Hardness Of Approximation

Lecture 9: Bird's-eye view: Towards proving UGC

Instructor: Irit Dinur and Amey Bhangale *Scribe: Amey Bhangale*

Constraint satisfaction problems are one of the most fundamental problems in computer science. So far in this course, we have seen two hardness results of Max-CSPs - the first one is the NP-hardness of Max-3LIN and the second one is the UG-hardness of Max-Cut. The impressive power of the proofs that we saw in the course is that they not only prove hardness results for these specific CSPs but rater show hardness results for a wide range of CSPs. We elaborate on this more in this lecture.

1 Label Cover and Long Code

In the be beginning of this course, we use Long Code encoding with arbitrary projection Label Cover instance to prove the NP-hardness of gap-$(0.99, 1-\varepsilon)$-3LIN, for any small constant $\varepsilon > 0$. Although, we did not prove the optimal gap of $(\frac{1}{2}+\varepsilon, 1-\varepsilon)$ using this approach, this can be achieved. The first optimal NP-hardness result of Max-3LIN in fact used this approach. This was shown by Håstad in his celebrated paper "Some optimal inapproximability results" [Hås01]. In this course, we took a different approach to show the optimal NP-hardness of Max-3LIN.

2 Linear Label Cover and Hadamard code

To show the optimal NP-hardness of Max-3LIN, we started with a Label Cover instance where the projection constraints are *linear/affine* in nature. The advantage of using such a Label Cover instance is that we can replace the Long Code with the Hadamard code. We saw that the Hadamard encoding has a subcode covering property and we crucially used it to get the optimal result for Max-3LIN using the BLR linearity test.

To generalize the result for any linearity test, let's abstract out parameters of linearity tests. Given a Boolean function $f : \{0,1\}^n \to \{0,1\}$ (we only consider balanced Boolean functions), we want to check whether f is a linear function or *far* from any linear function. We associate the following parameters with any linearity test.

The number of queries q. The distribution $\mathcal{D}$ of those q queries. The predicate P that the tester applies to the q queried bits. P can be a family of predicates, but for simplicity we will consider P as just a single predicate $P : \{0,1\}^q \to \{0,1\}$. The completeness c is the minimum probability (over $\mathcal{D}$) that the test accepts any linear function. Finally, the soundness s is the maximum probability (over $\mathcal{D}$) that the test accepts a function that is at most $\frac{1}{2}+\delta$ close to any linear function.

In the Problem Set 2, we have seen many different linearity tests and then use of those to get NP-hardness of the corresponding predicate. One can generalize this to any linearity test and get the following theorem.

Theorem 2.1. *(Linearity test gives NP-hardness) Suppose we have a linearity test with parameters as described above. Then, for every constant $\varepsilon > 0$, given a Max-P-CSP instance $\mathcal{I}$, it is NP-hard to distinguish between the following two cases:*

1. $val(\mathcal{I}) \geq c - \varepsilon$

2. $val(\mathcal{I}) \leq s + \varepsilon$

We already saw the above theorem for a few predicates. The first one is one of the most fundamental predicates - `3LIN` predicate. BLR linearity test uses this predicate and has completeness 1 and soundness $1/2 + \delta$. The second predicate (rather class of predicates) is the one in the graph linearity test that we saw in the Problem Set 2. This predicate is not so natural as `3LIN` but it has good trade-off between the number of queries and the soundness parameter.

The advantage of Theorem 2.1 is that it shows NP-hardness of approximating Max-CSP, but the disadvantage is that it does not give tight NP-hardness of all the predicates. For instance, suppose we want to get the NP-hardness of gap-3SAT$(\frac{7}{8} + \varepsilon, 1 - \varepsilon)$ using Theorem 2.1. It is enough to come up with a 3 query linearity test with completeness 1 and soundness $7/8$. **However, one can show that such a test does not exist!**

3 Unique Games and Long Code

Unique Games and Long Codes are the best buddies! We try to sketch here what we mean by this.

