COS 598I Lecture 4: Refuting constraint satisfaction problems

Lecturer: Pravesh Kothari Scribe: Ştefan Tudose

Spring 2025

In this lecture we look at the δ-refutation problem for random k-XOR with k even. We give a polynomial time algorithm in the regime where the number of clauses m satisfies $m \gtrsim n^{k/2} \log n$, where n is the number of variables. We then give a refutation of random k-XOR (with k even) in the case when the number of clauses m is between n and $n^{k/2}$ which runs in subexponential time. To do so, we introduce the Kikuchi matrices and see how they transform a hypergraph problem into a graph problem. We briefly discuss at the end the connection between random 3-SAT and random k-XOR.

Contents

1 Introduction

The 3-SAT problem is a collection of m clauses (or constraints) on n Boolean variables. Each clause is a disjunction of three variables (possibly negated), such as $x_i \vee \neg x_j \vee x_k$. The goal is to find $x_1, \ldots, x_n$ which satisfy all the constraints. This is known to be an NP-complete problem.

In the random 3-SAT, we still have a collection of m clauses (each of which is a disjunction of three variables) on n variables, but the m clauses are chosen i.i.d. from the set of all $2\binom{n}{3}$ possible clauses. We are interested in the random 3-SAT because it provides an "average case" instance of 3-SAT which is easier to analyze.

The bigger the number the clauses m is, the more likely it is that the formula (the set of clauses) will be unsatisfiable. To be more precise:

Lemma 1. For *random* 3-SAT, if $m > cn/\varepsilon^2$, then for any assignment $x = (x_1, \ldots, x_n)$, x satisfies a fraction of at most $\frac{7}{8} + \varepsilon$ of the constraints with high probability[1].

Proof. If $C_i(x)$ is 1 if x satisfies clause i and 0 otherwise, it is easy to see that $C_1(x), \ldots, C_m(x)$ are i.i.d. taking the value 1 with probability 7/8 and the value 0 with probability 1/8. By Hoeffding's inequality

$$\mathbb{P}\left(\left(C_1(x) - \frac{7}{8}\right) + \ldots + \left(C_m(x) - \frac{7}{8}\right) > \varepsilon m\right) \leq e^{-2m\varepsilon^2} \leq e^{-cn}$$

$$\text{so } \mathbb{P}\left(\text{number of clauses satisfied by } x \geq \left(\frac{7}{8} + \varepsilon\right) m\right) = \mathbb{P}\left(C_1(x) + \ldots + C_m(x) \geq \left(\frac{7}{8} + \varepsilon\right) m\right) \leq e^{-cn}.$$

□

Given an unsatisfiable formula (collection of clauses), our goal is to find a short certificate of unsatisfiability. The word short is not very precise, but it's just to draw attention to the fact that the brute-force method of listing all 2^n possible values $x = (x_1, \ldots, x_n)$ and plugging them in the formula to check is already an upper bound on the length of the certificate, but ideally we would like to do better.

Let us now introduce the *δ-refutation problem*. Fix m, n in the *random* 3-SAT problem. We want to find an algorithm which does the following:

Input: A 3-SAT formula ψ on n variables and with m clauses.

Output: A value $v \in [0, 1]$.

Correctness: v should upper bound the maximum fraction of clauses satisfied by any of the 2^n possible assignments $x = (x_1, \ldots, x_n)$ when plugged into ψ.

Utility: v satisfies $v < 1 - \delta$ with probability at least 0.99 (over the draws of m clauses). Note that δ depends on m and n.

Note that the first three "conditions" (input, output and correctness) are deterministic and have nothing to do with the randomness of the problem. The randomness comes in only through the last condition, utility. The utility condition is necessary in order to say something meaningful, otherwise the algorithm can always output $v = 1$.

In other words, we want an algorithm which takes as input a 3-SAT formula and outputs either $1 - \delta$ or outputs *don't know*. If it outputs $1 - \delta$, then it must be true that the formula which it took as input has a fraction of at most $1 - \delta$ satisfiable clauses. A trivial algorithm would be to always output *don't know* (which is equivalent to always saying $v = 1$). But that's what the utility condition is for: to make sure that the algorithm says something notrivial and that in at least 99% of the cases it guarantees that we have at most $1 - \delta$ satisfiable clauses. Clearly δ must depend on m, n, but we won't have that $\delta > 0$ for any m, n. If m is too small then (almost) all formulas will be satisfiable. We clearly must be in a regime where most formulas are not satisfiable, which for example is the case if $m \gg n$ by Lemma 1. But even in this regime, it seems intuitively clear that the larger m is, the more likely it should be for a formula to not be satisfiable, so the complexity/runtime of the algorithm should be lower.

