Tutorial on Approximability of CSPs
Lecture 3-4: Inapproximability: Long Code based reductions, Unique Games-hardness

VENKATESAN GURUSWAMI*

June 2011

1 Overview

The PCP theorem implies — and is in fact equivalent to — the statement that $(\alpha, 1)$-distinguishing 3-SAT is NP-hard for some absolute constant $\alpha < 1$. In turn, via the standard NP-hardness reductions from 3SAT, this implies the APX-hardness of other CSPs. In fact, in all cases where the satisfiability of a CSP is known to NP-hard, the $(\gamma, 1)$-distinguishing problem is also NP-hard for some constant $\gamma < 1$. The value of α given by the PCP theorem is very close to 1, and thus these results are only able to show rather weak inapproximability bounds.

Following the PCP theorem, a natural and important research goal was to improve the inapproximability factors and hopefully determine their optimal value (the "Dream Goal" mentioned in the first part of the lectures). This pursuit has been remarkably successful, and it is the purpose of these lectures to discuss a small sample of striking results in this vein and give a taste of the basic methods used to prove them. In particular, we will highlight the general paradigm involving reductions from Label Cover (or its more structured variant, Unique Games), Long code encodings, dictatorship testing, and analysis using Fourier spectrum and invariance principle, that is used to prove strong, and often optimal, inapproximability results for CSPs.

1.1 Label Cover – the mother of strong inapproximability results

The PCP and parallel repetition theorems together imply that $(\varepsilon, 1)$-distinguishing Label Cover is NP-hard for all $\varepsilon > 0$. Recall that Label Cover is a binary CSP over a large domain D, which we will identify with $\{1, 2, \ldots, R\}$,[1] where the constraint relations are "projection functions" where the label to one variable determines that of the other. Further, for convenience, we will assume that the instance is bipartite with projection maps going from vertices on one side to the other. Formally, an instance of Label Cover(R) is given by a bipartite graph $G = (V, W, E)$, with a constraint

*Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213. Email: guruswami@cmu.edu

[1] Somehow this is the most commonly used notation in the literature, so we will stick with it.

$\pi_{w\to v} : [R] \to [R]$ for each edge $(v, w) \in E$. A labeling $\ell : V \cup W \to [R]$ satisfies an edge $(v, w) \in E$ if $\pi_{w\to v}(\ell(w)) = \ell(v)$. The value of a labeling is the fraction of edges it satisfies, and the value of the Label Cover instance is the maximum value of any labeling.

For all $\varepsilon > 0$, there exists $R = R(\varepsilon)$ such that is NP-hard to tell if the value of a Label Cover(R) instance is 1 or at most ε. This hardness result needs a domain size $R = \mathrm{poly}(1/\varepsilon)$. (In fact, one *needs* $R \geq 1/\varepsilon$ for such a result, why?) Thus, while the inapproximability factor is excellent, the domain size is large.

The approach which has been very successful is to reduce the $(\varepsilon, 1)$-distinguishing problem for Label Cover to the problem of approximating other more natural CSPs like 3-SAT, 3-LIN, 2-SAT, NAE-3-SAT, 3-COLOR, etc. In effect, one trade-offs the large gap (ε vs. 1) for a smaller one (like $7/8 + \varepsilon$ vs. 1 for 3-SAT), while reducing the domain size (eg. from $\mathrm{poly}(1/\varepsilon)$ to 2).

In developing these reductions, it is fruitful to think of a Label Cover instance as a game[2] between two (non-communicating) provers and a verifier, where the verifier picks a random edge (v, w) of the Label Cover instance, asks the first prover for the label $\ell(v)$ of vertex v, and the second prover for the label of w, and checks that their answers satisfies the constraint $\pi_{w\to v}(\ell(w)) = \ell(v)$. The provers can decide beforehand about their strategy, but cannot communicate with each other after the verifier asks the question.

Exercise 1. *Convince yourself that the value of this game, i.e., the maximum acceptance probability of the verifier under the optimal strategy for the provers, is exactly equal to the optimum value of the Label Cover instance.*

With this view, the idea is to perform the check $\pi_{w\to v}(\ell(w)) = \ell(v)$ by reading far fewer bits than the full labels (which come from a $\mathrm{poly}(1/\varepsilon)$-size domain). Of course, this is impossible as such, and so the verifier expects the labels to be encoded in a redundant form (using a special error-correcting code) that aids in performing this check efficiently by querying few bits. The idea is to trade-off "soundness" — the verifier's acceptance probability on at most ε-satisfiable instances of Label Cover — for a big savings in the number of bits it asks the provers.