For any dictatorship test, we have similar parameters as in the linearity test described above - $q, \mathcal{D}, P, c, s$. The only difference is the interpretation of the completeness and soundness parameters c and s. Recall the UG-hardness result of Max-Cut. The underlying tool in that proof is a 2-query dictatorship test with $\neq$ predicate. We used the following *low degree influence* as the measure of a function being far from any dictator.

Definition 3.1 (k-degree influence of bit i on f)**.** *Let $f : \{0,1\}^n \to \mathbb{R}$. The k-degree influence of bit i on f is defined as:*

$$Inf_i^k(f) \coloneqq \sum_{i \in S,\, |S| \leq k} \widehat{f}(S)^2$$

f is said to be δ far from any dictator function if $Inf_i^k(f) \leq \delta$ for all $i \in [n]$. Coming back to the Max-Cut UG-hardness, what we essentially showed is the following: If there is a 2-query dictatorship test with completeness c and soundness s which uses $\neq$ as a predicate then it is UG-hard to distinguish between the cases when a given Max-Cut instance has maxcut value at least $c - \varepsilon$ or value at most $s + \varepsilon$, for any constant $\varepsilon > 0$.

It turns out Max-Cut predicate (or $\neq$ predicate) is not so special in the above reduction. In fact, we can get the UG-hardness from *any* long code test. More formally,

Theorem 3.2. *(Long code test gives UG-hardness) Suppose we have a long code/dictatorship test with parameters $q, \mathcal{D}, P, c, s$. Then, for every constant $\varepsilon > 0$, given a Max-P-CSP instance $\mathcal{I}$, it is UG-hard to distinguish between the following two cases:*

1. $val(\mathcal{I}) \geq c - \varepsilon$
2. $val(\mathcal{I}) \leq s + \varepsilon$

Why is this theorem useful compared to Theorem 2.1? The family of long code tests is much *wider* than the family of linearity tests. For instance, 1) we have 2-query long code test but there is no 2-query linearity test. 2) For the $3SAT$ predicate, there is a long code test with completeness 1 and soundness $\frac{7}{8} + \delta$!

That's not all. A beautiful result of Raghavendra [Rag08] shows that Theorem 3.2 gives optimal inapproximability for any Max-P-CSP problem, assuming the Unique Games Conjecture. In other words,

if $\text{gap}(s+\varepsilon, c-\varepsilon)$-Max-$P$-CSP is UG-hard for every constant $\varepsilon > 0$, then there exists a long code test with the predicate P that has completeness c and soundness s.

Thus, the advantage of linear Label Cover + linearity test is that we get unconditional results (based on P$\neq$ NP) but the disadvantage is that we do not get tight results for every predicate.

Subcode Covering: If we use Long code with the Unique Games, where the constraints are permutation constraints, then this composition has a *perfect subcode covering* property. To see this, suppose (u, v) is an edge in an Unique Games instance $G = (U \cup V, [L], E)$ with the permutation constraint $\pi_{u,v}$. For any pair of matching labels (i, j) to u and v respectively (i.e $\pi_{uv}(i) = j$). If we let $f_u : \{0,1\}^L \to \{0,1\}$ and $f_v : \{0,1\}^L \to \{0,1\}$ be the Long Codes of i and j respectively, then there is a one-to-one mapping of evaluations f_u and f_v, given by $f_u(x) = f_v(x \circ \pi_{uv}^{-1})$ (where we define for $x \in \{0,1\}^L$ and permutation π the following: $x \circ \pi \in \{0,1\}^L$, $\forall i : (x \circ \pi)_i = x_{\pi(i)}$). Thus, f_v uniformly covers f_v in case of 'correct' long code labeling.

The advantage of perfect subcode covering property is that if we want to query $f_u(x)$ then this can be simulated by querying $f_v(x \circ \pi_{u,v}^{-1})$. Most Label Cover based reductions have two important components: 1) codeword test (like long code/ hadamard tests), where we want to conclude that most functions f_u, f_v are close to some codeword and 2) consistency test, where we conclude that these codewords are 'matching' codeword with respect to the Label Cover constraints. With the perfect subcode covering property, we can get rid of 2). Thus, morally speaking our task is reduced to just the codeword test! In this way, we can hope to convert a property testing result to a complexity theoretic result. This is the essence of Theorem 2.1 and Theorem 3.2.

