Advanced Approximation Algorithms **(CMU 15-854B, Spring 2008)**

Lecture 25: Hardness of Max-kCSPs and Max-Independent-Set

April 15, 2008

Lecturer: Ryan O'Donnell *Scribe: Eric Blais*

1 Introduction

In this lecture, we obtain hardness of approximation results for two problems: Max-kCSP, and Max-Independent-Set. In the process of proving these hardness results, we will introduce three general tricks for obtaining hardness results.

1.1 Hardness of Max-kCSP

The hardness result we prove for the maximum constraint satisfaction problem is the following.

Theorem 1.1. *For all constant $\epsilon > 0$ and every $k \in \mathbb{N}$, the $(1-\epsilon)$ vs. $\frac{2^{O(\sqrt{k})}}{2^k}$ decision problem for Max-kCSP is NP-hard.*

Recall that an instance of the Max-kCSP problem is a collection of constraints, each of which is defined over k variables. A random assignment to the variables in a constraint satisfies it with probability $1/2^k$, so a random assignment satisfies a $1/2^k$ fraction of the constraints in a Max-kCSP instance in expectation. This shows that the hardness result is close to optimal, since there is a trivial absolute $1/2^k$ approximation algorithm for Max-kCSP.

Theorem 1.1 was originally proved by Samorodnitsky and Trevisan [3]. Håstad and Wigderson [2] later simplified the proof of the theorem. We will sketch the proof of this theorem in Section 4.

1.2 Hardness of Max-Independent-Set

The second result we show is a hardness of approximation result for the Max-Independent-Set problem.

Theorem 1.2. *For all constant $\epsilon > 0$, no factor-$\frac{1}{n^{1-\epsilon}}$ polynomial-time algorithm exists for Max-Independent-Set unless NP $\subseteq$ BPP.*

This is a fairly striking statement, since there is a completely trivial $\frac{1}{n}$-approximation algorithm to the problem: return any vertex of the graph. Theorem 1.2 shows that no other algorithm can expect to do much better than this trivial algorithm.

As an aside, note that this problem is also often formulated as the Max-Clique problem. The two problems are equivalent, since the maximum clique of a graph G is also the maximum independent set in the complement graph $\bar{G}$.

The proof of Theorem 1.2 is obtained by first proving Theorem 1.1, and then applying three reduction "tricks" to convert a hardness result for the Max-kCSP problem to a strong hardness result for the Max-Independent-Set problem. The details of this reduction are outlined in the next sections.

2 Three hardness reduction tricks

We present three hardness reduction tricks in this section: random sparsification, the FGLSS reduction, and serial repetition. The immediate motivation for these tricks is to prove Theorem 1.2. However, these three tricks are also interesting in their own right, and can be useful for proving other hardness results.

2.1 Random sparsification

The random sparsification trick can be presented with an example:

Say ϕ is a 3-bit predicate of the form $\phi(a, b, c) = a \vee (b \oplus c)$. Suppose Max-$\phi$ is shown to be 1 vs. $\frac{3}{4} + \epsilon$ NP-hard,[1] and that the reduction that proves this hardness result outputs instances of Max-ϕ with n variables and $m \sim n^3$ constraints. In other words, the reduction proves that the the Max-ϕ problem restricted to instances with n variables and $m \sim n^3$ constraints is 1 vs. $\frac{3}{4} + \epsilon$ NP-hard.

Suppose that we wanted to show that Max-ϕ instances with n variables and $m = O(n)$ constraints are also hard. Can we use the above result to prove this claim? We claim that we can: simply pick (or retain) $O(n)$ of the n^3 constraints at random. This is the trick we refer to as *random sparsification*. The following two propositions show that the random sparsification trick preserves the hardness of approximation of the original problem.

Proposition 2.1 (Completeness)**.** *Let I be a satisfiable instance of the Max-ϕ problem with n variables and $m \sim n^3$ constraints. Then the instance I' of the Max-ϕ problem with n variables and $m = O(n)$ constraints obtained by taking a random sparsification of I is also satisfiable.*

Proof. Consider any satisfying assignment of I. Since the constraints of I' are a subset of the constraints of I, the same assignment also satisfies I'. □

Proposition 2.2 (Soundness)**.** *Let I be an instance of the Max-ϕ problem with n variables and $m \sim n^3$ constraints with optimal value $\mathsf{Opt}(I) \leq \frac{3}{4} + \epsilon$. Then the instance I' with n variables and $m = O(n)$ constraints obtained by taking a random sparsification of I has optimal value $\mathsf{Opt}(I') \leq \frac{3}{4} + 2\epsilon$ with very high probability.*

[1] This is in fact true, by a proof of Håstad.