1.2 Long Code and reducing Label Cover to 3-SAT

The first question to ask is what code one should use to encode labels from $[R] = \{1, 2, \ldots, R\}$. The brilliant insight of Bellare, Goldreich, and Sudan was to suggest the *Long Code*. The Long Code is a very redundant code (in some sense, the *most* redudant code). It encodes elements of $[R]$ (i.e., $\log R$ bits of information) into a codeword of length 2^R, with one position corresponding to each Boolean function on $[R]$ (equivalently, strings in $\{0, 1\}^R$). When R is a constant, we can afford such a redundant encoding. Intuitively, the usefulness of the Long Code arises from the fact that the value of any desired function of a can be obtained by a single query into its long code encoding.

Definition 1. *The (Boolean) Long Code encoding the domain $[R]$ is defined as follows. For $a \in [R]$, its long code encoding $\mathsf{LONG}_a : \{0, 1\}^R \to \{0, 1\}$ is defined by*

$$\mathsf{LONG}_a(x) = x_a \ \ \forall x \in \{0, 1\}^R .$$

[2]This explains the term Unique "Game" for Label Cover when the functions are bijections.

In other words, the long code encoding consists of the value of every Boolean function on $[R]$ at a.

Long codes are also called Dictators, *since the function* LONG_a *depends only on the a'th coordinate of the bit vector x and ignores all other coordinates.*

How does one use the long code to reduce Label Cover to some other CSP, say 3-SAT? The idea is to perform the Label Cover test $\pi(b) = a$ for some function $\pi : [R] \to [R]$ given purported long code encodings of labels a and b. Specifically, given oracle access to functions $f, g : \{0,1\}^R \to \{0,1\}$, we must test, with good confidence, if f (resp. g) looks like the long code encodings of some $a \in [R]$ (resp. $b \in [R]$) satisfying $\pi(b) = a$. And we must do this by probing three positions in f, g that are randomly chosen from some cleverly designed distribution, and checking a 3-SAT constraint on these bits.

For instance, we can read $f(x)$, $g(y)$ and $g(z)$ and check the 3-SAT constraint $(f(x) \vee g(y) \vee g(z))$. How should x, y, z be picked? One requirement x, y, z should satisfy is that legal long codes of consistent labels (a, b that obey $\pi(b) = a$) are guaranteed to pass the test. This means that for each $j \in [R]$, we must have $y_j \vee z_j \vee x_{\pi(j)}$ is true. (Why?) The crux is to show that if $(f(x) \vee g(y) \vee g(z))$ is true with good probability ($7/8 + \delta$ for a tight hardness result) over the choice of (x, y, z), then f, g can be "decoded" into labels a, b satisfying $\pi(b) = a$. It suffices for this decoding to be probabilistic and to succeed with some small but positive probability (that depends on δ but not on R). This is due to the fact that the Label Cover instance is assumed to be at most ε-satisfiable and we can assume ε is sufficiently small (compared to δ) with an appropriately large choice of R. A convenient way to get such a probabilistic decoding is to *list decode* f, g into a small set of labels $L_f, L_g \subseteq [R]$ such that there exists $a \in L_f$ and $b \in L_g$ satisfying $\pi(b) = a$. Picking a random element from L_f, L_g independently then gives the desired decoding that produces consistent labels with non-negligible probability.

The astute reader may have noticed that a test like the above one can't work, since simply taking all the tables f, g to be constant 1 functions will satisfy every 3-SAT constraint! Fortunately, a simple (but quite powerful) idea called *folding* gives a work-around this problem. The idea is to use $\overline{f(\overline{x})}$ in place of $f(x)$ in the test with probability $1/2$ (and similarly for $g(y)$ and $g(z)$, independently). In the case f is a legal long code, it obeys the folding requirement that $f(\overline{x}) = \overline{f(x)}$, so this change still maintains the validity (or completeness) of the test. Note that with this change, the constant 1 functions are accepted with probability $7/8$.

Implementing the above methodology for 3-SAT requires substantial technical effort, and this was achieved by Håstad who proved the following striking theorem.

