An Exposition of Dinur-Khot-Kindler-Minzer-Safra's Proof for the 2-to-2 Games Conjecture*

Mitali Bafna Chi-Ning Chou Zhao Song

April 18, 2018

Abstract

In a recent paper [KMS18] proved a certain combinatorial hypothesis, which completed the proof of (the imperfect completeness variant of) Khot's 2-to-2 games conjecture (based on an approach initiated by [KMS17]). This was a major step towards proving the unique games conjecture and the strongest evidence we have that the UGC is true.

The proof is obtained by combining the following results. In [DKK+16] the authors had reduced the 2-to-2 games conjecture to a certain hypothesis about an agreeement test on the Grassman graph. Later, [BKS18] proved that a so called "inverse short code conjecture" on the short code graphs [BGH+15] implies the Grassman agreement hypothesis of [DKK+16]. Finally, [KMS18] (building on techniques in [DKK+17]) proved the latter "inverse short code conjecture" thus completing the proof of the 2-to-2 games conjecture.

In these notes we provide an exposition of why the inverse short code conjecture implies the 2-2 games conjecture, without going into Grassman graphs. Assuming the inverse short code conjecture, our goal is to reduce (a variant of) smooth label cover, which is known to be NP-hard to 2-1 games thus proving its NP-hardness. In this first version of the notes though, we will only reduce from unique games to unique games over shorter alphabet size. In the next version, we will generalize the above to reduce from smooth label cover.

*These are notes from Boaz Barak's Feb 23rd and March 2nd 2018 talks at CMSA. Scribes: Mitali Bafna, Chi-Ning Chou, and Zhao Song.

Contents

1 Preliminaries

In this section, we provide some background for understanding the whole proof. We decided to give a complete overview of this topic instead of going to the details immediately and hopefully can let the reader have a better big picture of what is going on.

1.1 Notations

Let us put some notations here without explanation. Things will become clearer and clearer and please refer to here when you feel confused.

- Parameters: Numbers of variables $n, m \geq 1$. The length of the original alphabet size $D \geq 1$. The log[1] of the alphabet ratio d. The length of the reduced alphabet size $\ell \geq 1$. Soundness of the original game $\epsilon \in (0,1)$ and soundness of the reduced game $\delta \in (0,1)$. The dimension of monochromatic set $r \geq 1$. The goodness of monochromatic set $\tau \in [0,1]$. See subsection 4.1 for discussion on the relation among parameters.
- Math objects: $\mathbb{F}_2$, $\textsc{Lin}(n,\ell)$ is the set of all possible linear function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^\ell$. $\mathcal{A}(D,\ell)$ is the set of all affine functions from $\mathbb{F}_2^D$ to $\mathbb{F}_2^\ell$. $\mathcal{R}^1(D,\ell)$ is the set of all rank one linear functions from $\mathbb{F}_2^D$ to $\mathbb{F}_2^\ell$. Concretely, for any $e \in \mathcal{R}^1(D,\ell)$, there exists $e' \in \mathbb{F}_2^{\ell \times D}$ rank one such that $e(x) = e'x$ for any $x \in \mathbb{F}_2^D$.
- For $v \in \mathcal{A}(D,\ell)$, $[v]$ is the canonical representation of v under invertible affine transformation.
- Label-Cover related: I is CSP instance, $X = \{x_i\}, Y = \{y_j\}$ are variable sets, $\Sigma_{\text{Big}}, \Sigma_{\text{Small}}$ are alphabet sets, and $\Pi = \{f_{i,j}\}$ is a collection of constraints. P_1, P_2 are provers and V is the verifier in the game view.

1.2 Definitions

Let us start with the definition of *constraint satisfaction problem (CSP)*.

Definition 1.1 (CSP)**.** *A constraint satisfaction problem (*CSP*) instance $I = (X, \Sigma, \Pi)$ is given by a variable set X, an alphabet set Σ, and a collection of constraints Π where $\pi : \Sigma^{|X|} \to \{0,1\}$ for each $\pi \in \Pi$. Let $c : X \to \Sigma$ be a assignment (or coloring) for I, we define its value to be $\mathrm{val}(c) = \mathbb{E}_{\pi \sim \Pi}[\pi(c(X))]$. Define the value of I to be the maximum over all possible assignment, i.e., $\mathrm{val}(I) = \max_c \mathrm{val}(c)$. We say that I is satisfiable if $\mathrm{val}(I) = 1$.* ♢

Many familiar computational problems can be formulated into the form of CSP, *e.g.,* 3SAT, 3Lin, MaxCut, etc. Note that CSP and many of its special cases are **NP**-hard. As a result, a natural question is to study in CSP is the approximation. This needs the definition of Gap-CSPdefined as follows.

Definition 1.2 (Gap-CSP)**.** *Let $0 \leq s < c \leq 1$. The (c,s)-*Gap-CSP *problem is to determine, given a* CSP *instance I whether :*

- $\mathrm{val}(I) \geq c$*, in which case we say I is a YES instance of (c,s)-*Gap-CSP*.*
- $\mathrm{val}(I) \leq s$*, in which case we say I is a NO instance of (c,s)-*Gap-CSP*.*

Here, c refers to completeness and s refers to soundness. ♢

It turns out that many special cases of Gap-CSP are also **NP**-hard. For instance, Håstad [Hås01] showed that $(1, 7/8 + o(1))$-Gap-3SAT is **NP**-hard, which is tight in the sense that there is an efficient greedy algorithm for $(1, 7/8)$-Gap-3SAT.

However, many of the special cases of CSP do not have tight approximation results. For example, Goemans and Williamson [GW95] designed a polynomial time algorithm[2] for $(1, 0.878)$-Gap-MaxCut while

[1] Sometimes we use d directly as the alphabet ratio. It should be clear depending on the context.

[2] This algorithm is a SDP-based algorithm and can be formulated as degree-2 *Sum-of-Squares* algorithm.

the best known **NP**-hardness result is for $(1, 0.941)$-Gap-MaxCut. See Figure 2 in [Kho10] for more examples. Nevertheless, in a seminal paper [Kho02], Khot identified an important special case of Gap-CSP: the Unique-Games, in which the approximation results of many special cases of CSP become tight if $(1-o(1), \delta)$-Gap-Unique-Games is **NP**-hard. Khot then conjectured $(1-o(1), \delta)$-Gap-Unique-Games to be **NP**-hard and this has been known as the *Unique Games Conjecture (UGC)*.

After motivating the study of Unique-Games, let us wrap up this subsection by its formal definition. Unique-Games is a special case of Label-Cover, which is also a special case of CSP defined as follows

Definition 1.3 (Label-Cover)**.** *A* Label-Cover *instance is a* CSP *instance* $(X, Y, \Sigma_{\text{Big}}, \Sigma_{\text{Small}}, \Pi)$ *where* $X = \{x_i\}_{i\in[n]}, Y = \{y_j\}_{j\in[m]}$ *are variable sets,* $\Sigma_{\text{Big}}, \Sigma_{\text{Small}}$ *are the corresponding alphabet sets, and* Π *contains constraints of the form* $y_i = f_{i,j}(x_i)$*. We say it is a* d*-to-*1 *Games if all the constraints are* d*-to-*1 *and thus* $|\Sigma_{\text{Big}}| = d \cdot |\Sigma_{\text{Small}}|$*. Specifically, if* $d = 1$*, we call it Unique Games.* ◊

1.3 Historical background

Håstad [Hås01] gave optimal inapproximability results for various problems, such as 3LIN, 3SAT and Clique. One of their main results was showing that approximating the Label-Cover is hard which was the starting point of many hardness of approximation results, including our trial of proving d-to-1 games conjecture.