[1] Here probability refers to the random draws of m clauses.

The best results available for 3-SAT are as follows. For $m \lesssim n^2$ we only have exponential time algorithms. If[2] $m \gtrsim n^3$, then there exists a polynomial time approximation scheme by [AKK95]. However, in between, if $m \gtrsim n^{2+\delta}$, there exist a *subexponential* approximation scheme with runtime $2^{O(n^{1-\delta})}$ by [FLP16] (for $0 < \delta < 1$). The exponential time hypothesis would imply that these results are sharp.

The situation for *random* 3-SAT is much better. For $m \lesssim n$ we only have exponential time algorithms for the refutation problem. If $m \gtrsim n^{3/2^+}$, then there exist polynomial time refutation algorithms by [GK01] and [FG01]. In between, if $m \gtrsim n^{1+\delta}$, there exists a refutation algorithm with runtime $2^{O(n^{1-2\delta})}$ by [RRS17] (for $0 < \delta < 1/2$).

Instead of working with random 3-SAT, we will be working with random k-XOR for convenience of notation. There is a way to recover bounds for random 3-SAT which we will see in the last section.

2 k-XOR

We are going to look at the δ-refutation problem for the k-XOR. The k-XOR problem can be formulated as follows: we are given m clauses in n variables $x_1, \dots, x_n \in \{-1, 1\}$. Each clause is a k-tuple $(a_1, \dots, a_k)$ and a number $b_{a_1,a_2,\dots,a_k} \in \{-1, 1\}$. We say that an assignment $x = (x_1, \dots, x_n)$ satifies all the constraints if

$$x_{a_1} \cdot x_{a_2} \cdot \ldots \cdot x_{a_k} = b_{a_1,a_2,\dots,a_k}$$

for all m clauses.

We consider *random* k-XOR, where the m clauses are chosen i.i.d. uniformly from the set of $2\binom{n}{k}$ possible clauses. It's easy to see as before that if $m > cn/\varepsilon^2$, then any assignment $x = (x_1, \dots, x_n)$ satisfies a fraction of at most $\frac{1}{2} + \varepsilon$ of the constraints with high probability. The δ-refutation algorithm is defined as for the 3-SAT.

For $k = 2$ and $k = 4$ (and in fact any even k) we are going to show the following

Theorem. If $m \gtrsim n^{k/2} \log n$, then there exists $\delta = \delta(m, n) > 0$ so that the random k-XOR has a δ-refutation algorithm which runs in polynomial time.

We are going to show in fact something even stronger: we don't need to take fully random k-XOR, we can have the m k-tuples $(a_1, \dots, a_k)$ fixed and then only the values $b_{a_1,\dots,a_k}$ are chosen randomly. This is what random k-XOR will mean for us below.

We are then going to prove an interpolation result as well:

Theorem. For *random* 4-XOR, if $m \gtrsim n^{1+\varepsilon} \log n$, then we can construct a $\delta(m, n)$-refutation algorithm for *random* 4-XOR with runtime $2^{n^{1-\varepsilon}}$.

The result again works for any even k, but for ease of notation we prove it for $k = 4$.

[2] Note that there are at most $2\binom{n}{3}$ clauses possible, so $m \gtrsim n^3$ is equivalent to $m \simeq n^3$.

2.1 The *random* 2-XOR refutation algorithm

In this section we show that if $m \gtrsim n \log n$, then random 2-XOR has a polynomial time δ-refutation algorithm.

If we let H be the set of clauses, we label the clauses as $x_C = b_C$ where $x_C = \prod_{i \in C} x_i$. Note that we can view this as a graph, where there is an edge between vertices i, j if $(i, j) \in H$ (i.e. if there is a constraint $x_i x_j = b_{ij}$).