Theorem 1. *For every $\delta > 0$, $(7/8 + \delta, 1)$-distinguishing 3-SAT is NP-hard.*

Moreover, the hardness holds when each variable has exactly 3 literals. Note that in this case simply picking a random assignment satisfied an expected fraction $7/8$ of constraints (and thus every instance is at least $7/8$-satisfiable). This algorithm can also be derandomized using the method of conditional expectations to find a $7/8$-satisfying assignment deterministically.

The proof of the above theorem is quite difficult. Instead, we turn to Håstad's tight result for 3-LIN, which is technically easier to prove, and now one of the classic results of the field.

2 A tight inapproximability result for 3-LIN

In this section we will sketch the key elements in the proof of the following theorem due to Håstad.

Theorem 2. *For every $\gamma, \varepsilon > 0$, $(1/2 + \gamma, 1 - \varepsilon)$-distinguishing 3-LIN is NP-hard.*

Note that $(1, 1)$-approximating 3-LIN is easy, as if a system of linear equations is satisfiable, then a satisfying assignment can be found in polynomial time via Gaussian elimination. Hence the "imperfect completeness" $1 - \varepsilon$ in the above theorem is necessary. Also the factor $1/2 + \gamma$ is best possible, since a random assignment would satisfy a fraction $1/2$ of the linear equations in expectation.

Via a simple gadget reduction replacing $x \oplus y \oplus z = 0$ by the 4 3-SAT constaints

$$(\bar{x} \vee \bar{y} \vee \bar{z}),\ (\bar{x} \vee y \vee z),\ (x \vee \bar{y} \vee z),\ (x \vee y \vee \bar{z})$$

and similarly for constraints $x \oplus y \oplus z = 1$, the above implies the following (which is weaker than Theorem 1 in that the hardness is only shown for near-satisfiable instances).

Corollary 3. *For every $\gamma, \varepsilon > 0$, $(7/8 + \gamma, 1 - \varepsilon)$-distinguishing 3-SAT is NP-hard.*

Remark 1 (Approximation Resistance)**.** The hardness results for approximating 3-SAT and 3-LIN show that these CSPs are *approximation resistant*, in the sense that it is hard, in the worst-case, to improve on the performance ratio achieved by the naive algorithm that simply picks a random assignment (without any regard to the structure of the instance). Håstad's machinery was able to show that many important CSPs are approximation resistant, such as k-SAT for $k \geq 3$, linear equations modulo p, Not-all-Equal k-SAT for $k \geq 4$, etc. On the other hand, it is known that *binary* CSPs (of arity two) are never approximation resistant. Tight hardness results for binary CSPs are only known under the Unique Games Conjecture.

For simplicity, though we stress that it is *not* needed, we will assume the Unique Games conjecture (UGC), and reduce from the problem of $(\delta, 1 - \varepsilon)$-distinguishing Unique Games (UG). This will allow us to illustrate the importance of *Dictatorship testing* and Fourier analysis in proving strong inapproximability results for CSPs.

2.1 Reduction from Unique Games

Assume we are given a *bi-regular* instance of Unique Games $G = (V, W, E)$ over domain $[R]$, and bijection constraints $\pi_{w \to v} : [R] \to [R]$ for each edge $(v, w) \in E$. A labeling $\ell : V \cup W \to [R]$ satisfies an edge $(v, w) \in E$ if $\pi_{w \to v}(\ell(w)) = \ell(v)$. The value of a labeling is the fraction of edges it satisfies, and the value of the UG instance is the maximum value of any labeling.

For a node $v \in V$, we denoted by $N(v) \subseteq W$ the set of its neighbors. We will reduce such a UG instance to an instance of the CSP 3-LIN, describing the instance as a verifier checking 3-LIN constraints involving positions of supposed long codes of labels to W.

Remark 2. The resulting CSP instance will be a weighted instance, with weights of constraints corresponding to the probability with which they are sampled by the verifier. The weights therefore sum up to 1. It is known how to convert the weighted instance to an unweighted instance

while preserving the approximation factor in fairly general settings (for all CSPs without unary constraints), so we will be content with obtaining the hardness for weighted instances. The acceptance probability of the verifier on any proof corresponds to the weight of 3-LIN constraints satisfied by the assignment given by that proof.