Theorem 1.4 (Håstad [Hås01])**.** *For every* $\delta > 0$ *there is some* d*,* $\Sigma_{\text{Big}}, \Sigma_{\text{Small}}$*, such that it is* **NP***-hard to distinguish between a* d *to* 1 *game over these alphabets with value* 1 *(completeness) and value* $\leq \delta$ *(soundness).*

Moreover, Moshkovitz and Raz [Hås01] showed that the reduction can be made to have *quasilinear* blowup (e.g. $n^{1+o(1)}$) which implies that under the exponential time hypothesis (ETH), for every $\epsilon > 0$, there is no algorithm with running time $2^{n^{1-\epsilon}}$ to approximate 1 vs δ d-to-1 games.

In the above though, the ratio between the sizes of the alphabet sets was allowed to depend on the soundness parameter, specifically, $d(\delta) \approx 2^{\text{poly}(1/\delta)}$. Khot conjectured that the label cover problem is not only hard for large enough d, but for every d and every δ the above is true. The d-to-1 conjecture is as follows,

Conjecture 1.5 (Khot [Kho02])**.** *For any* $d > 1$*, the* d*-to-1 Conjecture states that for any* $\delta > 0$*, the* $(1, \delta)$-Gap-d-to-1-Games *is* **NP***-hard when the alphabet set is large enough*[3]*. When* $d = 1$*, the Unique Games Conjecture states that for any* $\delta > 0$*, the* $(1-\delta, \delta)$-Gap-Unique-Games *is* **NP***-hard when the alphabet set is large enough.*

Note on perfect completeness. Khot phrased the d-to-1 conjecture with perfect completeness for $d > 1$. However, in this paper we would consider the slightly weaker *imperfect completeness* variant. For the case $d = 1$ (i.e., unique games), imperfect completeness is essential for hardness, as there is a polynomial-time algorithm in the perfect completeness case. Moreover, looking ahead, in these notes we will be only interested in d to 1 games with affine constraints, in which case the Gaussian elimination algorithm shows that imperfect completeness is essential for hardness regardless of d. In particular, we will use the (known) variant of Theorem 1.4 where the constraints are affine and the completeness parameter is $\geq 1 - \delta$.

On the algorithmic side, [ABS10] gave a sub-exponential Sum-of-Squares algorithm for approximating the d-to-1 problem. (Specifically, [ABS10] gave an algorithm for the unique games problem, while Steurer [Ste10] analyzed the same algorithm for the d-to-1 games as well.)

Theorem 1.6 (Arora-Barak-Steurer [ABS10], Steurer [Ste10])**.** *For every* $d \geq 1$ *and* $\delta > 0$ *there exist a constant* $c > 0$ *and an algorithm that runs in time* $2^{n^{O(dn^{\beta(d,c,\delta)})}}$ *for* (c, δ)-Gap-d-to-1-Games *where for every fixed* c *and* d*,* $\beta(d, c, \delta)$ *tends to zero as* δ *tends to zero.*[4]

The consequences of Theorem 1.6 and Conjecture 1.5 are very interesting in the following way. (i) intermediate complexity (ii) necessary blowup in gadget reduction.

[3] The *large enough* here only depends on δ and d but not on n.

[4] Specifically, for every fixed d and c, $\beta(d, c, \delta) = O(1/\operatorname{poly}\log(1/\delta))$. For the case $d = 1$ and $c = 1 - \eta$, $\beta(1, 1-\eta, \delta) = \operatorname{poly}(\eta)/\operatorname{poly}\log(1/\delta)$.

Intermediate complexity If Conjecture 1.5 is true for some d, then modulo our belief in $\mathbf{P} \neq \mathbf{NP}$, there is no polynomial time algorithm for d-to-1 Games problem and in fact under the ETH the best running time for this problem is at least 2^{n^β} for some $\beta > 0$. However, Theorem 1.6 provided a sub-exponential time algorithm for d-to-1 Games. That is, modulo d-to-1 Games conjecture and $\mathbf{P} \neq \mathbf{NP}$, there is a constraint satisfaction problem whose optimal algorithm runs strictly between polynomial time and exponential time! This is quite surprising as a priori one might have expected that CSP's have a "threshold" or "zero one law" type of behavior where for some approximation ratio the running time is polynomial and for others the running time is $2^{\Omega(n)}$. For example, a consequence of the ETH and the (now proven) dichotomy conjecture, this is the behavior of the *exact solvability* problem for all CSP's. See Figure 1 and Boaz's blog post for more details in intermediate complexity.

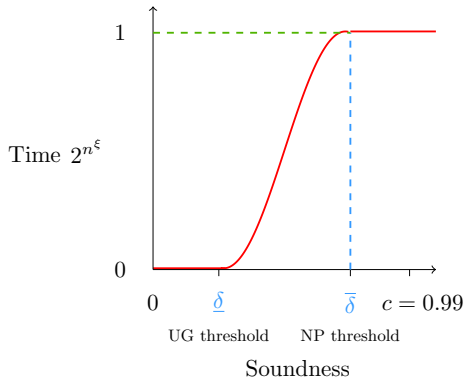


Figure 1: A qualitative schematic of the running time (in log of exponent scale) as a function of approximation quality for (c, δ)-Gap-d-to-1-Games. Think of the case that c is some constant such as 0.99 or even 0.49 and we vary the soundness parameter δ. A priori one might have thought that there would be a *threshold* δ^*, such that (c, δ)-Gap-d-to-1-Games would be solvable in polynomial time for $\delta < \delta^*$ and would require $2^{\tilde{\Omega}(n)}$ time for $\delta > \delta^*$. However under the ETH, combining the proof of the 2 to 2 conjecture with the [ABS10] algorithm we see that there are actually *two thresholds* $0 < \underline{\delta} < \overline{\delta} < 1$. For $\delta < \underline{\delta}$ the problem is solvable in polynomial time, for $\delta > \overline{\delta}$ the problem requires exponential (i.e. $2^{\tilde{\Omega}(n)}$ time), while for $\delta \in (\underline{\delta}, \overline{\delta})$ the problem is solvable in time $2^{n^{\xi(\delta)}}$ for some $\xi(\delta)$ that is strictly between 0 and 1. The unique games conjecture precisely predicts the value of $\underline{\delta}$, but the value of $\overline{\delta}$ and the shape of $\xi(\delta)$ are not known.

Necessary blowup in gadget reduction The famous *Exponential Time Hypothesis (ETH* [IP01] and its stronger version *Strong Exponential Time Hypothesis (SETH)* [IPZ01] conjecture that any deterministic algorithm for SAT requires exponential time. If we believe ETH or even SETH are true, then Theorem 1.6 implies that one cannot hope for gadget reduction from standard label cover to Unique-Games with only linear blowup. Otherwise, it would yield a sub-exponential algorithm for SAT, which contradicts to ETH and SETH. We summarize this consequence in the following corollary.

