|\Sigma|-1}$ variables in the Max-2Lin(3) instance we construct which will be viewed in the following way. For each $w \in V$, there are $2^{|\Sigma|-1}$ variables that we interpret as the compact representation of a folded function $f_w : \{-1, +1\}^\Sigma \to \{-1, +1\}$. In this way, a setting of particular $\{-1, +1\}$ values to the variables gives us one folded function for each $w \in V$.

Now for the constraints. For two different $\mathbf{y}, \mathbf{z} \in \{-1, +1\}^n$ we let $\mathbf{y} \cdot \mathbf{z} \in \{-1, +1\}^n$ be the bit string whose i'th coordinate is $y_i \cdot z_i$. Also, for a constraint π_e and some $\mathbf{y} \in \{-1, +1\}^\Sigma$ we let $\pi_e^{-1}(\mathbf{y}) \in \{-1, +1\}$ be the bit string where entry s is $\mathbf{y}_{\pi_e(s)}$. In other words, $\pi_e^{-1}(\mathbf{y})$ is simply the unique bit string $\mathbf{y}'$ such that $y'_{\pi(s)} = y_s$ for each $s \in \Sigma$. It is helpful to think of $\pi_e^{-1}(\mathbf{y})$ as the *preimage* bit string of $\mathbf{y}$. See Figure 32.1 for an illustration.

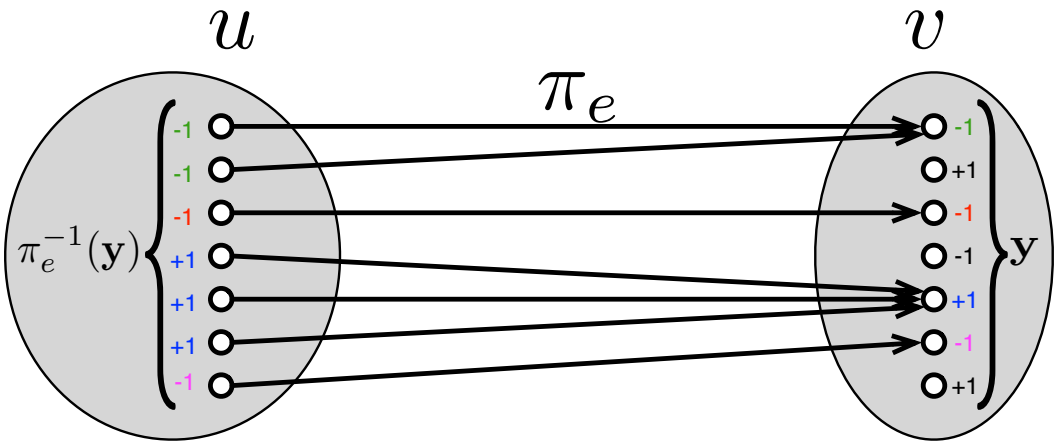

Figure 32.1: Illustration of the construction of $\pi_e^{-1}(\mathbf{y})$ for an edge $e = (u, v)$. The small nodes on each side represent labels in Σ and the arrows between them represent the mapping π_e. A specific $\mathbf{y} \in \{-1, +1\}^\Sigma$ is given on the right, and the bits of $\pi_e^{-1}(\mathbf{y})$ on the left are simply the corresponding bits of $\mathbf{y}$ when following the π_e mapping.

Algorithm 1 describes the reduction in a somewhat unusual manner. Really, it simply samples a single constraint according to a particular distribution. In the *real* reduction, a constraint C is included with weight w_C equal to the probability that C is sampled by Algorithm 1. It is a simple exercise to see this can be computed in polynomial time (recalling that $|\Sigma|$ is a constant).

In this algorithm, we call $\mathbf{z}$ an ϵ-*noise* vector because multiplying any $\mathbf{w}$ by $\mathbf{z}$ is the same as flipping each entry of $\mathbf{w}$ with probability ϵ. So, in particular, the term $\mathbf{x} \cdot \mathbf{z}$ in the last line can be viewed as a "noisy" version of $\mathbf{x}$.

Again, for emphasis, recall that the functions f_u, f_v in the last step correspond to folded functions that are

Algorithm 1 Sampling a single constraint.