Assume for each $w \in W$, the prover provides a table $f_w : \{0,1\}^R \to \{0,1\}$, supposedly the long code encoding of the label $\ell(w)$ from a good labeling ℓ to the UG instance. (The positions in these tables correspond to the variables of the 3-LIN instance.) The verifier operates as follows:

1. Pick $v \in V$ at random.
2. Pick $w_1, w_2, w_3 \in N(v)$ uniformly and independently at random (i.e., sample with replacement).
3. Pick $x, y \in \{0,1\}^R$ uniformly and independently at random. Pick $\mu \in \{0,1\}^R$ from the ε-biased distribution, i.e., for each $i \in [R]$ independently, $\mu_i = 1$ with prob. ε and 0 with prob. $1-\varepsilon$. Set $z = x \oplus y \oplus \mu$ (the $\oplus$ of vectors is defined coordinate-wise).
4. (Pull back x, y, z according to the bijections) Let $x' = x \circ \pi_{w_1 \to v}$, i.e., $x'_j = x_{\pi_{w_1 \to v}(j)}$, and similarly let $y' = y \circ \pi_{w_2 \to v}$ and $z' = z \circ \pi_{w_3 \to v}$.
5. With probability $1/2$, check that

$$f_{w_1}(x') \oplus f_{w_2}(y') \oplus f_{w_3}(z') = 0 \tag{1}$$

and with probability $1/2$ check that

$$f_{w_1}(x') \oplus f_{w_2}(y') \oplus f_{w_3}(\overline{z'}) = 1 \tag{2}$$

(This part is the "folding" that is employed to get around the all 0's solution, which will clearly always satisfy (1).)

2.2 Analysis of the 3-LIN verifier

Let us first argue the easy completeness part.

Lemma 4. *If the UG instance has a labeling satisfying $(1-\varepsilon)$ of the constraints, then there exist tables f_w that make the verifier accept with probability at least $1 - 4\varepsilon$.*

Proof. The natural thing to do is to take f_w to be the long code $\mathsf{LONG}_{\ell(w)}$ of the label $\ell(w)$, for each $w \in W$. Note that since the graph is regular, (v, w_1) is a random edge in E. Therefore, with probability $1-\varepsilon$, we have $f_{w_1}(x') = x'_{\ell(w_1)} = x_{\pi_{w_1 \to v}(\ell(w_1))} = x_{\ell(v)}$. By a union bound, with probability at least $1-3\varepsilon$, we also have $f_{w_2}(y') = y_{\ell(v)}$ and $f_{w_3}(z') = z_{\ell(v)}$. It is easy to see now that the conditions (1) and (2) are both satisfied if $\mu_{\ell(v)} = 0$ which happens with probability $(1-\varepsilon)$. Thus the overall probability of acceptance by the verifier is at least $1 - 4\varepsilon$. □

We next turn to the soundness analysis, where our aim is to prove the following, which together with Lemma 4 implies Theorem 2 (since δ, ε can be arbitrarily small in the $(\delta, 1-\varepsilon)$-distinguishing problem for Unique Games that we reduce from).

Lemma 5. *Suppose the UG instance is at most δ-satisfiable. Then the verifier accepts with probability $\frac{1}{2}+\gamma$ where $\gamma \leq O(\sqrt[5]{\delta/\varepsilon})$.*

The remainder of this section is devoted to proving the above lemma.

We begin by making a what will be a hugely convenient notation switch. We will assume that the functions f_w have range $\{1,-1\}$ instead of $\{0,1\}$ by using $(-1)^b$ to represent bit $b \in \{0,1\}$. Note that in this notation, the $\oplus$ operation becomes multiplication. The acceptance probability of the verifier, denote it by ρ, can be expressed exactly as follows:

$$\begin{aligned}\rho &= \frac{1}{2} \mathop{\mathbb{E}}_{\substack{v,w_1,w_1,w_3\\x,y,z}} \left[\frac{1+f_{w_1}(x')f_{w_2}(y')f_{w_3}(z')}{2}\right] + \frac{1}{2} \mathop{\mathbb{E}}_{\substack{v,w_1,w_1,w_3\\x,y,z}} \left[\frac{1-f_{w_1}(x')f_{w_2}(y')f_{w_3}(\overline{z'})}{2}\right] \\ &= \frac{1}{2} + \frac{1}{4} \mathop{\mathbb{E}}_{v} \left[\mathop{\mathbb{E}}_{x,y,z}[g_v(x)g_v(y)g_v(z)] - \mathop{\mathbb{E}}_{x,y,z}[g_v(x)g_v(y)g_v(\overline{z})]\right] \qquad (3)\end{aligned}$$

where $g_v : \{0,1\}^R \to [-1,1]$ is defined as

$$g_v(x) = \mathop{\mathbb{E}}_{w \in N(v)}[f_w(x \circ \pi_{w \to v})] \qquad (4)$$

(the second step (3) above used the independence of w_1, w_2, w_3). Here, g_v is the average of the Boolean functions f_w over the neighbors w of v.