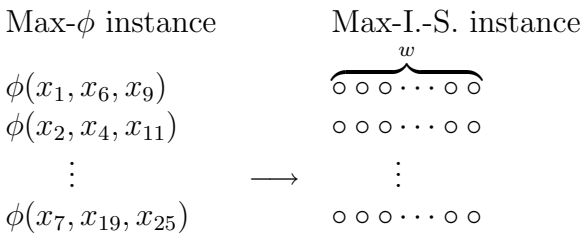
Proof sketch. Fix any assignment. It satisfies at most a $\frac{3}{4} + \epsilon$ fraction of the original constraints in I. So the expected fraction of constraints satisfied in the sparsified instance I' is also at most $\frac{3}{4} + \epsilon$. Chernoff bounds imply that when m is large enough, with very high probability, this assignment will satisfy at most a $\frac{3}{4} + 2\epsilon$ fraction of the retained constraints. In fact, with $m = O(n)$, we can obtain that the probability that more than a $\frac{3}{4} + 2\epsilon$ fraction of the constraints of I' are satisfied is $\ll 2^{-n}$. The soundness proposition then follows by applying the union bound over all 2^n assignments. □

The random sparsification trick is very useful when we want to obtain hardness results with factors that are a function of n, since it enables us to keep the same hardness results but lower the size of the instances. There is however a small price to pay: whereas the c vs. s decision problem was originally NP-hard, the random sparsification step introduces randomness to the reduction, so that in the sparsified problem, the c vs. s decision problem is hard only if we make the stronger assumption that NP $\not\subseteq$ BPP.[2]

2.2 FGLSS reduction

The FGLSS reduction is a generic reduction technique for going from CSP instances to independent set (or clique) instances. The reduction is named after Feige, Goldwasser, Lovász, Safra, and Szegedy, who introduced this technique [1] to analyze the hardness of approximation of the Max-Clique problem. We again describe the trick with an example.

Suppose you have shown that the 1 vs. s decision problem for Max-ϕ is NP-hard. Suppose furthermore that the predicate ϕ has w satisfying assignments. For example, if ϕ is of the form $\phi(a, b, c) = a \vee (b \oplus c)$, it has $w = 6$ possible satisfying assignments. The FGLSS reduction converts instances of the Max-ϕ problem to instances of the Max-Independent-Set problem.



For each constraint in the Max-ϕ instance, we add a row of w vertices in the Max-Independent-Set instance. We associate each vertex in a row with a satisfying partial assignment for the associated constraint. For example, given a constraint $\phi(x_1, x_2, x_3) = x_1 \vee (x_2 \oplus x_3)$, we create 6 vertices, one for each assignment to the variables x_1, x_2, x_3 that satisfies ϕ. (E.g., The first vertex is associated with the partial assignment $x_1 = 0, x_2 = 0, x_3 = 1$, the second with $x_1 = 0, x_2 = 1, x_3 = 0$, the third with $x_1 = 1, x_2 = 0, x_3 = 0$, and so on.)

[2] We can also use walks on expander graphs to derandomize the sparsification procedure. This optimization enables us to obtain the same hardness result with the slightly weaker assumption that NP $\not\subseteq$ ZPP.

We have defined the vertices in the Max-Independent-Set instance; now we define what edges are present in this instance. Put an edge between any 2 inconsistent partial assignments. Note that with this rule, each row of vertices in the Max-Independent-Set instance becomes a clique.

By our construction, the largest independent set in any instances created by the FGLSS reduction has fractional size at most $\frac{1}{w}$. The following propositions show that the hardness of the 1 vs. s decision problem for Max-ϕ translates to a factor-s hardness result for the Max-Independent-Set problem.