Consider the polynomial

$$\Psi(x) = \frac{1}{m} \sum_{c \in H} b_C x_C$$

We see that if x is an assignment satisfying all the constraint, then $\Psi(x) = 1$. Moreover, if $\Psi(x)$ does not satisfy exactly l out of the m constraints, then $\Psi(x) = 1 - 2l/m$. Therefore, if $\Psi(x) < 1 - 2\delta$, then x satisfies at most a fraction of $1 - \delta$ of the constraints. This will be the basis for our refutation algorithm.

2.1.1 A first attempt

Note that

$$\Psi(x) = \frac{1}{m} \sum_{(i,j) \in H} b_{ij} x_i x_j = x^T \left(\frac{1}{m} \sum_{(i,j) \in H} b_{ij} A_{ij} \right) x$$

where $A_{ij} = e_i e_j^T + e_j e_i^T$ is the matrix which has 1 in entries (i, j) and (j, i) and 0 everywhere else (note that $i \neq j$).

We can now bound

$$\Psi(x) \leq \|x\|^2 \left\| \frac{1}{m} \sum_{(i,j) \in H} b_{ij} A_{ij} \right\| = n \left\| \frac{1}{m} \sum_{i,j} b_{ij} 1_{(i,j) \in H} A_{ij} \right\|$$

Our algorithm will take as input a set ψ of m clauses (x_C, b_C) and output

$$v(\psi) = \min \left(1, n \left\| \frac{1}{m} \sum_{(i,j) \in H} b_{ij} A_{ij} \right\| \right)$$

Note that this is a polynomial time algorithm due to the following

Theorem. Given a $n \times n$ matrix, we can compute all eigenvalues of the matrix in polynomial time[3] up to δ additive error.

Correctness is satisfied by the remark above. The only thing we need to check is utility.

By the matrix Khintchine inequality,

$$\mathbb{E} \left\| \sum_{(i,j) \in H} b_{ij} A_{ij} \right\| \leq c \sqrt{\log n} \left\| \sum_{(i,j) \in H} A_{ij}^2 \right\|^{1/2} = c \sqrt{\log n} \sqrt{d_{\max}}$$

[3]Here polynomial time means polynomial in the dimension n, the complexity of the entries of the matrix and $\log 1/\delta$.

where $d_{\max} = \max_{1\le i\le n} \deg i = \max_{1\le i\le n} \#\{j|\ (i,j)\in H\}$. In fact, the inequality

$$\left\| \sum_{(i,j)\in H} b_{ij}A_{ij} \right\| \le c\sqrt{\log n} \left\| \sum_{(i,j)\in H} A_{ij}^2 \right\|^{1/2} \le c\sqrt{\log n}\sqrt{d_{\max}}$$

holds with high probability.[4]

We therefore get the bound

$$v(\psi) \le cn\sqrt{\log n}\sqrt{d_{\max}}/m$$

Let's *assume* that the graph is (almost) regular. Then $d_{\max} \simeq d_{\text{avg}} = 2m/n$. Therefore

$$v(\psi) \le c\sqrt{\frac{n\log n}{m}}$$

Therefore, if $m \gtrsim n\log n$, $v(\psi) < 1-\delta$ with high probability.

The only thing that's left is to get rid of the assumption that the graph is (almost) regular. We will do this in the next section.

2.1.2 Getting rid of the regularity assumption

Lemma 2. If the matrix Γ is diagonal with $\Gamma_{ii} = \deg i + d_{\text{avg}}$ where $d_{\text{avg}} = 2m/n$, then we have

$$\left\| \Gamma^{-1/2} \left(\sum_{(i,j)\in H} b_{ij}A_{ij} \right) \Gamma^{-1/2} \right\| \le c\sqrt{\frac{\log n}{d_{\text{avg}}}}$$

with high probabiliy.

Proof. We write

$$\left\| \Gamma^{-1/2} \left(\sum_{(i,j)\in H} b_{ij}A_{ij} \right) \Gamma^{-1/2} \right\| = \left\| \sum_{(i,j)\in H} b_{ij} \left(\Gamma^{-1/2}A_{ij}\Gamma^{-1/2} \right) \right\|$$

We would like to apply the matrix Khintchine inequaliy and Talagrand's concentration inequality.