2.3 Detour into Fourier analysis

The simplification of the expression (3) will require us to expand the function g_v in the Fourier basis. The following exercise develops the needed background.

Exercise 2 ("Fourier basics."). *Let $\mathcal{F} = \{f \mid f : \{0,1\}^n \to \mathbb{R}\}$ be the space of real-valued functions on $\{0,1\}^n$. For two functions $f, g \in \mathcal{F}$ define their inner product*

$$\langle f, g \rangle = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)g(x)\,. \qquad (5)$$

For $\alpha \in \{0,1\}^n$, define the function $\chi_\alpha : \{0,1\}^n \to \mathbb{R}$ by $\chi_\alpha(x) = (-1)^{\alpha \cdot x}$ (here $\alpha \cdot x$ denotes the dot product of the vectors α and x over the reals).

a) Prove that the functions $\{\chi_\alpha\}_{\alpha \in \{0,1\}^n}$ form an orthonormal *basis for the space $\mathcal{F}$ with respect to the inner product (5) defined above.*

b) Conclude that every $f \in \mathcal{F}$ has a unique *representation as*

$$f(x) = \sum_{\alpha \in \{0,1\}^n} \widehat{f}(\alpha)\, \chi_\alpha(x) \qquad (6)$$

for real coefficients $\widehat{f}(\alpha)$ given by $\widehat{f}(\alpha) = \langle f, \chi_\alpha \rangle$.

c) Prove that for $f, g \in \mathcal{F}$, the following identity (called Parseval's identity) holds:

$$\langle f, g \rangle = \sum_{\alpha \in \{0,1\}^n} \widehat{f}(\alpha)\, \widehat{g}(\alpha)\,.$$

In particular, $\sum_\alpha \widehat{f}(\alpha)^2 = \mathbb{E}_x[f(x)^2]$.

d) (This part is not really necessary for us, but a good way to test your understanding.) Suppose $f \in \mathcal{F}$ *is invariant under translations by a vector* $h \in \{0,1\}^n$*, i.e.,* $f(x \oplus h) = f(x)$ *for all* x*. Prove that* $\widehat{f}(\alpha) = 0$ *whenever* $\alpha \cdot h = 1$ *modulo* 2*.*

Armed with the above basics, let us simplify (3).

Lemma 6. *For* x, y, z *as chosen by the verifier, for any function* $g : \{0,1\}^R \to [-1,1]$*,*

$$\mathop{\mathbb{E}}_{x,y,z}[g(x)g(y)g(z)] = \sum_{\alpha \in \{0,1\}^R} \widehat{g}(\alpha)^3 (1-2\varepsilon)^{|\alpha|} \tag{7}$$

where $|\alpha| = |\mathrm{supp}(\alpha)|$ *denotes the number of* 1*'s in* α*. Similarly,*

$$\mathop{\mathbb{E}}_{x,y,z}[g(x)g(y)g(\overline{z})] = \sum_{\alpha \in \{0,1\}^R} \widehat{g}(\alpha)^3 (-1)^{|\alpha|} (1-2\varepsilon)^{|\alpha|} \tag{8}$$