3.1 Proof Sketch of Theorem 3.2

Fix $\varepsilon > 0$. We start the reduction with a $\text{gap}(\delta, 1-\delta)$-UG instance $G = (U \cup V, [L], E)$ for $0 < \delta \ll \Omega(\varepsilon^5)$. For each vertex in V we introduce 2^L variables of the Max-P-CSP instance. So the set of variables is $V' := V \times \{0,1\}^L$. We think of these variables as $folded$ i.e (v, x) and $(v, \bar{x})$ represent a same variable, one with positive literal and one negative.

We define the constraints of the P-CSP instance using the following distribution T:

- Select $u \in U$ uniformly at random.
- Select $v_1, v_2, \ldots, v_q \in N(u)$ independently uniformly at random.
- Select $x_1, x_2, \ldots, x_q \in \{0,1\}^L$ according to $\mathcal{D}$.
- Output $((v_1, x_1 \circ \pi_{uv_1}^{-1}), (v_2, x_2 \circ \pi_{uv_2}^{-1}), \ldots, (v_q, x_q \circ \pi_{uv_q}^{-1}))$.

where we define for $x \in \{0,1\}^L$ and permutation π the following:

$$x \circ \pi \in \{0,1\}^L, \ \forall i : (x \circ \pi)_i = x_{\pi(i)}$$

Completeness: Let $l : U \cup V \to [L]$ be a labeling for the UG instance with value at least $1-\delta$. So the probability an edge is not satisfied is at most δ. We define the assignment to the variables by assigning each vertex $v \times x$ in the cloud of a vertex $v \in V$ according to $f_v : \{0,1\}^L \to \{0,1\}$ which is defined as follows: $f_v(x) := x_{l(v)}$.

The value of this assignment is equal to

$$\begin{aligned}
&\mathbf{E}[P(f_{v_1}(x_1 \circ \pi_{uv_1}^{-1}), f_{v_2}(x_2 \circ \pi_{uv_2}^{-1}), \ldots, f_{v_q}(x_q \circ \pi_{uv_q}^{-1}))] \\
&= \mathbf{E}[P((x_1 \circ \pi_{uv_1}^{-1})_{l(v_1)}, (x_2 \circ \pi_{uv_2}^{-1})_{l(v_2)}, \ldots, (x_q \circ \pi_{uv_q}^{-1})_{l(v_q)})] \\
&= \mathbf{E}[P((x_1)_{\pi_{uv_1}^{-1}(l(v_1))}, (x_2)_{\pi_{uv_2}^{-1}(l(v_2))}, \ldots, (x_q)_{\pi_{uv_q}^{-1}(l(v_q))})] \\
&\geq (1 - q\delta) \cdot \mathop{\mathbf{E}}_{\mathcal{D}}[P((x_1)_j, (x_2)_j, \ldots, (x_q)_j)]
\end{aligned}$$

where in the last inequality, we used the fact that all the labels $l(v_i)$ map to the same label j w.r.t the permutation π_{uv_i} for all $i \in [q]$, with probability at least $(1 - q\delta)$. The value $(1 - q\delta)$ is due to the possibility of each edge (u, v_i) being violated by the assignment ℓ. Now, $\mathbf{E}_{\mathcal{D}}[P((x_1)_j, (x_2)_j, \ldots, (x_q)_j)]$ is precisely the completeness of a dictatorship test!

In conclusion, we get $val(\mathcal{I}) \geq (1 - q\delta) \cdot c$.