Sample an edge $e = (u, v) \in E$ uniformly at random.
Sample $\mathbf{x}, \mathbf{y} \in \{-1, +1\}^\Sigma$ independently and uniformly at random.
Sample $\mathbf{z} \in \{-1, +1\}^\Sigma$ by setting $z_i = +1$ with probability $1 - \epsilon$ and $z_i = -1$ with probability ϵ.
output the constraint $f_u(\mathbf{x}) \cdot f_v(\mathbf{y}) \cdot f_u(\mathbf{x} \cdot \mathbf{z} \cdot \pi_e^{-1}(\mathbf{y}))$

described by the $2^{|\Sigma|-1}$ variables associated with u, v. What this really means for f_u (and similarly for f_v) is that for $\mathbf{x} \in \{-1, +1\}^\Sigma$ with $x_1 = 1$ the term $f_u(\mathbf{x})$ really means the variable corresponding to position $\mathbf{x}$ in the compact representation and if $x_1 = -1$ then $f_u(\mathbf{x})$ really means the negation of the variable corresponding to position $-\mathbf{x}$ of the compact representation. In the latter case, we multiply both sides of the constraint by -1 to ensure the sampled constraint is of the form $v_i \cdot v_j \cdot v_k = b$ for some $b \in \{-1, +1\}$ where v_i, v_j, v_k are the appropriate variables in the corresponding compact representations.

From now on, we ignore this low level detail and simply proceed by interpreting a variable assignment as describing folded functions $\{f_w\}_{w \in V}$.

32.1.3 Completeness

Assume $x \in L$ and let $\sigma : V \to \Sigma$ be a labelling that satisfies all constraints of the Label Cover instance.

A good solution for the Max-2Lin(3) instance is obtained by setting each boolean function $f_w : \{-1, +1\}^\Sigma \to \{-1, +1\}$ to simply the function $f_w(\mathbf{z}) = z_{\sigma(w)}$. This is called a *dictator function* because the function depends only one one coordinate. Note that a dictator function is folded.

Claim 1 *The weight of satisfied constraints is at least* $1 - \epsilon$.

Proof. We simply show that with probability $1 - \epsilon$, Algorithm 1 will output a constraint that is satisfied by this assignment.

Algorithm 1 samples $z_{\sigma(u)} = 1$ with probability $1 - \epsilon$. When this happens, the output constraint is satisfied because

$$\begin{aligned} f_u(\mathbf{x}) \cdot f_v(\mathbf{y}) \cdot f_u(\mathbf{x} \cdot \mathbf{z} \cdot \pi_e^{-1}(\mathbf{y})) &= x_{\sigma(u)} \cdot y_{\sigma(v)} \cdot x_{\sigma(u)} \cdot z_{\sigma(u)} \cdot (\pi_e^{-1}(\mathbf{y}))_{\sigma(u)} \\ &= y_{\sigma(v)} \cdot (\pi_e^{-1}(\mathbf{y}))_{\sigma(u)} \\ &= y_{\sigma(v)} \cdot y_{\pi_e(\sigma(u))} \\ &= y_{\sigma(v)} \cdot y_{\sigma(v)} \\ &= 1 \end{aligned}$$

The second last inequality is because π_e is satisfied by σ. ∎

32.1.4 Soundness

Assume $x \notin L$, so that any labelling σ for G satisfies at most $\delta \cdot |E|$ edge constraints π_e. We will show that for an appropriately small constant δ that any solution to the Max-2Lin(3) instance satisfies at most a $\left(\frac{1}{2} + \epsilon\right)$-fraction of the total constraint weight.

In some sense we want to prove a weak converse of what we saw in the completeness case. For an edge $e = (u, v)$, if a large total weight of the constraints of the Max-2Lin(3) instance associated with e were satisfied then we

hope that f_u, f_v should resemble dictator functions whose corresponding labels satisfy π_e. If so, then we can conclude that there are not many such edges e by the soundness of the Label Cover instance.

However, without the "ϵ-noise" $\mathbf{z}$ this is not true as many of the folded functions $\chi_S, S \subseteq [n]$ could also be used for f_u, f_v to have satisfy *all* of e's associated constraints (for example, using $f_u = \chi_S, f_v = \chi_T$ where for each $t \in T$, $\pi_e(s) = t$ for exactly one $s \in S$). We will see that, in some sense, the noise $\mathbf{z}$ punishes the non-dictator functions so which eliminates cheating solutions of this kind.

Also, if we did not require the functions to be folded then we could simply set each f_w to be the constant boolean function that always takes the value 1, which would satisfy every constraint. The soundness analysis will critically rely on the fact that the functions are folded and the ϵ-noise $\mathbf{z}$ is added when sampling a constraint in Algorithm 1.