Corollary 1.7. *Theorem 1.6 implies that d-to-1 conjecture cannot be proven via gadget reduction from standard hardness result of* Label-Cover *such as Theorem 1.4.*

1.4 Three views of Label-Cover

In this subsection, we are going to introduce the three views of Label-Coverfor future usage and building intuition. The three views are the CSP view, the game view, and the graph view.

CSP view The CSP view of Label-Cover is basically the same as what we defined in Definition 1.1 and Definition 1.3.

Game view Label-Cover can also be viewed as a *2-Prover-1-Round game* [Kho02] as follows. Let $I = (X, Y, \Sigma_{\text{Big}}, \Sigma_{\text{Small}}, \Pi)$ be a Label-Cover instance. We can play it in the following *two-prover one-round game*. Let P_1, P_2 be two provers without communication[5] and V be the verifier. Given the Label-Cover instance I, the verifier uniformly sample a constraint $f_{i,j} \in \Pi$ and send x_i to P_1, y_j to P_2. The provers then send $\tilde{x}_i \in \Sigma_{\text{Big}}$ and $\tilde{y}_j \in \Sigma_{\text{Small}}$ to V. Finally, V output 1 if and only if $\tilde{y}_j = f_{i,j}(\tilde{x}_i)$. See Figure 2.

Note that the probability of V to output 1 is equal to the $\text{val}(C)$ and the proof is left as an exercise.

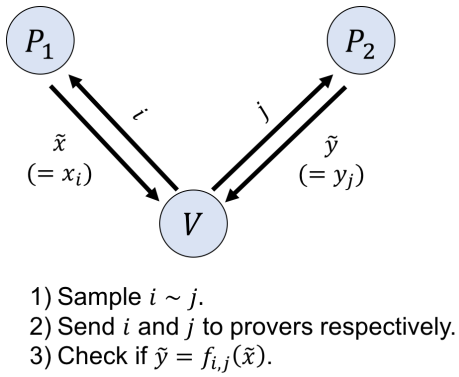


Figure 2: The game view of Label-Cover.

Graph view Given a Label-Cover instance I, one can define its constraint graph by treating each variable as vertex and and two vertices have an edge if and only if there is a constraint on the corresponding variables. Note that the constraint graph of Label-Cover instance is a bipartite graph. An assignment for the variables corresponds to a *coloring* of the constraint graph. The edge is then thought of as the constraint on the coloring of the two vertices. See Figure 3.

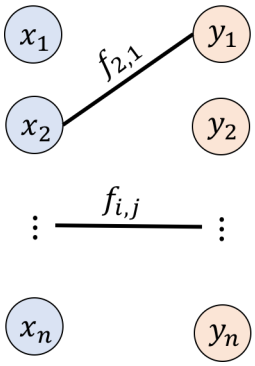


Figure 3: The graph view of Label-Cover. The provers is supposed to send the value of x_i and y_i back to the verifier.

Now we have enough background to understand the reduction and prove the NP-hardness of 2-1 games!

2 Reduction

Now, it is time for hardness reduction. Let us start with an overview of the reduction from the very beginning of the story. The starting point is the well-known **NP**-hard problem SAT. In [Hås01], SAT is reduced to

[5]Nevertheless, they might share some randomness before the game starts though this can be shown not to matter by the averaging argument.

3XOR with completeness $1-o(1)$ and $1/2+o(1)$. By some folklore[6], it can then be reduced to Label-Cover with some smoothness property that is going to be useful in the future. Finally, our goal is to reduce Label-Cover to Unique-Games with completeness $1/2-o(1)$ and soundness δ for arbitrary $\delta > 0$. The final step was done by [DKK+16] and [BKS18] modulo certain combinatorial theorem proved in [KMS18].[7] The following is a summary of the road map though here we only focus on the last step.

$$\text{SAT} \xRightarrow{\text{[Hås01]}} \text{3XOR}_{1-o(1),\frac{1}{2}+o(1)} \xRightarrow{\substack{\text{Smooth}\\ \text{Tensoring}}} \text{LC} \xRightarrow[\text{[BKS18]}]{\text{[DKK+16]}} \text{UG}_{\frac{1}{2}-o(1),\delta}.$$

Specifically, the last step from Label-Cover to Unique-Games had been shown in the following way. Dinur et. al. [DKK+16] first reduced Label-Cover to the 2-to-2 games with completeness $1-o(1)$ and soundness δ modulo the so called *Grassman test conjecture*. Then, the reduction to Unique-Games with completeness $1/2-o(1)$ and soundness δ is then due to some standard connection between 2-to-2 games and Unique-Games. In these notes, our goal is to directly reduce Label-Cover to Unique-Games.

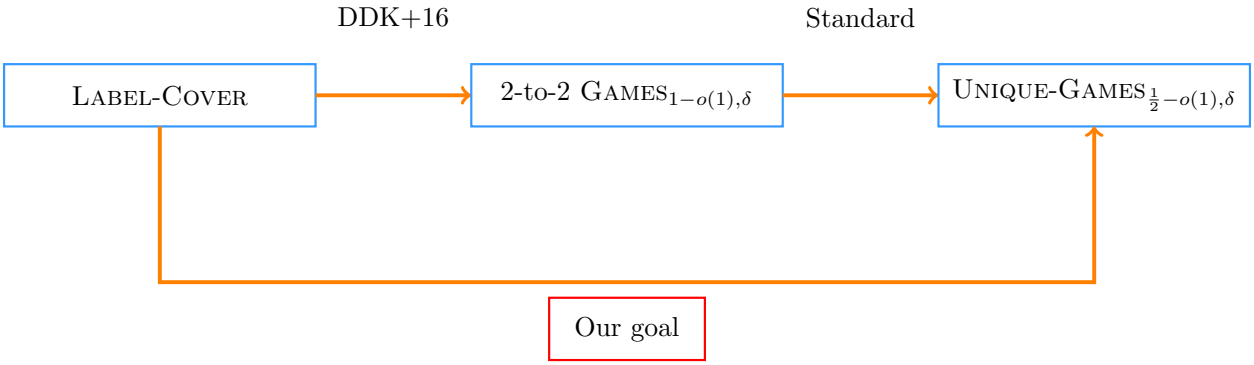


Figure 4: The game view of PCP composition paradigm.

Affine label cover. Starting from now, we are given Label-Cover instance I with constraints $\{y_j = f_{i,j}(x_i)\}$ where each $f_{i,j}$ is an *affine* map from $\mathbb{F}_2^D$ to $\mathbb{F}_2^{D-d}$. That is, we are considering a 2^d-to-1 game. We know that it is **NP**-hard to distinguish between $\text{val}(I) \geq 1-o(1)$ and $\text{val}(I) \leq \delta$ where $\delta \to 0$ when $d \to \infty$. Our goal here is to transform I into Unique-Games instance with constraints $\{\tilde{y}_j = g_{i,j}(\tilde{x}_i)\}$ where $g_{i,j} : \mathbb{F}_2^\ell \to \mathbb{F}_2^\ell$. The hardness we are aiming for is completeness $1/2-o(1)$ and soundness δ for arbitrary δ.


It is a good time for the reader to stop here and make sure understanding what we have and what we want.

Before we formally introduce the reduction, which in fact can be written within five lines, let us digress to the standard PCP composition paradigm and motivate the reduction. For those who are familiar with PCP composition paradigm, you can jump to the *code concatenation* part.