Proposition 2.3 (Completeness)**.** *If a Max-ϕ instance is satisfiable, the corresponding instance of Max-Independent-Set has optimal value $\frac{1}{w}$.*

Proof. Pick a satisfying assignment. Choose all vertices that are consistent with the assignment. Since no two of these vertices can be inconsistent with each other, no edges connect any two of the vertices we chose. And since the assignment was satisfying, there is one vertex picked per row. □

Proposition 2.4 (Soundness)**.** *If the optimal value of a Max-ϕ instance is less than s, then the optimal value of the corresponding Max-Independent-Set instance is less than $\frac{s}{w}$.*

Proof. We prove the contrapositive statement: given an independent set of size at least $\frac{s}{w}$, we can get an assignment satisfying at least s constraints in the original Max-ϕ instance. This is easy: if our set is independent, it means that all the vertices in the set are consistent with some assignment, and there can be at most 1 vertex per row, so the assignment satisfies s constraints. □

To recap, the FGLSS reduction gives us factor-s hardness for the Max-Indepedent-Set problem, and the size of the construction is mw, where m is the number of constraints in the original instances of the Max-ϕ problem.

The gap is "moved down" in the reduction: in the original Max-ϕ instance, the hardness result is 1 vs. s, whereas after the reduction, the hardness of the Max-Independent-Set problem is $\frac{1}{w}$ vs. $\frac{s}{w}$. Note that we can't really hope to get a reduction without this gap movement.

2.3 Serial repetition

The third and final trick that we examine in this section is serial repetition.[3] We again illustrate the trick through an example.

Suppose we proved 1 vs. s hardness for the Max-ϕ problem, where ϕ is a type of constraint with k literals (i.e., Max-ϕ is a restriction of the Max-kCSP problem). Furthermore, suppose that the reduction that proved the hardness results outputs m constraints.

For the serial repetition, fix $t \in \mathbb{N}$. Taking an instance I of the Max-ϕ problem, we create a new instance I' by taking all m^t lists of t constraints in I, and converting each list to an AND

[3]The name is chosen to distinguish this method of repetition from parallel repetition.

of its constraints. With this reduction, when I is a kCSP instance, the resulting problem instance I' is a kt-CSP. Also, if the constraints ϕ in I each had w satisfying assignments, the constraints $\phi \wedge \cdots \wedge \phi$ in I' each have w^t satisfying assignments.

Serial repetition is used to amplify the size of the approximation gap.

Proposition 2.5. *Let I be a satisfiable instance of the Max-ϕ problem. Then the instance I' obtained by taking a serial repetition of I is also satisfiable.*

Proposition 2.6. *Let I be an instance of the Max-ϕ problem with optimal value $\mathsf{Opt}(I) \leq s$. Let I' be an instance of the Max-ktCSP problem obtained by taking the serial repetition of I with parameter t. Then the optimal value of I' is $\mathsf{Opt}(I') \leq s^t$.*

We leave the proof of both propositions as exercises.

Serial repetition is a good reduction trick to use if the goal is to increase the size of the of the gap and we are not concerned about the size of the instances or the nature of the constraints. Note that we can only do this trick a constant number of times (or maybe a logarithmic number of times) before the size of the problem instances becomes exponential.

One implication of the serial repetition method is that if we have 1 vs. $\frac{1}{2}$ hardness for the Max-kCSP problem, then we also have 1 vs. η hardness for the Max-CSP problem, for any $\eta > 0$. (See Homework 1.)

Reducing the size of instances. While the serial repetition trick necessarily increases the size of the problem instances, we would like to limit the size of the resulting instances as much as possible. And indeed, when we crunch the probabilities (using Chernoff bounds, union bounds, etc.), we see that we can obtain a gap increase while only retaining $\frac{1}{s^t}$ of the new constraints.

The optimized version of serial repetition generates problems with the following characteristics:

Completeness:	1
Soundness:	ms^t
Size of instances:	$\frac{1}{s^t}$ constraints on n variables

One fine point that we need to mention is that the above characteristics only hold for the optimized serial repetition reduction when $t \gg \frac{\log m}{\log 1/s}$. For our purposes, we will take $t = O(1) \cdot \log m$.

3 Proofs of Theorems 1.1 and 1.2

The proof of Theorem 1.2 uses all three reduction tricks covered in the previous section. Specifically, we start with a hardness result for Max-kCSP. We then use serial repetition to increase the size of the gap, and do random sparsification to reduce the size of the instances.

Finally, we apply the FGLSS reduction to convert the resulting constraint satisfaction problem to a Max-Independent-Set problem.