From now on, suppose $\{f_w\}_{w \in V}$ are the folded functions corresponding to a particular assignment of $\{-1, +1\}$-values to the variables.

Definition 3 *Say an edge $e \in E$ is* good *if*

$$\Pr[\textit{Algorithm 1 outputs a satisfied constraint} \,|\, \textit{Algorithm 1 samples } e] \geq \frac{1}{2} + \frac{\epsilon}{2}.$$

Let g_E be the probability of sampling a good edge in the first step of Algorithm 1 (i.e. g_E is the fraction of good edges).

Similar to the Set Cover reduction, we will show that there cannot be too many good edges, thus there cannot be too many satisfied constraints.

Lemma 1 *For an appropriate choice of constant δ, $g_E \leq \epsilon/2$.*

We will prove this soon, but let us see how to conclude the soundness analysis under this assumption.

The weight of satisfied constraint is precisely the probability that Algorithm 1 outputs a constraint that is satisfied by the given solution $\{f_w\}_{w \in V}$. Consider the following two indicator random variables (over the random choices made by Algorithm 1). Let $X_E \in \{0, 1\}$ be 1 if and only if the sampled edge e is good and let $Y_C \in \{0, 1\}$ be 1 if and only if the constraint that is output is satisfied by $\{f_w\}_{w \in V}$.

$$\begin{aligned}
(\text{Total weight of constraints satisfied by } \{f_w\}_{w \in V}) &= \Pr[Y_C] \\
&= \Pr[Y_C | X_e] \cdot \Pr[X_e] + \Pr[Y_C | \neg X_e] \cdot \Pr[\neg X_e] \\
&\leq 1 \cdot g_E + \left(\frac{1}{2} + \frac{\epsilon}{2}\right) \cdot (1 - g_E) \\
&\leq \frac{\epsilon}{2} + \left(\frac{1}{2} + \frac{\epsilon}{2}\right) = \frac{1}{2} + \epsilon
\end{aligned}$$

All that is left is to prove Lemma 1. To do this, we need to introduce a new analysis tool.

32.1.5 Interlude: A Very Brief Introduction to Fourier Analysis

The soundness analysis is following the same basic idea behind the Set Cover soundness analysis we saw earlier. We identified *good* edges as those that contribute a lot to the weight of satisfied constraints. We will show that a certain random labelling $\sigma : V \to \Sigma$ will satisfy good edges with constant probability. Thus, there cannot be many good edges because the soundness in the Label Cover reduction is small.

Showing that a good edge satisfies an edge constraint with constant probability (and even defining the random labelling σ) requires very cool machinery that we have not seen yet: **Fourier analysis of boolean functions**.

Unfortunately, we do not have time to explore the many fascinating applications of this tool in hardness proofs. At least we will see one here. We will only cover the very basics, consult the in-depth book by Ryan O'Donnell if you want to learn more [O14] (you can download the .pdf from the book's webpage).

Boolean functions in vector spaces

View the set of functions $f : \{-1,+1\}^n \to \mathbb{R}$ as a 2^n-dimensional $\mathbb{R}$-vector space where $(f+g)(\mathbf{x}) = f(\mathbf{x})+g(\mathbf{x})$ and $(\alpha \cdot f)(\mathbf{x}) = \alpha \cdot f(\mathbf{x})$ for $\alpha \in \mathbb{R}$. In particular, the boolean functions $\{-1,+1\}^n \to \{-1,+1\}$ are members of this vector space.

Consider the following inner product for this space. For functions $f, g : \{-1,+1\}^n \to \mathbb{R}$ we define

$$\langle f, g\rangle = E_{\mathbf{x}}[f(\mathbf{x}) \cdot g(\mathbf{x})].$$

In this definition, $\mathbf{x}$ is sampled uniformly at random from $\{-1,+1\}^n$. It can be easily verified that $\langle \cdot, \cdot \rangle$ satisfies the properties of an inner product over this vector space.

A very nice basis

Let $[n] = \{1, \ldots, n\}$. We identify a special basis of this vector space called the *Fourier basis*. For any $S \subseteq [n]$ consider the boolean function $\chi_S : \{-1,+1\}^n \to \{-1,+1\}$ given by $\chi_S(\mathbf{x}) = \prod_{i \in S} x_i$. To be clear, $\chi_\emptyset$ is the constant boolean function that always takes the value 1.