[6] Boaz will probably cover this in the second lecture.

[7] This is not a chronological discussion. The first paper to introduce the ideas of using the Grassman graph in this context was [KMS17]. [DKK+16] used these ideas to give a proof of the 2 to 2 conjecture modulo a combinatorial hypothesis on the soundness of an agreement test on the Grassman graph, and [DKK+17] gave preliminary results on the expansion on the Grassman graph. Then [BKS18] showed that an inverse expansion hypothesis on either the Grassman or Short code graph will imply the hypothesis of [DKK+16], hence completing the proof of the 2 to 2 conjecture. Finally, [KMS18] proved the inverse expansion hypothesis for both the short code and Grassman graphs.

2.1 PCP composition paradigm for Label-Cover

Label-Cover has been used in many hardness of approximation results and most of them follow the standard PCP composition paradigm explained as follows. This paradigm is also called code concatenation or alphabet reduction in the literature. In the following, we introduce a special case of PCP composition paradigm in order to motivate the our reduction. For the interested readers, please refer to classic papers such as [Hås01] or survey [Kho10].

Given a Label-Cover instance I with alphabet set[8] Σ. Let $E : \Sigma \to \Sigma_{\text{target}}^{\ell}$ be an *error-correcting code (ECC)* mapping from the original alphabet set Σ to a length-ℓ string of elements in the new alphabet set Σ_{target}. For each assignment $x \in \Sigma$ to a variable in the original instance I, we map it to $E(x)$. For each constraint $y_j = f_{i,j}(x_i)$ in the original instance I, we map it to a collection of constraints in the form $\{\tilde{y}_{j,v} = g_{i,j,u,v}(\tilde{x}_{i,u})\}$ where (i) $\tilde{x}_{i,u}$ is supposed to be $E(x_i)$ (resp. $\tilde{y}_j$ is supposed to be $E(y_j)$), (ii) $u, v \in [\ell]$ are the coordinate of $E(x_i)$ and $E(y_j)$ respectively, and (iii) g is a function from Σ_{target} to Σ_{target}. To get more feeling about the PCP composition paradigm described above, let's illustrate it in the three different views introduced before.

CSP view What we described above is exactly the CSP view of PCP composition paradigm. See Table 1 for a comparison of the original CSP and the new CSP.

	Original CSP	New CSP
Variable sets	$\{x_i\}$ and $\{y_j\}$	$\{\tilde{x}_{i,u}\}$ $(= E(x_i)_u)$ $\{\tilde{y}_{j,v}\}$ $(= E(y_j)_v)$
Alphabet set	Σ	$\Sigma_{\text{target}}^{\ell}$
Constrains	$y_j = f_{i,j}(x_i)$	$\tilde{y}_{j,v} = g_{i,j,u,v}(\tilde{x}_{i,u})$

Table 1: CSP view of PCP composition paradigm.

Game view Recall that the original game view is in Figure 2. In the 2-Prover-1-Round game for the new instance, the verifier first sample $i \sim j$ as usual. Then, she samples $u, v \in [\ell]$, which are *coordinates* of the encodings $E(x_i)$ and $E(y_j)$. The provers receive queries (i, u) and (j, v) respectively and are supposed to return $E(x_i)_u$ and $E(y_j)_v$. See Figure 5

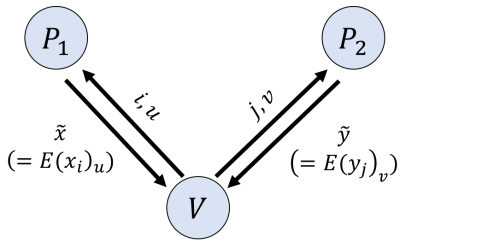


Figure 5: The game view of PCP composition paradigm.

Note that the difference between the original game and here is that the provers here can *cheat* in the sense that his corresponding strategy does not correspond to a valid encoding. Concretely, denote the responding strategy of prover 1 as $F : [n] \times [\ell] \to \Sigma_{\text{target}}$. On query $(i, u) \in [n] \times [\ell]$, $F(i, u)$ is supposed to be $E(x_i)_u$ for

[8]For convenience, here we think of the two alphabet sets are the same.

some $x_i \in \Sigma$. However, it could be the case that when we fix an $i \in [n]$, $\{F(i,u)\}_{u\in[\ell]}$ does not correspond to a valid encoding of E. This kind of *malicious prover* makes the soundness analysis of PCP composition paradigm non-trivial.

Graph view Recall that the original graph view is in Figure 3. In the constraint graph of the new instance, each vertex in the original graph becomes a *cloud* of vertices. For each $i \in [n]$, the i^{th} cloud contains ℓ vertices indexed by $x_{i,u}$ for some $u \in [\ell]$. Each constraint $g_{i,j,u,v}$ is associated with an edge between $x_{i,u}$ and $y_{j,v}$. The colors for each vertex is now from the alphabet set Σ_{target}. See Figure 6.

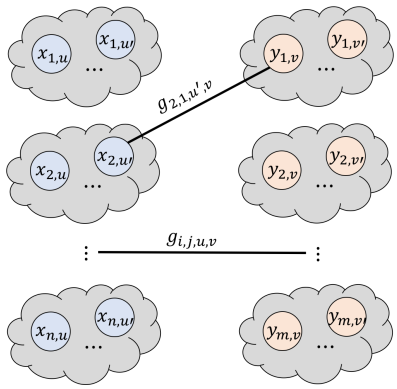


Figure 6: The graph view of PCP composition paradigm.

Apart from the three views illustrated above, another way to view the PCP composition paradigm is from the perspective of coding theory. This is the reason why sometimes people also call PCP composition paradigm *code concatenation*.

> Make sure you understand the PCP composition paradigm before you move on.

Code concatenation Code concatenation is a standard technique used in the construction of PCPs to reduce a large alphabet to a smaller one. If we have a PCP over the alphabet Σ, we encode the symbols of Σ using an error correcting code over a smaller alphabet size, i.e. $E : \Sigma \to \Sigma_{\text{Small}}^{L}$. This code is chosen in such a way that it plays well with the functions computed by the verifier. For example, each coordinate in the encoding $E(x_i)$ corresponds to an evaluation of an affine function on x_i. Table 2 lists some important codes used in the PCP literature.

Code	Σ_{Small}	Length	Encoding
Long Code	$\mathbb{F}_2$	$2^{2^{O(D)}}$	Every function
Short Code	$\mathbb{F}_2$	$2^{O(D^2)}$	Affine functions
Hadamard	$\mathbb{F}_2$	2^D	Affine functions
ℓ-tensored Hadamard or unbalanced short code	$\mathbb{F}_2^\ell$	$2^{O(\ell D)}$	Affine functions

Table 2: Some common codes that have been used in code concatenation. The last one is the code we are going to use.