Note that $\Gamma^{-1/2}A_{ij}\Gamma^{-1/2}$ has the value

$$\frac{1}{\sqrt{(\deg i + d_{\text{avg}})\cdot(\deg j + d_{\text{avg}})}}$$

[4] If $X = \sum_{(i,j)\in H} b_{ij}A_{ij}$, then recall from the second lecture that we have a way of going from Rademacher to Gaussian coefficients and then apply the matrix Khintchine inequality:

$$\mathbb{E}\,\|X\| \le \sqrt{\frac{\pi}{2}}\mathbb{E} \left\| \sum_{(i,j)\in H} g_{ij}A_{ij} \right\| \le c\sigma(X)\sqrt{\log n}$$

From the third lecture we know that Talagrand's concentration inequality gives that

$$\mathbb{P}(\|X\| - \mathbb{E}\,\|X\| \ge t) \le e^{-t^2/(2\sigma(X)^2)}$$

so $\mathbb{P}(\|X\| \ge C\sigma(X)\sqrt{\log n}) \le \mathbb{P}(\|X\| \ge \mathbb{E}\,\|X\| + c\sigma(X)\sqrt{\log n}) \le e^{-cn}$.

in entries (i,j) and (j,i) and 0 otherwise. Therefore $\left(\Gamma^{-1/2}A_{ij}\Gamma^{-1/2}\right)^2$ has the value

$$\frac{1}{(\deg i + d_{\text{avg}}) \cdot (\deg j + d_{\text{avg}})}$$

in entries (i,i) and (j,j) and 0 otherwise. Therefore,

$$\sum_{(i,j)\in H} \left(\Gamma^{-1/2}A_{ij}\Gamma^{-1/2}\right)^2$$

is diagonal and on the ith diagonal entry it has value

$$\sum_{(i,j)\in H} \frac{1}{(\deg i + d_{\text{avg}}) \cdot (\deg j + d_{\text{avg}})}$$

so the norm of $\sum_{(i,j)\in H} \left(\Gamma^{-1/2}A_{ij}\Gamma^{-1/2}\right)^2$ is the maximum of the above expressions. Note that

$$\begin{aligned}
\sum_{(i,j)\in H} \frac{1}{(\deg i + d_{\text{avg}}) \cdot (\deg j + d_{\text{avg}})} &= \frac{1}{\deg i + d_{\text{avg}}} \cdot \sum_{(i,j)\in H} \frac{1}{\deg j + d_{\text{avg}}} \\
&\leq \frac{1}{\deg i + d_{\text{avg}}} \cdot \sum_{(i,j)\in H} \frac{1}{d_{\text{avg}}} \\
&= \frac{\deg i}{\deg i + d_{\text{avg}}} \cdot \frac{1}{d_{\text{avg}}} \leq \frac{1}{d_{\text{avg}}}
\end{aligned}$$

and therefore

$$\left\| \sum_{(i,j)\in H} \left(\Gamma^{-1/2}A_{ij}\Gamma^{-1/2}\right)^2 \right\| \leq \frac{1}{d_{\text{avg}}}$$

We can now apply matrix Khintchine and Talagrand's concentration inequality to conclude. □

If $\left\|\Gamma^{-1/2}A\Gamma^{-1/2}\right\| \leq K$ then $x^T\left(\Gamma^{-1/2}A\Gamma^{-1/2}\right)x \leq Kx^Tx$ for any $x \in \mathbb{R}^n$, so $x^TAx \leq K \cdot x^T\Gamma x$ for any $x \in \mathbb{R}^n$. In particular, if $x \in \{-1,1\}^n$, we obtain from this observation and the previous lemma that

$$\Psi(x) = x^T\left(\frac{1}{m}\sum_{(i,j)\in H} b_{ij}A_{ij}\right)x \leq \left\|\Gamma^{-1/2}\left(\sum_{(i,j)\in H} b_{ij}A_{ij}\right)\Gamma^{-1/2}\right\| c(x^T\Gamma x)/m \leq c\sqrt{\frac{\log n}{d_{\text{avg}}}}\left(x^T\Gamma x\right)/m$$

with high probability. But note that if $x \in \{-1,1\}^n$ then $x^T\Gamma x = \operatorname{tr}\Gamma = \sum_{i=1}^{n}(\deg i + d_{\text{avg}}) = 4m$, so

$$\Psi(x) \leq c\sqrt{\frac{\log n}{d_{\text{avg}}}} = c\sqrt{\frac{n\log n}{m}}$$

with high probability.