Proof. We will only prove (7). The proof of (8) is very similar. Recall that $z = x \oplus y \oplus \mu$, where μ is a random ε-biased vector. Using the Fourier expansion (6), we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{x,y,z}[g(x)g(y)g(z)] &= \mathop{\mathbb{E}}_{x,y,z}\left[\left(\sum_\alpha \widehat{g}(\alpha)\chi_\alpha(x)\right)\left(\sum_\beta \widehat{g}(\beta)\chi_\beta(y)\right)\left(\sum_\xi \widehat{g}(\xi)\chi_\xi(z)\right)\right] \\
&= \sum_{\alpha,\beta,\xi} \widehat{g}(\alpha)\widehat{g}(\beta)\widehat{g}(\xi) \mathop{\mathbb{E}}_{x,y,\mu}[\chi_\alpha(x)\chi_\beta(y)\chi_\xi(x \oplus y \oplus \mu)] \\
&= \sum_{\alpha,\beta,\xi} \widehat{g}(\alpha)\widehat{g}(\beta)\widehat{g}(\xi) \mathop{\mathbb{E}}_{x}\left[\chi_\alpha(x)\chi_\xi(x)\right] \mathop{\mathbb{E}}_{y}\left[\chi_\beta(y)\chi_\xi(y)\right] \mathop{\mathbb{E}}_{\mu}\left[\chi_\xi(\mu)\right] \\
&= \sum_\alpha \widehat{g}(\alpha)^3 \mathop{\mathbb{E}}_{\mu}\left[\chi_\alpha(\mu)\right] \quad \text{using orthonormality of the functions } \chi_\alpha\text{'s} \\
&= \sum_\alpha \widehat{g}(\alpha)^3 \prod_{i=1}^R \mathop{\mathbb{E}}_{\mu_i}[(-1)^{\alpha_i \mu_i}] \\
&= \sum_\alpha \widehat{g}(\alpha)^3 (1-2\varepsilon)^{|\alpha|} \,.
\end{aligned}$$

□

2.4 Continuation of soundness analysis

Plugging Lemma 6 into (3), we get the verifier's acceptance probability equals

$$\rho = \frac{1}{2} + \frac{1}{2} \mathop{\mathbb{E}}_{v}\Bigg[\underbrace{\sum_{\substack{\alpha \in \{0,1\}^R \\ |\alpha| \text{ odd}}} \widehat{g_v}(\alpha)^3 (1-2\varepsilon)^{|\alpha|}}_{\theta_v} \Bigg] . \tag{9}$$

Suppose that $\rho = 1/2 + \gamma$. Our goal is to prove the claim of Lemma 5 that γ is small. By (9), we have $\mathbb{E}_v[\theta_v] = 2\gamma$. By an averaging argument, at least γ fraction of $v \in V$ satisfy $\theta_v \geq \gamma$. Call such vertices ***good.*** The following is an easy consequence of Parseval's identity.

Exercise 3. *If v is good, then there exists α, $1 \le |\alpha| \le \varepsilon^{-1}\log(2/\gamma)$ such that $|\widehat{g_v}(\alpha)| \ge \gamma/2$.*

Decoding labels. We now describe how to decode a labeling ℓ for vertices in $V \cup W$ based on the functions f_w and g_v.

For $v \in V$, if v is not good, pick an arbitrary label for v, say $\ell(v) = 1$. For good vertices v, let α be an arbitrary nonzero string with $|\widehat{g}(\alpha)| \ge \gamma/2$ and $|\alpha| \le \varepsilon^{-1}\log(2/\gamma)$ (as guaranteed by Exercise 3). Set $\ell(v)$ to be an arbitrary element in $\mathrm{supp}(\alpha) = \{i \mid \alpha_i = 1\}$.

For $w \in W$, list decode the set of labels

$$\mathsf{L}_w \triangleq \{b \mid \exists \beta, \mathrm{supp}(\beta) \ni b, |\beta| \le \varepsilon^{-1}\log(2/\gamma), |\widehat{f_w}(\beta)| \ge \gamma/4\}\,.$$

and define $\ell(w)$ to be a random element of L_w if this set is nonempty, and an arbitrary label otherwise.

It is easy to verify (again using Parseval's identity) that L_w is not too large.

Exercise 4. *Prove that L_w as defined above satisfies $|\mathsf{L}_w| \le O(\varepsilon^{-1}\gamma^{-2}\log(1/\gamma))$.*

The following claim shows that the label for good vertices v has a consistent label in the sets L_w for a sizeable fraction of neighbors $w \in N(v)$.

Lemma 7. *Suppose v is good. For at least $\gamma/4$ fraction of $w \in N(v)$,*

$$\ell(v) \in \pi_{w\to v}(\mathsf{L}_w) \triangleq \{\pi_{w\to v}(b) \mid b \in \mathsf{L}_w\}\,.$$

Proof. Let α be such that $\ell(v) \in \mathrm{supp}(\alpha)$, $|\alpha| \le \varepsilon^{-1}\log(2/\gamma)$ and $|\widehat{g_v}(\alpha)| \ge \gamma/2$. (Such an α must exist by our choice of $\ell(v)$ for good vertices v.) Recalling the definition (4) of g_v, it follows that $\widehat{g_v}(\alpha) = \mathbb{E}_{w\in N(v)}\left[\widehat{f_w}(\alpha \circ \pi_{w\to v})\right]$. Hence $\mathbb{E}_w\left[|\widehat{f_w}(\alpha \circ \pi_{w\to v})|\right] \ge |\widehat{g_v}(\alpha)| \ge \gamma/2$.