The code we will use is the ℓ-tensored Hadamard code, a.k.a. the unbalanced short code, defined as follows. Let the original alphabet set to be $\mathbb{F}_2^D$. The encoding function E maps each symbol in $\mathbb{F}_2^D$ to a string over the alphabet $\mathbb{F}_2^\ell$ where $\ell \ll D$. That is, E decreases the size of the alphabet sets. As to the length of the codeword $E(x)$, here $E(x)$ has $|\mathcal{A}(D,\ell)| \approx 2^{\ell D+\ell}$ coordinates where each coordinate corresponds to an affine function $u : \mathbb{F}_2^D \to \mathbb{F}_2^\ell$. Recall that $\mathcal{A}(D,\ell)$ contains all affine function from $\mathbb{F}_2^D$ to $\mathbb{F}_2^\ell$. Finally, for each $x \in \mathbb{F}_2^D$ and $u \in \mathcal{A}(D,\ell)$, $E(x)_u$ is naturally defined as $u(x)$. Note that $\ell = 1$ gives us the usual Hadamard code and hence this is just an ℓ-tensored Hadamard code if we ignore folding[9]. See Figure 7.

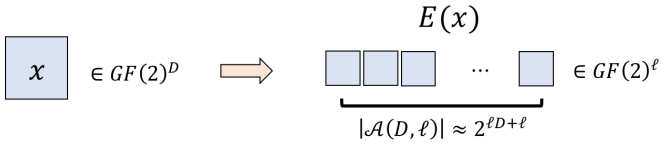


Figure 7: The ℓ-tensored Hadamard code without folding.

We need to have an explicit description for each affine function $u \in \mathcal{A}(D,\ell)$ so that the encoding can be efficiently used in our reduction. The simplest way is viewing u as a matrix $A_u \in \mathbb{F}_2^{\ell \times D}$ and a vector $w_u \in \mathbb{F}_2^\ell$ such that $u(x) = A_u x + w_u$. Note that the matrix A_u is not necessary full-rank[10]. We call such u *degenerate*.

Folding Notice that the codeword $E(x)$ has a lot of redundancies, there are many indices of E that differ by an *invertible function*. Concretely, let $u \in \mathcal{A}(D,\ell)$ and $g : \mathbb{F}_2^\ell \to \mathbb{F}_2^\ell$ be an invertible affine function. For a valid codeword $E(x)$, we have that,

$$E(x)_{g\circ u} = g \circ u(x) = g(E(x)_u).$$

Namely, once we know the value of $E(x)_v$, we know the value of $E(x)_{g\circ v}$. This might not necessarily hold for an invalid codeword, but nevertheless we would like to impose this condition. It is then natural to define the *equivalence class* for u as follows.

$$[u] = \{g \circ u : \ g : \mathbb{F}_2^\ell \to \mathbb{F}_2^\ell \text{ affine and invertible}\}.$$

When we see $[u]$, we think of it as the *canonical representation*[11] of the equivalence class. As a result, the encoding no longer needs to store the value for every element in $\mathcal{A}(D,\ell)$. Instead, the codeword only contains the value of elements in $[\mathcal{A}(D,\ell)] := \{[u] : \ u \in \mathcal{A}(D,\ell)\}$. Note that even *degenerate* u, have their own equivalence class. We do not have to use the folded code: we can also run all our protocols directly on the unfolded version by appropriately randomizing the verifier's queries. The folded code though gives us a simpler analysis.

Finally, we can now defined the ℓ-tensored Hadamard code to be the folded version of the code we introduced previously. That is, $E : \mathbb{F}_2^D \to [\mathcal{A}(D,\ell)]$ and $E(x)_{[u]} = [u](x)$ for any $x \in \mathbb{F}_2^D$ and $[u] \in [\mathcal{A}(D,\ell)]$.

[9] We will come to folding in a minute

[10] This is a minor issue as only a very small fraction of the matrices are not full rank. This is actually the main difference between the Grassman code used in [DKK+16] and the code we are using here. More details will be provided later.

[11] Canonical representation can be defined as the smallest element in the equivalence class according to some ordering say lexicographical order.

V There are two ways to interpret this code. For convenience of notation, here we use the unfolded version of the code.

- As we did in the previous discussion, treat a coordinate of $E(x)$ as an affine functions from $\mathbb{F}_2^D$ to $\mathbb{F}_2^\ell$ and E assigns value to each affine function.
- Think of E as a linear function from $E : \mathbb{F}_2^D \to (\mathbb{F}_2^\ell)^{2^{\ell D}}$ [a]. Given $x \in \mathbb{F}_2^D$, the codeword $E(x) \in (\mathbb{F}_2^\ell)^{2^{\ell D}}$ is indexed by subspaces of dimension exactly ℓ, where each entry of $E(x)$ is the *restriction* of x on an ℓ dimensional subspace S. Concretely, let $a_1, a_2, \ldots, a_\ell$ be the canonical basis for S. Define $E(x)_S = (\langle a_1, x\rangle, \langle a_2, x\rangle, \ldots, \langle a_\ell, x\rangle) \in \mathbb{F}_2^\ell$. Note that $\ell = 1$ gives us the usual Hadamard code and hence this is just an ℓ-tensored Hadamard code.

Note that these two codes are the same modulo the fact that our code contains even smaller subspaces whereas the first encoding only consists of ℓ dimensional subspaces. [DKK+16] follow the second view whereas we will analyze the first view.

[a] The length of the codeword is the number of ℓ-dimension subspaces in $\mathbb{F}_2^D$. This number is roughly $2^{\ell D}$ so we think of it as $2^{\ell D}$ here for simplicity.

Codeword test Suppose one had to test whether a given string $E \in \mathbb{F}_2^{2^{\ell D}}$ is a valid codeword, one natural test would be choosing two $\ell \times D$ matrices M_1, M_2 and test whether $E_{M_1} + E_{M_2} = E_{M_1+M_2}$. This test has completeness 1 but uses 3 queries, which is undesirable for us. If we were only allowed two queries though, one possible test would be to choose M_1 at random from all $\ell \times D$ matrices, choose M_2 at random from the space of rank-1 $\ell \times D$ matrices and check whether $E_{M_1} = E_{M_1+M_2}$. Note that a rank one matrix is of the form $ab^\top$, for some vectors a, b and hence for all x, $M_2 x = ab^\top x = 0^\ell$ with probability $1/2$. Hence this test has completeness only $1/2$. We will use a similar test for the label cover game with reduced alphabet size.

A final remark about the code concatenation we are using. Because we want to reduce to Unique-Games, we need to map both Σ_{Big} and Σ_{Small} to an alphabet set of the same size. Concretely, in the following we think of $\Sigma_{\text{Big}} = \mathbb{F}_2^D$ and $\Sigma_{\text{Small}} = \mathbb{F}_2^{D-d}$ where D is a super large constant and d is quite small comparing to D. Note that here we start from 2^d-to-1 Games. To make this a unique game, we will reduce both Σ_{Big} and Σ_{Small} to $\mathbb{F}_2^\ell$ on both sides, for some $\ell \ll D - d$, to be chosen later. Now, we are ready for the reduction from 2^d-to-1 Games to Unique-Games. We present the reduction in the game view of PCP composition paradigm.

S Before we move to the protocol, please make sure you understand what is our goal right now. Especially, make sure you feel comfortable with PCP composition paradigm and the game view of the reduction.

2.2 Protocol

Recall that a reduction in PCP composition paradigm under game view is basically designing a protocol with a code E. See Figure 5. In the following, we let $\Sigma_{\text{Big}} = \mathbb{F}_2^D$ and $\Sigma_{\text{Small}} = \mathbb{F}_2^{D-d}$ and our goal is reducing a 2^d-to-1 instance to a UNIQUE-GAMES instance.