Lemma 2 *The functions $\{\chi_S\}_{S\subseteq[n]}$ form an orthonormal basis for the space of $\{-1,+1\}^n \to \mathbb{R}$ functions with respect to this inner product $\langle \cdot, \cdot \rangle$.*

Proof. We have $1 = f(\mathbf{x})^2$ for any $\mathbf{x}$ and any boolean function f, so $\langle f, f\rangle = 1$. In particular, it holds for the χ_S functions.

Now consider different $S, T \subseteq \{1, \ldots, n\}$ and say $i \in S - T$ (if $S \subseteq T$ then simply swap the roles of T and S in this proof). We have $\chi_S(\mathbf{x}) = \chi_T(\mathbf{x})$ for exactly half of the inputs $\mathbf{x} \in \{-1,+1\}^n$ because if $\mathbf{x}'$ is obtained by negating only the i'th bit of $\mathbf{x}$ then $\chi_S(\mathbf{x}) \neq \chi_S(\mathbf{x}')$ while $\chi_T(\mathbf{x}) = \chi_T(\mathbf{x}')$. Thus, $\mathrm{E}_{\mathbf{x}}[\chi_S(\mathbf{x}) \cdot \chi_T(\mathbf{x})] = 0$.

This shows $\{\chi_S\}_{S\subseteq[n]}$ is an orthonormal collection. There are exactly 2^n of them (one for each $S \subseteq [n]$) and the vector space has dimension 2^n, so they in fact form a basis for this space. ∎

Fourier coefficients

By Lemma 2, every function $f : \{-1,+1\} \to \mathbb{R}$ can be uniquely decomposed as a linear combination of the functions $\{\chi_S\}_{S\subseteq[n]}$. That is, we can write $f = \sum_{S\subseteq[n]} \widehat{f}(S) \cdot \chi_S$ where each $\widehat{f}(S) \in \mathbb{R}$. These $\widehat{f}(S)$ values are the *Fourier coefficients* of f.

Example 1 *Let $f : \{-1,+1\}^2 \to \{-1,+1\}$ be the* AND *function, so $f(x_1, x_2) = \max\{x_1, x_2\}$. It is easy to verify that*

$$f = \frac{\chi_\emptyset}{2} + \frac{\chi_{\{1\}}}{2} + \frac{\chi_{\{2\}}}{2} - \frac{\chi_{\{1,2\}}}{2},$$

so $\widehat{f}(S) = \frac{1}{2}$ for $S \neq \{1,2\}$ and $\widehat{f}(\{1,2\}) = -\frac{1}{2}$.

Lemma 3 *For any $f, g : \{-1,+1\}^n \to \mathbb{R}$ we have $\langle f, g\rangle = \sum_{S\subseteq[n]} \widehat{f}(S)\cdot\widehat{g}(S)$.*

Proof.

$$\begin{aligned}
\langle f, g\rangle &= \mathrm{E}_{\mathbf{x}}[f(\mathbf{x})\cdot g(\mathbf{x})] \\
&= \mathrm{E}_{\mathbf{x}}\left[\left(\sum_{S\subseteq[n]} \widehat{f}(S)\cdot\chi_S(\mathbf{x})\right)\cdot\left(\sum_{T\subseteq[n]} \widehat{g}(T)\cdot\chi_T(\mathbf{x})\right)\right] \\
&= \sum_{S,T\subseteq[n]} \widehat{f}(S)\cdot\widehat{g}(T)\cdot\mathrm{E}_{\mathbf{x}}[\chi_S(\mathbf{x})\cdot\chi_T(\mathbf{x})] \\
&= \sum_{S,T\subseteq[n]} \widehat{f}(S)\cdot\widehat{g}(T)\cdot\langle\chi_S,\chi_T\rangle \\
&= \sum_{S\subseteq[n]} \widehat{f}(S)\cdot\widehat{g}(S)
\end{aligned}$$

The last equality uses the fact that the Fourier basis is orthonormal. ■

Finally, there is a convenient way to compute Fourier coefficients of a function.

Lemma 4 *For any $f : \{-1,+1\}^n \to \mathbb{R}$ and any $S \subseteq [n]$, we have $\langle f, \chi_S\rangle = \widehat{f}(S)$. In particular, we have $\mathrm{E}_{\mathbf{x}}[f(\mathbf{x}] = \widehat{f}(\emptyset)$.*

Proof.

$$\langle f,\chi_S\rangle = \left\langle \sum_{R\subseteq[n} \widehat{f}(R)\cdot\chi_R, \chi_S\right\rangle = \sum_{R\subseteq[n]} \widehat{f}(R)\cdot\langle\chi_R,\chi_S\rangle = \widehat{f}(S)$$