At the end of the reduction, we obtain an instance of the Max-Independent-Set problem with the following two properties:

1. The size of the graph is $N := O(1) \cdot \frac{1}{s^t} w^t$.
2. The problem has factor-(ms^t) hardness.

When $w^t \ll \frac{1}{s^t}$ (i.e., when $\log w = o(\log 1/s)$), then the hardness factor for Max-Independent-Set is $\frac{1}{N^{1-o(1)}}$ when we take $t = c \log m$ for some large constant c. So to complete the proof of Theorem 1.2, we want to obtain a good hardness result for Max-kCSP. Specifically, we want to reach the following goal:

Goal: A predicate ϕ, such that we have 1 vs. s hardness for Max-ϕ, and such that the number w of satisfying assignments for ϕ satisfies $\log w \ll \log \frac{1}{s}$.

Note that in our goal, there is a trade-off between the soundness of Max-ϕ and the size of the constraints. For example, Max-LIN is not a good candidate to reach the goal since for this problem, $\log w \not\ll \log 1/s$.

The following theorem of Samorodnitsky and Trevisan [3] shows that our goal can be achieved.

Theorem 3.1 (Samorodnitsky, Trevisan '00)**.** *For any constant k, there exists a predicate ϕ on $q := O(k^2)$ bits with $w = 2^k$ satisfying assignments for which we have $1 - \epsilon$ vs. $\frac{2^k}{2^q} + \epsilon$ NP-hardness for all $\epsilon > 0$.*

Theorem 3.1 immediately implies Theorem 1.1, since the Max-ϕ problem with predicate ϕ defined in Theorem 3.1 is a Max-qCSP, and is is $1-\epsilon$ vs. $\frac{2^{O(\sqrt{q})}}{2^q}$ hard. In fact, this theorem is slightly stronger than we need, since Theorem 1.1 only requires us to show 1 vs. $\frac{2^{O(\sqrt{q})}}{2^q}$ hardness.

Theorem 3.1 also satisfies the requirements for our goal, since with the predicate ϕ from this theorem, $\log w = k$ and $\log 1/s = O(k^2)$, so in fact we can get an arbitrarily large gap between the two values. As a result, Theorem 1.2 follows as a corollary of the theorem of Samorodnitsky and Trevisan.

4 Proof sketch for Theorem 3.1

In this section, we sketch the proof of Theorem 3.1, which completes the proof of hardness for the Max-kCSP and Max-Indepedent-Set problems.

The proof of Theorem 3.1 is very much like Håstad's proof of 1 vs. $\frac{1}{2}$ hardness for Max-3LIN. Somehow, we want to iterate the technique used in the hardness proof for Max-3LIN, and we want this iteration to be a non-serial repetition of the technique.

Recall that the original 3LIN hardness proof is based on a "Dictator test". The simplest version of the dictator test can be described as follows:

Naïve dictator test. Given a function, pick $\boldsymbol{x}, \boldsymbol{y}$ uniformly and independently at random. Test whether $f(\boldsymbol{x})f(\boldsymbol{y})f(\boldsymbol{x} \circ \boldsymbol{y}) = 1$.

As we saw in Lecture 22, the probability that f passes the naïve dictator test is

$$\mathbf{E}\left[\frac{1}{2} + \frac{1}{2}f(\boldsymbol{x})f(\boldsymbol{y})f(\boldsymbol{x} \circ \boldsymbol{y})\right] = \cdots \leq \frac{1}{2} + \frac{1}{2}\max_S |\hat{f}(S)|.$$

In Lecture 23, we saw how we can apply two tricks to convert the naïve dictator test to a correct dictator test. First, we test against a random sign b to get rid of constant functions, and second, we add some random noise λ to get rid of large parity functions. In this section, we present a naïve parity test. By applying the same two tricks, we can convert this test to one that enables us to prove Theorem 3.1.

The main idea of the parity test that we now introduce is to copy the naïve dictator test, except that instead of picking two strings to test, we will pick k strings at random and test that every pair of strings in this set satisfies the linearity constraint.

Naïve parity test: Pick $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ uniformly and independently at random. Test that for every pair $\{i, j\} \in \binom{[k]}{2}$, the constraint $f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)f(\boldsymbol{x}_i \circ \boldsymbol{x}_j) = 1$ holds.

The usefulness of the parity test is established by the following theorem.