Let us start with the unfolded version to motivate the usage of the folded version. As described in the previous section, we replace each symbol $x_i \in \mathbb{F}_2^D, y_j \in \mathbb{F}_2^{D-d}$ by its codeword, $E(x_i), E(y_j)$. The verifier instead of asking for the value of $y_j \in \mathbb{F}_2^{D-d}$ to Prover 2(P2), asks P2 to give $\tilde{y} = E(y_j)_v$ which is in $\mathbb{F}_2^\ell$, for a random function $v \in \mathcal{A}(D-d, \ell)$. We want to finally check whether $y_j = f_{i,j}(x_i)$ which implies that $v(y_j) = v(f_{i,j}(x_i))$. If we define $u = v \circ f_{i,j}$ and ask P1 to give us $\tilde{x} = E(x_i)_u$, then for a valid assignment we would have that $\tilde{x} = \tilde{y}$ and this test would have completeness 1. See Figure 8.

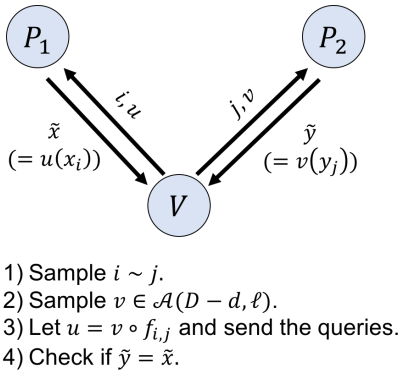


Figure 8: The first protocol without folding.

However, one can immediately come up with a trivial attack for this protocol: both provers always return 0^ℓ. This will give soundness 1, which is meaningless. The way to solve this issue is by *folding* v with all functions $v' = gv$, for invertible functions g and representing this equivalence class via its canonical representation $[v]$. Concretely, the verifier sends $[u]$ and $[v]$ to P1 and P2 respectively and checks whether $v \circ [v]^{-1}(\tilde{y}) = u \circ [u]^{-1}(\tilde{x})$. See Figure 9. We note that as an alternative to folding, the verifier can "randomize" the question by sending v to P1 but sending $g \circ v$ to P2 for a random invertible affine $g : \mathbb{F}_2^\ell \to \mathbb{F}_2^\ell$. It can then apply g^{-1} to P2's answer before running the test above.

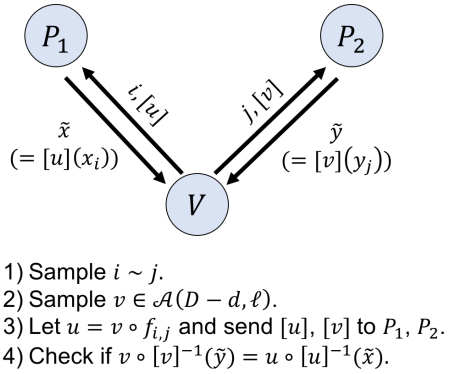
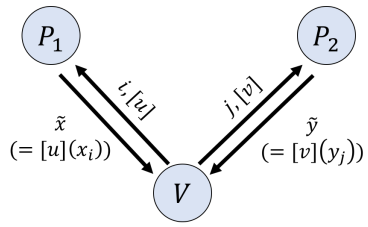


Figure 9: The noiseless protocol with folding.

This test has completeness 1, so have we proved the UGC?! It turns out that there exists an attack even for the simple instance where the constraints are all an affine function of the type equal to an identity

plus a shift. Under some mild assumption, one can even show that the attack succeeds with probability 1. See subsection 2.3 for details. There are two high-level message from the attack. (i) The noiseless protocol is vulnerable under easy instances. (ii) The attack highly exploits the affine nature of the code.

To circumvent the attack, we use a test like one described in the previous section. Specifically, we let $u = v \circ f_{i,j} + e$, where e is chosen uniformly from $\mathcal{R}^1(D-d,\ell)$. Recall that $\mathcal{R}^1(D-d,\ell)$ contains all rank 1 affine functions from $\mathbb{F}_2^{D-d}$ to $\mathbb{F}_2^{\ell}$. We call this protocol the noisy protocol. See Figure 10.



1) Sample $i \sim j$.
2) Sample $v \in \mathcal{A}(D-d,\ell)$.
3) Sample $e \in \mathcal{R}^1(D-d,\ell)$.
4) Let $u = v \circ f_{i,j} + e$ and send $[u]$, $[v]$ to P_1, P_2.
5) Check if $v \circ [v]^{-1}(\tilde{y}) = u \circ [u]^{-1}(\tilde{x})$.

Figure 10: The noisy protocol.

Here are two observations on the noisy protocol. Firstly, we do not know if the attack is immediately resolved. So far we just intuitively believe this should work. In fact, if one shows that there is no attack for the noisy protocol, then he/she proves the soundness of the reduction, which is what we will do in the upcoming sections. Secondly, this test has completeness half. For any fixed $x \in \mathbb{F}_2^D$ and e sampled from $\mathcal{A}(D-d,\ell)$, $e(x) = 0^\ell$ with probability $1/2$. We have that for a valid assignment that satisfies $y_j = f_{ij}(x_i)$, the honest response $\tilde{x}_i = [u](x_i)$ and $\tilde{y}_j = [v](y_j)$ satisfy $v[v]^{-1}\tilde{y} = vy_j = u[u]^{-1}\tilde{x} = ux_i$ with probability $1/2$, thus giving us completeness $1/2$.

Now we will describe why we need to add the rank one noise by showing an attack on the noiseless protocol.

2.3 An attack to the noiseless protocol

The attack we are going to present is on a simple case of instances. Before we go into details, let's summarize a high-level message we want to emphasize here. Firstly, the noiseless protocol is vulnerable even on simple instances. This in some sense indicates the fundamental difficult in proving hardness of Unique-Games with almost complete completeness. Secondly, the power of noise. Though we haven't shown why the noisy protocol works, it is of interest from a high-level point of view why adding noise can make the reduction work.

Now, let's describe the attack. In the following ,we consider a Unique-Games instance with constraints of the form $I = \{y_j = x_i + \sigma_{i,j}\}$. Note that with these constraints, the queries $[u]$ and $[v]$ in the noiseless protocol are actually the same! The reason is that u is actually an affine shift of v and thus they are in the same equivalence class. Furthermore, we assume that I has the following property,

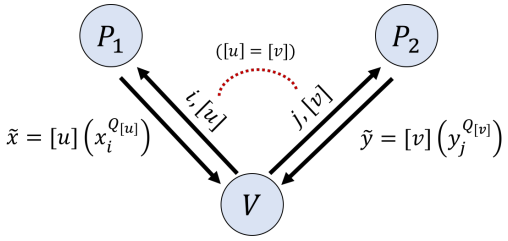
Assumption. *For any subspace $Q \subseteq \mathbb{F}_2^D$ of co-dimension at least $t = 100\log(1/\delta)$, the modified instance $I(Q) = \{\Pi_Q(y_j) = \Pi_Q(x_i) + \Pi_Q(\sigma_{i,j})\}$ is perfectly satisfiable, i.e.,* $\mathrm{val}(I(Q)) = 1$. ◊

That is, the assumption guarantees that for each subspace Q of co-dimension large enough, there exists $x_1^Q, x_2^Q, \ldots, x_n^Q \in \mathbb{F}_2^D$ and $y_1^Q, y_2^Q, \ldots, y_m^Q \in \mathbb{F}_2^D$ such that this is a satisfying assignment for $I(Q)$. One reason why such an assumption makes sense is that while the original constraints were unique, after projecting to Q of co-dimension t we get a 2^t-to-2^t constraints which are much easier to satisfy. So in some sense we replace

a single equation with the OR of 2^{2t} equations (for all possible projections) and so if t is sufficiently large as a function of the original soundness δ, we could expect that this relaxation will allow us to get a satisfying assignment.