In particular, $\mathrm{E}_{\mathbf{x}}[f(\mathbf{x})] = \langle f,\chi_\emptyset\rangle = \widehat{f}(\emptyset)$. ■

The Fourier distribution of a boolean function
Let $f : \{-1,+1\}^n \to \{-1,+1\}$ be a boolean function. By Lemma 3 and the fact $f(\mathbf{x})^2 = 1$ for each $\mathbf{x} \in \{-1,+1\}^n$ we have $1 = \sum_{S\subseteq[n]} \widehat{f}(S)^2$. This, plus the obvious fact that $\widehat{f}(S)^2 \geq 0$ for each $S \subseteq [n]$, suggests a probability distribution over subsets of $[n]$.

Definition 4 *The* Fourier distribution *of f over subsets of $[n]$ is the distribution that places weight $\widehat{f}(S)^2$ on $S \subseteq [n]$.*

32.1.6 There Are Few Good Edges

We focus on proving Lemma 1. To do this, we randomly construct a labelling $\sigma : V \to \Sigma$ such that every good edge e has π_e satisfied with constant probability.

The way to sample σ is elegant. For each $w \in V$, we set $\sigma(w) \in \Sigma$ by first sampling some $S \subseteq [n]$ from the Fourier distribution of f_w (i.e. S is chosen with probability $\widehat{f_w}(S)^2$). By Lemma 4 and the fact that f_w is folded, $\widehat{f_w}(\emptyset) = 0$ so $S \neq \emptyset$. We then select $\sigma(w)$ to be a uniformly chosen random element of S.

Our main technical lemma is the following.

Lemma 5 *If e is good, then over this random construction of σ we have* $\Pr[\sigma \text{ satisfies } \pi_e] \geq \epsilon^3$.

If so, then we conclude by noting

$$\epsilon^3 \cdot g_E \leq \mathrm{E}[\text{fraction of } \pi_e \text{ satisfied by } \sigma] \leq \delta.$$

so $g_E \leq \delta/\epsilon^3$. If we choose $\delta = \epsilon^4/2$ then we have $g_E \leq \epsilon/2$, as desired.

Proof of Lemma 5. Let e be a good edge. For any $T \subseteq [n]$, let $\alpha_1(T) = \{s \in \Sigma : \pi_e(s') = s \text{ for some } s' \in T\}$. So, for any $s \in \alpha_1(T)$ if we choose $s' \in T$ uniformly at random then $\Pr[\pi_e(s') = s] \geq \frac{1}{|T|}$. Therefore,

$$\Pr[\sigma \text{ satisfies } \pi_e] \geq \sum_{\substack{T \subseteq [n] \\ S \subseteq \alpha_1(T)}} \widehat{f_u}(T)^2 \cdot \widehat{f_v}(S)^2 \cdot \frac{1}{|T|} \tag{32.1}$$

We will lower bound this expression by ϵ^3.

Now we employ a useful trick: *arithmetizing* the probability that a Algorithm 1 outputs a constraint that is satisfied by f_u, f_v, given that it samples e. The function in the first expected value statement below is 0 if the sampled constraint is not satisfied and is 1 if the constraint is satisfied. This, plus the fact that e is good, justifies the first inequality.

$$\begin{aligned}
\frac{1}{2} + \frac{\epsilon}{2} &\leq \mathrm{E}_{\mathbf{x},\mathbf{y},\mathbf{z}}\left[\frac{1 + f_u(\mathbf{x}) \cdot f_v(\mathbf{y}) \cdot f_u(\mathbf{x} \cdot \mathbf{z} \cdot \pi_e^{-1}(\mathbf{y}))}{2}\right] \\
&= \frac{1}{2} + \frac{1}{2} \cdot \mathrm{E}_{\mathbf{x},\mathbf{y},\mathbf{z}}\left[\left(\sum_{R \subseteq [n]} \widehat{f_u}(R) \cdot \chi_R(\mathbf{x})\right) \cdot \left(\sum_{S \subseteq [n]} \widehat{f_v}(S) \cdot \chi_S(\mathbf{y})\right) \cdot \left(\sum_{T \subseteq [n]} \widehat{f_u}(T) \cdot \chi_T(\mathbf{x} \cdot \mathbf{z} \cdot \pi_e^{-1}(\mathbf{y}))\right)\right] \\
&= \frac{1}{2} + \frac{1}{2} \cdot \sum_{R,S,T \subseteq [n]} \widehat{f_u}(R) \cdot \widehat{f_v}(S) \cdot \widehat{f_u}(T) \cdot \mathrm{E}_{\mathbf{x},\mathbf{y},\mathbf{z}}[\chi_R(\mathbf{x}) \cdot \chi_S(\mathbf{y}) \cdot \chi_T(\mathbf{x} \cdot \mathbf{z} \cdot \pi^{-1}(\mathbf{y}))] \\
&= \frac{1}{2} + \frac{1}{2} \cdot \sum_{R,S,T \subseteq [n]} \widehat{f_u}(R) \cdot \widehat{f_v}(S) \cdot \widehat{f_u}(T) \cdot \mathrm{E}_{\mathbf{x}}[\chi_R(\mathbf{x}) \cdot \chi_T(\mathbf{x})] \cdot \mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))] \cdot \mathrm{E}_{\mathbf{z}}[\chi_T(\mathbf{z})]
\end{aligned}$$