Since $|\widehat{f_w}(\alpha \circ \pi_{w\to v})| \le 1$, by an averaging argument, for at least a $\gamma/4$ fraction of $w \in N(v)$, we must have $|\widehat{f_w}(\alpha \circ \pi_{w\to v})| \ge \gamma/4$. Also $|\alpha \circ \pi_{w\to v}| = |\alpha| \le \varepsilon^{-1}\log(2/\gamma)$. Therefore, for such w, $\mathrm{supp}(\alpha \circ \pi_{w\to v}) \subseteq \mathsf{L}_w$. Since $\ell(v) \in \mathrm{supp}(\alpha)$, we therefore have $\pi^{-1}_{w\to v}(\ell(v)) \in \mathsf{L}_w$, as desired. □

Combining Exercise 4, Lemma 7, and the fact that a fraction γ of $v \in V$ are good, we conclude that the decoding strategy satisfies at least a fraction

$$\gamma \cdot \frac{\gamma}{4} \cdot \Omega\left(\frac{\varepsilon\gamma^2}{\log(1/\gamma)}\right) \ge \Omega(\varepsilon\gamma^5)$$

of the edges of the UG instance. We know that this must be at most δ since the UG instance is at most δ-satisfiable, so we have $\gamma \le O((\delta/\varepsilon)^{1/5})$. This finishes the proof of Lemma 5.

3 Some Reflections

We now reflect on the role of "dictatorship tests" in reductions of the above form, and also briefly touch upon a connection to *polymorphisms* which are a key part of the algebraic approach to studying the complexity dichotomy of CSPs. We will be not be very precise and keep the discussion at an informal level.

3.1 Converting dictatorship tests to Unique Games hardness

The approach behind the above hardness for 3-Lin consisted of the analysis of a *dictatorship test* of the function $g = g_v$ defined on $\{0,1\}^R$ and taking values which are $\{1,-1\}$ valued random variables. The test involved picking three locations x, y, z and checking a 3-Lin constraint on the ± 1 values of g at these points (sampled independently from the respective random variables).

To get a hardness result for other CSPs, one needs to check a constraint corresponding to the predicate of that CSP (eg. for Max Cut, one should check $g(x) \neq g(y)$). The dictatorship test should have the following two properties:

- COMPLETENESS: If g is a dictator (i.e., $g(x) = (-1)^{x_i}$ for some $i \in [R]$), the test accepts with probability at least c.
- SOUNDNESS: If g is far from a dictator, the test accepts with probability at most $s + \varepsilon$.

Above, far from dictator means that g has no variable with *influence* more than $\tau = \tau(\varepsilon)$.[3] In the case of Boolean functions, the influence of $i \in [R]$ on g, denoted $\mathrm{Inf}_i(g)$, is defined as $\mathrm{Inf}_i(g) = \mathbb{P}_x[g(x) \neq g(x \oplus e_i)]$ where $x \oplus e_i$ denotes x with i'th bit flipped.

Exercise 5. *Let $g : \{0,1\}^R \to \{1,-1\}$. Prove that $\mathrm{Inf}_i(g) = \sum_{\alpha:\alpha_i=1} \widehat{g}(\alpha)^2$. Deduce that the total influence of all variables equals $\sum_{i=1}^R \mathrm{Inf}_i(g) = \sum_\alpha |\alpha| \widehat{g}(\alpha)^2$.*

Remark 3 (Noise sensitivity)**.** The above notion of influence the "sensitivity" of f to the value of the i'th bit. Another notion called "noise sensitivity," which plays an important role in hardness results for Max Cut, measures how sensitive f is to flipping a random subset of coordinates of certain size (say εn). The actual definition uses flipping each coordinate independently with probability ε. Formally, for $0 \leq \varepsilon \leq 1$, define the ε-noise sensitivity of f as

$$\mathrm{NS}_\varepsilon(f) = \mathop{\mathbb{P}}_{x,\mu}[f(x) \neq f(x \oplus \mu)]$$

where x is uniform in $\{0,1\}^R$ and μ is ε-biased (i.e., $\mu_i = 1$ with probability ε, independently for each i). The noise sensitivity can be expressed in terms of f's Fourier coefficients.