To make good use of this property, we first note that the queries $[u]$ and $[v]$ are mapping vectors from high dimension ($\mathbb{F}_2^D$) to low dimension ($\mathbb{F}_2^\ell$). As a result, both of them must have a large *kernel*[12]. Also, the kernel of $[u]$ and $[v]$ must contain some Q that has the property stated in the assumption. Thus, a natural way to design an attack is then associating each $[u]$ and $[v]$ with some *canonical* subspaces $Q_{[u]}$ and $Q_{[v]}$. These canonical subspaces are predefined[13] and thus are known to both provers.

Now on query $(i, [u])$ and $(j, [v])$, the malicious provers work as follows. (i) Find $Q_{[u]}$ and $Q_{[v]}$ respectively. Note that as we discussed above, since $[u] = [v]$, we have $Q_{[u]} = Q_{[v]}$, *i.e.,* the two provers agree on the same subspace. (ii) Prover 1 returns $[u](x_i^{Q_{[u]}})$ and Prover 2 returns $[v](y_j^{Q_{[v]}})$. See Figure 11



1) Sample $i \sim j$.
2) Sample $v \in \mathcal{A}(D-d, \ell)$.
3) Let $u = v \circ f_{i,j}$ and send $[u]$, $[v]$ to P_1, P_2.
4) Check if $v \circ [v]^{-1}(\tilde{y}) = u \circ [u]^{-1}(\tilde{x})$.

Figure 11: Attack for noiseless protocol.

We claim that $v \circ [v]^{-1} \circ [v]((y_j^{Q_{[v]}})) = u \circ [u]^{-1} \circ [u]((x_i^{Q_{[u]}}))$ and thus the verifier will always accept. Consider the right-hand-side,

$$v \circ [v]^{-1} \circ [v]((y_j^{Q_{[v]}})) = v(y_j^{Q_{[v]}}) = v \circ \Pi_{Q_{[v]}}(y_j^{Q_{[v]}}).$$

Similarly,

$$u \circ [u]^{-1} \circ [u]((x_i^{Q_{[u]}})) = u \circ \Pi_{Q_{[u]}}(x_i^{Q_{[u]}}).$$

By the definition of $I(Q)$, we have

$$v \circ \Pi_{Q_{[v]}}(y_j^{Q_{[v]}}) = v \circ \Pi_{Q_{[v]}}(x_i^{Q_{[v]}} + \Pi_{Q_{[v]}}(\sigma_{i,j})).$$

Since $[u] = [v]$ and u is defined as $v \circ f_{i,j}$ in the noiseless protocol, we have

$$v \circ \Pi_{Q_{[v]}}(x_i^{Q_{[v]}} + \Pi_{Q_{[v]}}(\sigma_{i,j})) = u\Pi_{[u]}(x_i^{Q_{[u]}} + \Pi_{Q_{[u]}}(\sigma_{i,j})).$$

To sum up, we have $v \circ [v]^{-1} \circ [v]((y_j^{Q_{[v]}})) = u \circ [u]^{-1} \circ [u]((x_i^{Q_{[u]}}))$ and thus the verifier will always accept.

We conclude this section with some remarks about the attack. Note that here the attack really needs the two queries $[u], [v]$ to be affine shifts of each other. When we add noise to the queries, it becomes unclear whether this attack will still work because this relation between $[u]$ and $[v]$ is broken.

[12] Defined as $N([u]) = \{x \in \mathbb{F}_2^D : \ [u](x) = 0\}$. Also known as null space.

[13] One can define it to be the smallest subspace of co-dimension t that is contained in the the kernel of $[u]$ under lexicographical order.

S So far, we are on half way to our final goal. Please make sure you are comfortable with the reduction and take a deep breath before we go into the soundness analysis.

3 Soundness analysis of reducing from Unique Games

In this section, we are going to do the soundness analysis. To get the main ideas across, we first consider reducing from Unique Games instead of Smooth Label Cover. That is, $d = 0$ both $x_i, y_j \in \mathbb{F}_2^D$, and after the reduction, the alphabet size will go down to $\mathbb{F}_2^\ell$. This is just an alphabet reduction, where each symbol is replaced by a cloud indexed by matrices in $\mathcal{A}(D, \ell)$.

As our main lemma, we would like to prove that, if the reduced game is not sound then in fact the original game was not sound. Formally stated,

Lemma 3.1. *For any $\epsilon > 0$, there exists ℓ, D, δ. Let I be a Unique Games instance with constraints of the form $\{y_j = f_{i,j}(x_i)\}$ where $x_i, y_j \in \mathbb{F}_2^D$ and $f_{i,j}$'s are affine. Suppose there exists an assignment $F_1, \ldots, F_n$ to the x_i's and $G_1, \ldots, G_m$ to the y_j's where $F_i, G_j : \mathcal{A}(D, \ell) \to \mathbb{F}_2^\ell$ for each $i \in [n], j \in [m]$. Let u, v, e be sampled according to the protocol $\mathcal{P}_{i,j}$, that is, $v \sim \mathcal{A}(D, \ell), e \in \mathcal{R}^1(D, \ell), u = v \circ f_{i,j} + e$. Then if,*

$$\Pr_{\substack{i\sim j \\ (u,v,e)\sim\mathcal{P}_{i,j}}} [G_j(v) = F_i(u)] \geq \epsilon, \tag{3.2}$$

Then there exists $x_1, \ldots, x_n, y_1, \ldots, y_m \in \mathbb{F}_2^D$ such that

$$\Pr_{i\sim j}[y_j = f_{i,j}(x_i)] \geq \delta(\epsilon). \tag{3.3}$$

Basically, the soundness lemma says that once there's assignment (F, G) satisfying ϵ fraction of the tests, then there exists solution $\{x, y\}$ to the unique games instance satisfying δ fraction of the constraints where δ only depend on ϵ(not on D!). See subsection 4.1 for more discussion on the choice of parameters.

N The analysis involves many averaging arguments and we will omit the blow-up in probabilities as long as it does not depend on D.

For simplicity of notation, instead of thinking of our graph as bipartite with F and G nodes, we will think of it as a constraint graph over the F nodes only since it is a unique game anyway and the encodings of x_i, y_j are the same.

It is difficult to come up with an assignment $\{x_i\}$ to the original label cover problem directly from the F_i's. So instead of coming up with one assignment, we output a "short" list of possible assignments! This is done via *list decoding*.