If $m \gtrsim n\log n$, we can take

$$v(\psi) = \min\left(1, \left\|\sum_{(i,j)\in H} b_{ij}\left(\Gamma^{-1/2}A_{ij}\Gamma^{-1/2}\right)\right\|\right)$$

and this will give a $\delta_{m,n}$-refutation algorithm which runs in polynomial time.

2.2 The *random* 4-XOR refutation algorithm

For random 4-XOR we can do something similar. If we let $H \subset \binom{[n]}{4}$ be the set of clauses (which are chosen uniformly at random i.i.d.), then we define

$$\Psi(x) = \frac{1}{m} \sum_{c \in H} x_C b_C$$

Again, if exactly l assignments are not satisfied, then $\psi(x) = 1 - 2l/m$, so if $\psi(x) < 1 - 2\delta$ then a fraction of at most $1 - \delta$ assignments is satisfied.

We can write

$$\Psi(x) = \frac{1}{m} \sum_{\{i,j,k,l\} \in H} x_i x_j x_k x_l b_{ijkl} = \left(x^{\otimes 2}\right)^T \left(\sum_{\{i,j,k,l\} \in H} b_{ijkl} A_{ij,kl} \right) x^{\otimes 2}$$

where $A_{ij,kl}$ is a $n^2 \times n^2$ matrix which has 1 on entries (ij, kl) and (kl, ij) and 0 in all the other. One can then proceed as in the case of 2-XOR. We have proven

Theorem 3. For random 4-XOR, if $m \gtrsim n^2 \log n$, then there exissts a polynomial time $\delta_{m,n}$-refutation algorithm.

Clearly this strategy works for any even dimension, so we have that for random k-XOR with k even, if $m \gtrsim n^{k/2} \log n$, then there exists a polynomial time refutation algorithm.

We are now going to prove the interpolation result

Theorem 4. For *random* 4-XOR, if $m \gtrsim n^{1+\varepsilon} \log n$, then we can construct a $\delta(m, n)$-refutation algorithm for *random* 4-XOR with runtime $2^{n^{1-\varepsilon}}$.

Recall that $H \subset \binom{[n]}{4}$ is the set of clauses (which are chosen uniformly at random i.i.d.).

We now introduce the so-called *Kikuchi matrices*, which are a very versatile tool to translate hypergraph problems into graph problems (of a bigger dimension).

Fix l a positive integer. Given $c \in H$ (so c is a subset with 4 elements of $\{1, 2, \ldots, n\}$), we define the $\binom{n}{l} \times \binom{n}{l}$ matrix K_c having as rows and columns all the subsets S with l elements of $\{1, 2, \ldots, n\}$. The entry corresponding to subsets S and T will have the value

$$(K_c)_{S,T} = 1_{S \Delta T = c} = \begin{cases} 1 & \text{if } S \Delta T = c \\ 0 & \text{otherwise} \end{cases}$$

where $S \Delta T = (S - T) \cup (T - S)$ is the symmetric difference of S and T.

The Kikuchi matrix $K = \sum_{c \in H} K_c$ is just the adjacency matrix of the graph on $\binom{n}{l}$ vertices where vertices S, T (corresponding to two sets with l elements of $\{1, 2, \ldots, n\}$) have an edge iff there exists $c \in H$ such that $S \Delta T = c$ (note that if c exists, it is unique).

The reason why the Kikuchi matrices K_c are useful in this case is that we can write $\Psi(x) = \frac{1}{m}\sum_{c\in H} x_c b_c$ in terms of a quadratic form of the K_cs. To that end, note that we have the following

Lemma 5. Given S a subset with l elements of $\{1, 2, \ldots, n\}$, if we define $y_S = \prod_{j\in S} x_j$, then

$$y^T K_c y = \binom{4}{2}\binom{n-4}{l-2} x_c$$

Proof. This is a simple counting argument. Note that if $S\Delta T = c$, then

$$4 = |c| = |S| + |T| - 2|S \cap T| = 2l - |S \cap T|$$

so $|S \cap T| = l - 2$. This means that S and T each have two elements of c and then $l-2$ other elements in common. To count how many S and T we have such that $S\Delta T = c$, note that the 4 elements in c can be split into two subsets of 2 elements in $\binom{4}{2}$ ways, and then we have to add $l-2$ more elements from the remaining $n-4$ elements, which can be done in $\binom{n-4}{l-2}$. The conclusion follows after noticing that $y_S \cdot y_T = \prod_{i\in S\Delta T} x_i$. □