Simply expand each f_u and f_v in terms of their Fourier coefficients to get the first equality. The last equality is justified because the values $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are sampled independently and because $\chi_S(\mathbf{x} \cdot \mathbf{y}) = \chi_S(\mathbf{x}) \cdot \chi_S(\mathbf{y})$ for any $S, \mathbf{x}, \mathbf{y}$ simply by definition of χ_S.

We can perform some quick simplifications. First, because $\mathbf{x}$ is sampled uniformly from $\{-1, +1\}^n$ we have

$$\mathrm{E}_{\mathbf{x}}[\chi_R(\mathbf{x}) \cdot \chi_T(\mathbf{x})] = \langle \chi_R, \chi_T \rangle$$

which, by Lemma 2, is 1 if $R = T$ and 0 if $R \neq T$. This simplifies the last expression to

$$\frac{1}{2} + \frac{1}{2} \cdot \sum_{S,T \subseteq [n]} \widehat{f_u}(T)^2 \cdot \widehat{f_v}(S) \cdot \mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))] \cdot \mathrm{E}_{\mathbf{z}}[\chi_T(\mathbf{z})].$$

Next, because the bits of $\mathbf{z}$ are set independently to 1 with probability $1 - \epsilon$ and to -1 with probability ϵ, then we have $\mathrm{E}_{\mathbf{z}}[\chi_T(\mathbf{z})] = \prod_{s \in T} \mathrm{E}[z_i] = (1 - 2\epsilon)^{|T|}$ so we further simplify this expression to

$$\frac{1}{2} + \frac{1}{2} \cdot \sum_{S,T \subseteq [n]} \widehat{f_u}(T)^2 \cdot \widehat{f_v}(S) \cdot (1 - 2\epsilon)^{|T|} \cdot \mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))].$$

In other words, we have shown

$$\epsilon \leq \sum_{S,T\subseteq[n]} \widehat{f_u}(T)^2 \cdot \widehat{f_v}(S) \cdot (1-2\epsilon)^{|T|} \cdot \mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))] \tag{32.2}$$

Now we consider the term $\mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))]$. We claim for each S that this is 1 for precisely one T and is 0 for the other T, but which T?

Consider some $s \in \Sigma$. The bit y_s appears some number of times in the product

$$\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}) = \prod_{a\in S} y_a \cdot \prod_{a\in T} y_{\pi_e(a)}.$$

Let Q be the labels $s \in \Sigma$ such that y_s appears an odd number of times in the last expression. Then because $y_s^2 = 1$ for each $s \in \Sigma$ we have

$$\mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))] = \mathrm{E}_{\mathbf{y}}\left[\prod_{s\in Q} y_s\right] = \prod_{s\in Q} \mathrm{E}[y_s]$$

which is 1 if $Q = \emptyset$ and 0 if $Q \neq \emptyset$.

For each $T \subseteq [n]$, consider the set $\alpha_2(T) := \{s \in \Sigma : \pi_e(s') = s \text{ for an odd number of } s \in T\}$. See Figure 32.2 for an illustration of $\alpha_2(T)$.

Important Note: $\alpha_2(T) \subseteq \alpha_1(T)$ because for each $s \in \alpha_2(T)$ there are an odd number of $s' \in T$ such that $\pi_e(s') = s$. In particular, there is at least one such $s' \in T$ for each $s \in \alpha_2(T)$.