Exercise 6. *For every $\varepsilon \in [0,1]$, prove that*

$$\mathrm{NS}_\varepsilon(f) = \frac{1}{2} - \frac{1}{2} \sum_\alpha \widehat{f}(\alpha)^2 (1-2\varepsilon)^{|\alpha|} .$$

Using arguments of a similar flavor to Lemma 5 and in particular Lemma 7 (the list decoding of the tables g_v and f_w will now consist of coordinates with sizeable influence), one can show that the existence of a dictatorship test as above that works for functions whose values are $\{1,-1\}$-valued random variables[4], implies that $(s+\varepsilon, c-\varepsilon)$-distinguishing the associated CSP is Unique Games-hard (for any constant $\varepsilon > 0$). In other words, a dictatorship test automatically implies a matching

[3] Actually, it refers to a somewhat broader class of functions — we only need that all variables have "low-degree influences," but let us ignore this point for now.

[4] For CSPs over domain D, the range of the functions will be D-valued random values, with a suitable generalization of the notion of influence.

UG-hardness result! Thus, if one believes the UGC, the task of showing inapproximability results for CSPs reduces to designing good tests (using predicates associated with that CSP) to distinguish dictators from functions that are far from dictators.

Therefore, for our next inapproximability result (for Max Cut), we will restrict our attention to designing a dictatorship test with as large a completeness to soundness (c vs. s) ratio as possible.

3.2 Dictatorship tests and approximate polymorphisms

Under the algebraic dichotomy conjecture, the polynomial time decidability of a CSP is captured by the existence of "non-trivial" polymorphisms. A polymorphism (for a Boolean CSP(Γ)), is a function $f : \{0,1\}^R \to \{0,1\}$ such that given R satisfying assignments $x^{(1)}, x^{(2)}, \ldots, x^{(R)} \in \{0,1\}^n$ to any instance $\mathcal{I}$ of CSP(Γ) with n variables, the assignment y defined as $y_i = f(x_i^{(1)}, x_i^{(2)}, \cdots, x_i^{(R)})$ for $i = 1, 2, \ldots, n$ is also a satisfying assignment to $\mathcal{I}$. The dictator functions, $f(z) = z_j$, are trivially polymorphisms for every $j \in [R]$; let us say that a polymorphism that is not a dictator is non-trivial. (More precisely, under the algebraic dichotomy conjecture, the existence of a polymorphism with "weak near-unanimity" implies polynomial time decidability of the CSP.) For example, for 3-LIN, the parity function with an odd number of inputs is a non-trivial polymorphism, and for 2SAT, the majority function on an odd number of inputs is a non-trivial polymorphism.

Dictatorship tests have an interesting connection to "approximate" polymorphisms, which combine many approximate satisfying assignments (which satisfy say a c fraction of constraints of a CSP instance) into an assignment satisfying a fraction c' of constraints. Dictator functions achieve $c' = c$, and the soundness of the dictatorship test is equivalent to the fact that approximate polymorphisms which are far from dictators and have no influential coordinates produce assignments that only satisfy $\approx s$ fraction of constraints. Thus a c vs. s gap for approximate polymorphisms translates into a matching UG hardness result. The converse algorithmic result is the centerpiece of Raghavendra's Ph.D. thesis (though it is not stated in this language there) — the existence of such a non-trivial polymorphism (with no influential coordinates) implies that $(s - \varepsilon, c + \varepsilon)$-approximating the CSP is easy. Thus, under the UGC, the approximability of a CSP is precisely explained by the quality of its low-influence approximate polymorphisms.

4 Further topics

(Typeset notes for this part are not available, but I've made available some handwritten sketch.)

The plan is to discuss a dictatorship test for Max Cut, briefly comment on its connections to integrality gaps for the Goemans-Williamson SDP, and state the general theorem converting SDP integrality gaps into dictatorship tests with matching completeness to soundness ratio (which in turn implies a matching UG-hardness result). Time permitting (which is very unlikey), we may discuss some details of this general conversion in the context of Max Cut (some slides for this part are posted).