We can now write

$$\Psi(x) = \frac{1}{m}\sum_{c\in H} x_c b_c = \frac{1}{\binom{4}{2}\binom{n-4}{l-2}} y^T \left(\frac{1}{m}\sum_{c\in H} b_c K_c\right) y$$

2.2.1 A first attempt

We have the following spectral bound

$$\Psi(x) \leq \frac{1}{\binom{4}{2}\binom{n-4}{l-2}} \binom{n}{l} \frac{1}{m} \left\| \sum_{c\in H} b_c K_c \right\|$$

Our algorithm is then taking as input ψ and outputs

$$v(\psi) = \min\left(1, \frac{1}{\binom{4}{2}\binom{n-4}{l-2}} \binom{n}{l} \frac{1}{m} \left\| \sum_{c\in H} b_c K_c \right\|\right)$$

We need to show the utility, i.e. show that there exists $\delta_{m,n} > 0$ such that $v(\psi) < 1 - \delta_{m,n}$ with high probability.

Note that by the matrix Khintchine inequality and Talagrand's concentration inequality,

$$\left\| \sum_{c\in H} b_c K_c \right\| \leq c\sqrt{\log\binom{n}{l}} \left\| \sum_{c\in H} K_c^2 \right\|^{1/2} \leq c\sqrt{l \log n}\sqrt{d_{\max}}$$

with high probability. To see the last inequality, note that

$$(K_c^2)_{S,T} = \sum_U (K_c)_{S,U}(K_c)_{U,T} = \sum_U 1_{S\Delta U=c} \cdot 1_{U\Delta T=c} = 1_{S=T} \sum_U 1_{S\Delta U=c}$$

so K_c^2 is diagonal. We get that the (S,S) entry of $\sum_{c\in H} K_c^2$ is

$$\left(\sum_{c\in H} K_c^2\right)_{S,S} = \sum_{c\in H}\sum_U 1_{S\Delta U=c} = \deg S$$

Let us *assume* again that $d_{\max} \simeq d_{\text{avg}} = \dfrac{2\cdot\binom{4}{2}\binom{n-4}{l-2}m}{\binom{n}{l}}$. Then

$$\begin{aligned}\frac{1}{\binom{4}{2}\binom{n-4}{l-2}}\binom{n}{l}\frac{1}{m}\left\|\sum_{c\in H} b_c K_c\right\| &\le c\frac{1}{\binom{4}{2}\binom{n-4}{l-2}}\binom{n}{l}\frac{1}{m}\sqrt{l\log n}\sqrt{d_{\max}} \\ &= c\sqrt{\frac{l(\log n)\binom{n}{l}}{m\binom{4}{2}\binom{n-4}{l-2}}} \\ &\le c\sqrt{\frac{n^2\log n}{ml}}\end{aligned}$$

Thus, if $m \gtrsim \dfrac{n^2\log n}{l}$, there exists a $\delta_{m,n}$-refutation algorithm with running time $n^{O(l)}$. It's immediate to see that this gives Theorem 4.

Of course, we need a proof which does not rely on the assumption that $d_{\max} \simeq d_{\text{avg}}$. We will see this in the next subsection.

2.2.2 Getting rid of the regularity assumption

Just like before, we define the $\binom{n}{l} \times \binom{n}{l}$ diagonal matrix Γ with entries $\Gamma_{S,S} = \deg S + d_{\text{avg}}$. We then have

Lemma 6. If $K = \sum_{c\in H} K_c$ is the Kikuchi matrix, then

$$\left\|\Gamma^{-1/2} K \Gamma^{-1/2}\right\| \le c\sqrt{\frac{l\log n}{d_{\text{avg}}}}$$