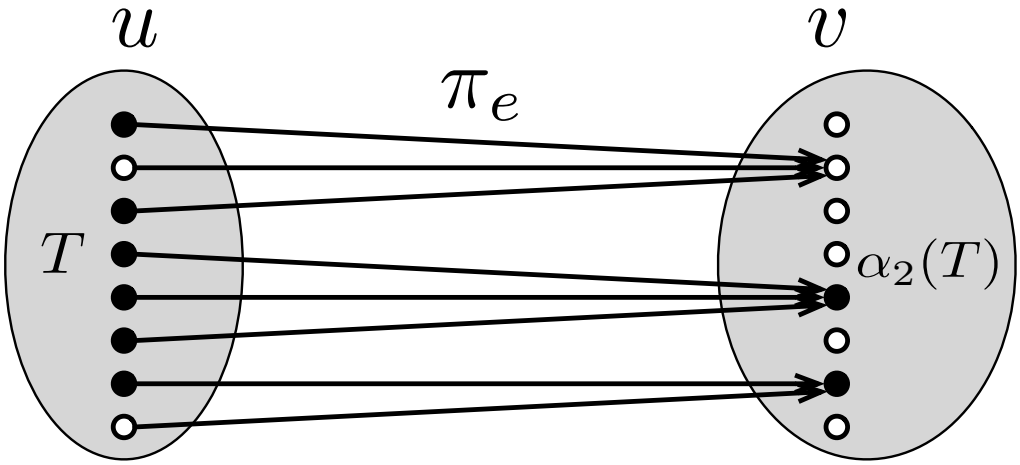

Figure 32.2: Illustration of $\alpha_2(T)$. The solid nodes on the left are the members of T and the solid nodes on the right are members of $\alpha_2(T)$. Each $s \in \alpha_2(T)$ has an odd number of preimages (under π_e) that lie in T.

From the above discussion, we see that $\mathrm{E}_{\mathbf{y}}[\chi_S(\mathbf{y}) \cdot \chi_T(\pi_e^{-1}(\mathbf{y}))] = 1$ if $S = \alpha_2(T)$, otherwise it is 0. So, (32.2) further simplifies to

$$\epsilon \leq \sum_T \widehat{f_u}(T)^2 \cdot \widehat{f_v}(\alpha_2(T)) \cdot (1-2\epsilon)^{|T|} \leq \sum_T \widehat{f_u}(T)^2 \cdot |\widehat{f_v}(\alpha_2(T))| \cdot \frac{1}{\sqrt{\epsilon \cdot |T|}} \tag{32.3}$$

The last bound is justified by recalling $1 - x \leq e^{-x}$ for all $x \geq 0$ and the easy-to-prove bound $\sqrt{y} \cdot e^{-2y} \leq 1$.

Rearranging (32.3), we have

$$\epsilon^{3/2} \leq \sum_T \widehat{f_u}(T)^2 \cdot |\widehat{f_v}(\alpha_2(T))| \cdot \frac{1}{\sqrt{|T|}} \leq \left(\sum_T \widehat{f_u}(T)^2\right)^{1/2} \cdot \left(\sum_T \widehat{f_u}(T)^2 \cdot \widehat{f_v}(\alpha_2(T)))^2 \cdot \frac{1}{|T|}\right)^{1/2} \tag{32.4}$$

where the last step uses the Cauchy-Schwarz inequality (Theorem 2, proven below).

The first sum on the right-hand side is just 1 by Lemma 3. Squaring both sides shows

$$\epsilon^3 \leq \sum_T \widehat{f_u}(T)^2 \cdot \widehat{f_v}(\alpha_2(T)))^2 \cdot \frac{1}{|T|}.$$

Because $\alpha_2(T) \subseteq \alpha_1(T)$, then by (32.1) we finally see $\epsilon^3 \leq \Pr[\sigma \text{ satisfies } \pi_e]$.

■

32.1.7 The Cauchy-Schwarz Inequality

Theorem 2 *Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ and let $\mathbf{a} \circ \mathbf{b}$ denote the usual dot product and $||\mathbf{a}|| = \sqrt{\mathbf{a} \circ \mathbf{a}}$ the standard norm of a vector. Then $|\mathbf{a} \circ \mathbf{b}| \leq ||\mathbf{a}|| \cdot ||\mathbf{b}||$.*

Proof. This is trivial if $\mathbf{b} = \mathbf{0}$, so assume otherwise. For any $t \in \mathbb{R}^n$ we have

$$0 \leq (\mathbf{a} - t \cdot \mathbf{b}) \circ (\mathbf{a} - t \cdot \mathbf{b}) = t^2 \cdot (\mathbf{b} \circ \mathbf{b}) - 2 \cdot t \cdot (\mathbf{a} \circ \mathbf{b}) + (\mathbf{a} \circ \mathbf{a}).$$

Select t to minimize the quadratic, rearrange the inequality, and take square roots.