Soundness: Let $f_v : \{0,1\}^L \to \{+1, -1\}$ be assignments to the variables with $val(\mathcal{I}) \geq s + \varepsilon$. We will use it to get a labeling for the UG instance with value at least $\Omega(\varepsilon^5)$. For $u \in U$, define $f_u : \{0,1\}^L \to \mathbb{R}$ as follows:

$$f_u(x) = \mathop{\mathbf{E}}_{v \in N(u)}[f_v(x \circ \pi_{uv}^{-1})].$$

With this notation, the value of the assignment is

$$s + \varepsilon \leq \mathop{\mathbf{E}}_{u} \mathop{\mathbf{E}}_{x_1, x_2, \ldots, x_q \sim \mathcal{D}}[P(f_u(x_1), f_u(x_2), \ldots, f_u(x_q))].$$

Note that although the domain of P is a set of Boolean strings, the expression $P(f_u(x_1), f_u(x_2), \ldots, f_u(x_q))$ still makes sense. This is because we can think of P as a multilinear polynomial over $\mathbb{R}$ by the virtue of Fourier analysis.

By a simple averaging argument, we can conclude that for at least $\varepsilon/2$ fraction of $u \in U$ the inner expectation is at least $s + \varepsilon/2$ (we call those u *good*). Thus, for any *good* u we can conclude that there exists at least one degree k influential variable, for some constant $k = O(\log(1/\varepsilon))$. We assign a random label from a list of influential labels to the vertex u (we already saw that the list size is upper bounded by $O(k/\varepsilon)$). Why should these labels be non-trivially consistent with each other for *good* u's? The reason why this is true is that if i^{th} variable is influential in f_u then for at least $\Omega(\varepsilon)$ fraction of $v \in N(u)$, the corresponding projected label $\pi_{uv}(i)$ is influential in f_v! This just follows from the definition of f_u.

Thus, for any $v \in V$, we assign a random label from the list of degree-k influential variables in f_v. $\Omega(\varepsilon^2)$ fraction of the edges (u, v) are such that both u is *good* and there is a pair (i, j) in the list of influential coordinates of f_u and f_v respectively such that $\pi_{u,v}(i) = j$. Thus, this randomized labeling satisfies $\Omega(\epsilon^2 \cdot \epsilon^2/k^2)$ fraction of UG constraints in expectation, which is a contradiction as $\delta \ll \Omega(\varepsilon^5)$.

Remark 3.3. *We stated the theorem for long code test of Boolean functions. We can similarly define long code tests for functions $f : [d]^n \to [d]$. This can be used to get UG-hardness of Max-P-CSP problem where P is a d-ary predicate.*

4 Can we prove UGC using linear Label Cover + Linearity testing?

With these tools in hand, suppose we want to prove NP-hardness of approximating Unique Games with gap $(\delta, 1 - \delta)$. We can try the following:

1. Attempt 1: 2-query linearity test (over large alphabet) with permutation predicate. If we have such a test with completeness almost 1 and soundness going to 0, then we can possibly try to use Theorem 2.1 to prove NP-hardness of Unique Games. Alas, such a test does not exist!

2. Attempt 2: Smooth parallel repetition with very few clause vs. variable rounds. When we do the smooth parallel repetition starting gap$(s, 1)$-3SAT, we kept at least $\Omega(k)$ rounds of clauses vs. variable games to get the gap of $(2^{-\Omega(k)}, 1)$. The number of preimages of each label in each projection constraint is at least an exponential in the number of clause *vs.* variable rounds. Thus, the final game is far from being a Unique Game. One way to reduce the number of preimages is by reducing the clause *vs.* variable rounds to say 5 or 10 out if k. In this way, we can reduce the preimages to a constant, independent of the number of rounds. However, in the soundness case, it can be shown that the value of the game is at least some fixed constant and thus, we cannot have a Label Cover with arbitrary low soundness with such an approach.

3. Attempt 3: We saw that the Hadamard encoding has a nice subcode covering property when composed with the smooth Label Cover instance. Long code is used to get the tight results but assuming Unique Games conjecture. Instead of the Hadamard encoding, can we have some other encoding which has the best of both the worlds? i.e an encoding which has subcode covering property and which can be tested using permutation constraints. The answer to this question is yes with one caveat that the constraints are *almost* permutation constraints (2-to-2 instead of 1-to-1, to be precise). This can be achieved with the Grassmann encoding!!!

We will discuss in more detail about the Grassmann encoding in the next lecture.

References

[Hås01] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.

[Rag08] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 245–254. ACM, 2008.