Proof. As in lemma 2, the only thing that needs to be done is to bound

$$\left\|\sum_{c\in H} (\Gamma^{-1/2} K_c \Gamma^{-1/2})^2\right\|^{1/2}$$

We again have that $(\Gamma^{-1/2}K_c\Gamma^{-1/2})^2$ is diagonal and

$$\left(\sum_{c\in H}(\Gamma^{-1/2}K_c\Gamma^{-1/2})^2\right)_{S,S} = \frac{1}{\deg S + d_{\text{avg}}}\sum_{c\in H}\sum_{S\Delta T=c}\frac{1}{\deg T + d_{\text{avg}}} \leq \frac{1}{d_{\text{avg}}}$$

by the same reasoning as before. □

If $m \gtrsim \frac{n^2 \log n}{l}$, we can take

$$v(\psi) = \min\left(1, \left\|\sum_{c\in H} b_c\left(\Gamma^{-1/2}K_c\Gamma^{-1/2}\right)\right\|\right)$$

This gives a $\delta_{m,n}$-refutation algorithm with complexity $n^{O(l)}$. The reasoning is almost identical to that of section 2.1.2.

2.3 A word on the *random* 3-XOR refutation algorithm

For the random 3-XOR problem we can write again the polynomial

$$\Psi(x) = \frac{1}{m}\sum_{c\in H} x_c b_c$$

The basic issue is that 3 is odd so none of the strategies from before work. Nevertheless, we can write

$$\begin{aligned}
\Psi(x) &= \frac{1}{m}\sum_{(i,j,k)\in H} x_i x_j x_k b_{ijk} \\
&= \frac{1}{m}\sum_{i=1}^{n} x_i \cdot \left(\sum_{c\in H,\ c\ni i} x_{c-\{i\}} b_C\right) \\
&\leq \frac{1}{m}\left(\sum_{i=1}^{n} x_i^2\right)^{1/2} \cdot \left(\sum_{i=1}^{n}\left(\sum_{c\in H,\ c\ni i} x_{c-\{i\}} b_c\right)^2\right)^{1/2} \\
&\leq \frac{n}{m}\sum_{C,C'\in H,\ c\ni i,\ c'\ni i} b_c b_{c'} x_{c-\{i\}} x_{c'-\{i\}}
\end{aligned}$$

Now $x_{c-\{i\}}x_{c'-\{i\}}$ contains an even number of variables, so we are in a good spot to try to do similar thing as before. There is a catch however, as the random coefficients are now $b_c b_{c'}$ instead of b_c, so we cannot simply apply the matrix Khintchine inequality. The proof is beyond the scope of the lecture.

3 Random 3-SAT refutation via random k-XOR refutation

Given a 3-SAT clause c, say $x_1 \vee x_2 \vee x_3$, we define

$$\Psi_c(x) = \frac{7}{8} + \frac{1}{8}\left(x_1 + x_2 + x_3 - x_1x_2 - x_2x_3 - x_3x_1 + x_1x_2x_3\right)$$

where we view -1 as false and 1 as true. Note that $\Psi_c(x)$ is 1 if and only if $x = (x_1, \ldots, x_n)$ satisfies the clause c and is 0 otherwise. Define

$$\Psi(x) = \sum_{c\in H}\Psi_c(x)$$

Then

$$\Psi(x) < (1 - \delta)m$$

will mean that a fraction of at least δ of the clauses must be violated by x. Given the bounds for random k-XOR, we can readily infer a bound for random 3-SAT as well.

References

[AKK95] Sanjeev Arora, David R. Karger, and Marek Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. *In Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, 29 May-1 June 1995, Las Vegas, Nevada, USA, pages 284–293. ACM, 1995.

[FG01] Joel Friedman and Andreas Goerdt. Recognizing more unsatisfiable random 3-SAT instances efficiently. In *Automata, languages and programming*, volume 2076 of *Lecture Notes in Comput. Sci.*, pages 310–321. Springer, Berlin, 2001

[FLP16] Dimitris Fotakis, Michael Lampis, and Vangelis Th. Paschos. Sub-exponential approximation schemes for csps: From dense to almost sparse. In *33rd Symposium on Theoretical Aspects of Computer Science*, STACS 2016, February 17-20, 2016, Orléans, France, volume 47 of LIPIcs, pages 37:1–37:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[GK01] Andreas Goerdt and Michael Krivelevich. Efficient recognition of random unsatisfiable k-SAT instances by spectral methods. In *STACS 2001 (Dresden)*, volume 2010 of *Lecture Notes in Comput. Sci.*, pages 294–304. Springer, Berlin, 2001

[RRS17] Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random csps below the spectral threshold. In *STOC*, pages 121–131. ACM, 2017.