This bound was applied in (32.4) by considering $a_T = \widehat{f_u}(T)$ and $b_T = \widehat{f_u}(T) \cdot |\widehat{f_v}(T)| \cdot \frac{1}{\sqrt{|T|}}$. ■

32.1.8 Summary

The number of variables is $|V| \cdot 2^{|\Sigma|-1}$. Since $|\Sigma| = c^\ell$ for some constant c, since $\ell = \log_2 \frac{1}{\delta}$ with $\delta = \epsilon^4/2$, and since $|V| = |x|^{O(\ell)}$, then the number of variables is polynomial in $|x|$. Also, it is easy to see how to generate all constraints and their weights in $|x|^{O(\ell)}$ time as well. So, the running time and size of the resulting instance is $|x|^{O(\log \epsilon^{-1})}$.

There is one technicality we have to address. The reduction is to Max-2Lin(3) but some of the constraints that are output in Algorithm 1 may depend on less than three different variables. This happens precisely when $\mathbf{x} = \pm\mathbf{x} \cdot \mathbf{z} \cdot \pi_e^{-1}(\mathbf{y})$ (which causes the same two bits of the compact representation of the folded function f_u to be included in the constraint) or, equivalently, when $\mathbf{z} = \pm\pi_e^{-1}(\mathbf{y})$.

This happens with very low probability: one way to see this is to dig into the Label Cover reduction a bit to see that the mappings $\pi_e : \Sigma \to \Sigma$ are at most b^ℓ-to-1 for some constant $b < c$ (here, $c^\ell = |\Sigma|$). For example, the reductions to Label Cover in both the Williamson and Shmoys and the Vazirani text have this property. Then for every $\mathbf{z}$ the probability that $\pi_e^{-1}(\mathbf{y}) = \pm\mathbf{z}$ for the random choice of $\mathbf{y}$ is at most $2 \cdot 2^{-(c/b)^\ell}$. For the same choice of ℓ, this is very small. Certainly less than $\epsilon/2$.

So, if we remove the constraints output by Algorithm 1 that involve less than three different variables and renormalize the weights, the completeness is at least $1 - \frac{3}{2} \cdot \epsilon$ and the soundness is $\frac{1/2+\epsilon}{1-\epsilon/2} \leq \frac{1}{2} + 3 \cdot \epsilon$. To get the

$1-\epsilon$ vs. $1/2+\epsilon$ hardness gap, we just have to scale ϵ by $\frac{1}{3}$ before running the reduction and removing the bad constraints.

32.2 Discussion

In an earlier lecture, we used Theorem 1 to establish a $1-\epsilon$ vs. $7/8+\epsilon$ hardness gap for Max-3SAT. Håstad also gives a more direct reduction from Label Cover that is in the same spirit as Algorithm 1 but tailored for Max-3SAT and establishes a hardness gap of 1 vs. $7/8+\epsilon$ [H99]. That is, if $x \in L$ then the resulting Max-3SAT instance is satisfiable.

Tools from the analysis of boolean functions and, in particular, Fourier analysis are indispensable in proving excellent (and often tight) hardness of approximation bounds for constraint satisfaction problems. We cannot survey the many important applications here. The interested reader is encouraged to consult O'Donnell's book [O14] on the topic to learn more of the foundations of boolean function analysis.

A great follow-up for learning more about these techniques is the Unique-Games hardness of approximating undirected Max Cut better than the constant $\min_{\theta \in [0,\pi]} \frac{2}{\pi} \cdot \frac{\theta}{1-\cos\theta} \approx 0.878$. The original paper by Khot et al. is a good place to start [KKMO07]. I also found the lecture notes from a course taught by Harsha to be quite helpful [H10].

References

H99 J. Håstad, Some optimal inapproximability results, Journal of the ACM, 48(4):798–859, 2001.

KKMO07 S. Khot, G. Kindler, E. Mossel, R. O'Donnell, Optimal inapproximability results for MAX-CUT and other 2-variable CSPs?, SIAM Journal on Computing, 37(1):319–357, 2007.

H10 P. Harsha, Unique-Games Hardness of MAXCUT, lecture notes, 2010
`http://www.tcs.tifr.res.in/~prahladh/teaching/2009-10/limits/lectures/lec12.pdf`

O14 R. O'Donnell, *Analysis of Boolean Functions*, Cambridge University Press, 2014.
`http://analysisofbooleanfunctions.org/`