Theorem 4.1. *Let $f : \{-1,1\}^n \to \{-1,1\}$ satisfy $\max_S |\hat{f}(S)| = \epsilon$. Suppose we choose k strings $\boldsymbol{x}^1, \ldots \boldsymbol{x}^k$ independently and uniformly at random and test that $f(\boldsymbol{x}^i)f(\boldsymbol{x}^j)f(\boldsymbol{x}^i \circ \boldsymbol{x}^j) = 1$ for all $1 \leq i < j \leq k$. Then*

$$\mathbf{Pr}[\textit{test passes}] \leq 2^{-\binom{k}{2}} + \epsilon.$$

Proof. We have

$$\begin{aligned}
\mathbf{Pr}[\text{test passes}] &= \mathbf{E}\left[\prod_{\{i,j\} \in \binom{[k]}{2}} \left(\tfrac{1}{2} + \tfrac{1}{2}f(\boldsymbol{x}^i)f(\boldsymbol{x}^j)f(\boldsymbol{x}^i \circ \boldsymbol{x}^j)\right)\right] \\
&= 2^{-\binom{k}{2}} \sum_{\mathcal{E} \subseteq \binom{[k]}{2}} \mathbf{E}\left[\prod_{\{i,j\} \in \mathcal{E}} f(\boldsymbol{x}^i)f(\boldsymbol{x}^j)f(\boldsymbol{x}^i \circ \boldsymbol{x}^j)\right] \\
&\leq 2^{-\binom{k}{2}} + \operatorname*{avg}_{\emptyset \neq \mathcal{E} \subseteq \binom{[k]}{2}} \mathbf{E}\left[\prod_{\{i,j\} \in \mathcal{E}} f(\boldsymbol{x}^i)f(\boldsymbol{x}^j)f(\boldsymbol{x}^i \circ \boldsymbol{x}^j)\right].
\end{aligned}$$

Thus it suffices to show that

$$\mathbf{E}\left[\prod_{\{i,j\} \in \mathcal{E}} f(\boldsymbol{x}^i)f(\boldsymbol{x}^j)f(\boldsymbol{x}^i \circ \boldsymbol{x}^j)\right] \tag{1}$$

is at most ϵ, for any nonempty $\mathcal{E} \subseteq \binom{[k]}{2}$.

Assume without loss of generality that $\{1,2\} \in \mathcal{E}$. It is always possible to find fixings $\boldsymbol{x}^3 = x^3, \ldots, \boldsymbol{x}^k = x^k$ such that the expectation in (1) does not decrease. We claim that regardless of these fixings, we can upper-bound (1) by ϵ. To see this, consider the quantity inside the expectation once we fix $x^3, \ldots, x^k$. We still have $f(\boldsymbol{x}^1)$, $f(\boldsymbol{x}^2)$, and $f(\boldsymbol{x}^1 \circ \boldsymbol{x}^2)$ inside the product. The remaining terms $f(x^i)$ in the product become ± 1 constants, and the remaining terms $f(\boldsymbol{x}^i \circ \boldsymbol{x}^j)$ (where one of i, j is in $\{1,2\}$ and the other is not) become certain other functions just of $f(\boldsymbol{x}^1)$ or $f(\boldsymbol{x}^2)$. If we collect together (by multiplying) all the functions of $\boldsymbol{x}^1$, and do the same for $\boldsymbol{x}^2$, we see that (1) becomes

$$\pm \mathop{\mathbf{E}}_{\boldsymbol{x}^1, \boldsymbol{x}^2} \left[g(\boldsymbol{x}^1) h(\boldsymbol{x}^2) f(\boldsymbol{x}^1 \circ \boldsymbol{x}^2) \right].$$

Now straightforward Fourier analysis gives that the above is

$$\begin{aligned} \pm \sum_{S \subseteq [n]} \hat{g}(S) \hat{h}(S) \hat{f}(S) &\leq \max_S |\hat{f}(S)| \sum_S |\hat{g}(S)||\hat{h}(S)| \\ &\leq \epsilon \sqrt{\sum_S \hat{g}(S)^2} \sqrt{\sum_S \hat{h}(S)^2} \\ &= \epsilon, \end{aligned}$$

where we used Cauchy-Schwarz. □

References

[1] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. of the ACM*, 43:268–292, 1996.

[2] Johan Håstad and Avi Wigderson. Simple analysis of graph tests for linearity and PCP. *Random Structures and Algorithms*, 22:139–160, 2008.

[3] Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proc. of 32nd STOC*, pages 191–199, 2000.