Lemma 3.4. *To show Eq.(3.3), it suffices to find lists $L_1, L_2, \ldots, L_n$ where $L_i \subseteq \mathbb{F}_2^D$ and $|L_i| \leq s(\epsilon, \ell)$ for each $i \in [n], j \in [n]$, such that*

$$\Pr_{i\sim j} [\exists y \in L_j,\ \exists x \in L_i,\ \text{s.t. } y = f_{i,j}(x)] \geq \delta'(\epsilon, \ell). \tag{3.5}$$

Here s, δ' are some functions in ϵ and ℓ.

A simple averaging argument shows that Lemma 3.4 implies Lemma Lemma 3.1.

> **S** It is a good time for you to stop here and make sure what our goal (reducing from what to what) is, what simplification we are using, and why this makes sense.

3.1 List decoding an assignment

Our goal from now onwards is to produce lists $\{L_i\}$ given assignments $\{F_i\}$. We now define an important graph which gives us a graph view of the verifier test.

Definition 3.6 (Short code graph)**.** *The short code graph $\mathcal{G}$ is the graph with vertices $M \in \mathcal{A}(D,\ell)$ and an edge between M_1, M_2 if $M_2 = M_1 + e$, for some rank one matrix $e \in \mathcal{R}^1(D,\ell)$. An assignment F_i, assigns an ℓ-bit string from $\mathbb{F}_2^\ell$ to every vertex in this graph and we call the corresponding instance $\mathcal{G}_i$.* ◊

Note that since we consider the folded code, our assignment F_i is such that $F_i(g \circ v) = g \circ F_i(v)$ for all invertible $g \in \mathcal{A}(\ell,\ell)$, and our graphs $\mathcal{G}_i$ will necessarily have these consistencies too.

> **N** Although we use a folded code, our graph contains all affine functions $\mathcal{A}(D,\ell)$.

Make sure you understand the definition of the short code graph and do the following exercise.

> **E** Prove that for any assignment F_i, all typical sets of F_i are of the same size. *Hint: Use the fact that F_i is a folded assignment.*

Here is a graph view of the verifier test.

> **V** Given an assignment $\{\mathcal{G}_k\}$, the verifier test corresponds to picking a random constraint $i \sim j$, a random edge $(v, v+e) \in \mathcal{G}$, and checking whether $\mathcal{G}_j(v) = \mathcal{G}_i((v+e) \circ f_{i,j})$.

Therefore the soundness analysis relies heavily on the properties of the base graph $\mathcal{G}$. The [KMS18] manuscript proved a combinatorial hypothesis about the expansion properties of the short code graph which completed the proof of the 2-1 games conjecture.

For each $\mathcal{G}_i$, we can *partition* the vertices into $|\mathbb{F}_2^\ell| = 2^\ell$ sets by the value F_i assigns to the vertices $v \in \mathcal{A}(D,\ell)$. We will refer to these sets as *typical sets*.

Furthermore, from Equation 3.2, we have the following lemma. Also, see Figure 12.

Lemma 3.7 (Typical sets are non-expanding)**.** *Suppose there exists an assignment $F_1, F_2, \ldots, F_n$ where $F_i : \mathcal{A}(D,\ell) \to \mathbb{F}_2^\ell$ satisfying*

$$\Pr_{\substack{i\sim j \\ v\sim\mathcal{A}(D,\ell) \\ e\in\mathcal{R}^1(D,\ell) \\ u=v\circ f_{i,j}+e}} [F_j(v) = F_i(u)] \geq \epsilon. \tag{3.8}$$

Then with probability at least $\epsilon/2$ over the choice of constraints $i \sim j$, for all typical sets $F_i^{-1}(\alpha), \alpha \in \mathbb{F}_2^\ell$, of F_i and all typical sets $F_j^{-1}(\alpha)$ of F_j, we have that,

$$\Pr_{\substack{v \sim F_i^{-1}(\alpha) \\ e \in \mathcal{R}^1(D,\ell)}} [F_i(v) = F_i(v+e) = \alpha], \geq \epsilon', \tag{3.9}$$

and

$$\Pr_{\substack{v \sim F_j^{-1}(\alpha) \\ e \in \mathcal{R}^1(D,\ell)}} [F_j(v) = F_j(v+e) = \alpha] \geq \epsilon', \tag{3.10}$$

where $\epsilon' = \Omega(\epsilon^3)$.

Proof. The idea is based on some averaging arguments and the expansion properties of the Cayley graph $\mathcal{G}$. Details can be found in subsection C.2. □

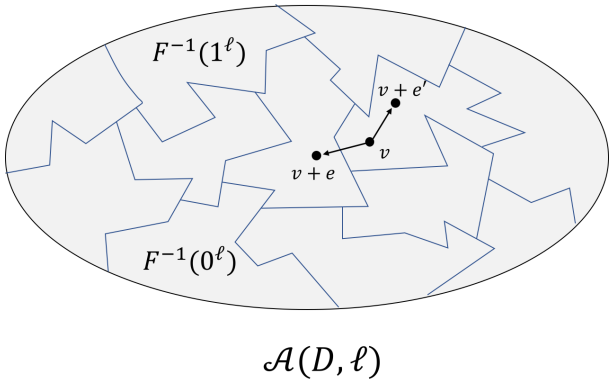


Figure 12: The partition of assignment F_i on $\mathcal{A}(D, \ell)$. Note that for elements in the same block, they have the same value of F_i. The typical set is non-expanding lemma tells us that a large fraction of edges e starting at a particular typical set, remain inside the typical set.

> **N** In the following, when say a set S in $\mathcal{A}(D, \ell)$ is non-expanding with respect to F_i, it means that $\Pr_{\substack{v \in S \\ e \in \mathcal{R}^1(D,\ell)}} [F_i(v) = F_i(v+e)]$ is not vanishing in n.

For a fixed i and a fixed typical set S inside $\mathcal{G}_i$, the probability that a random vertex $v \sim S$ and random edge e incident on v, stays inside S is equal to the probability over v, e that $F_i(v+e) = F_i(v)$. Since all typical sets are of the same size, Equation 3.2 is the average of this probability over all typical sets and it tells us that under a largely consistent assignment F_i, a random typical set is non-expanding.

If the short code graph $\mathcal{G}$ was a small set expander, previous PCP techniques, for example [SW06] tell us that our prover-verifier game with the reduced alphabet would be sound. Intuitively this is because a test that passes with high probability implies that there is some set S that is (i) non-expanding (as in Lemma 3.7) and (ii) induces a consistent assignment. By the small-set-expanding property, this means that this set S cannot be small, since all small sets expand. Now, we have that S is large and induces a consistent assignment. This means that our assignment is close to a valid codeword. As a valid codeword correspond to an assignment for the original instance, this implies that whenever our game is not sound, even the original

game was not sound and taking the contrapositive, we get a relation between the soundness of the new game with respect to the original game.

Unfortunately our graph is not a small set expander. But the good news is that we can "characterize" the non-expanding subsets of $\mathcal{G}$ using the following structured sets.

Definition 3.11 (Big and Small sets)**.** *Let $a \in \mathbb{F}_2^D$ and $b \in \mathbb{F}_2^\ell$. Define the corresponding big and small sets corresponding to points a, b as,*

$$\mathrm{BIG}_{a,b} = \{v \in \mathcal{A}(D,\ell) : v(a) = b\}$$

and

$$\mathrm{SMALL}_{a,b} = \{v \in \mathcal{A}(D,\ell) : \ \forall c \in \mathbb{F}_2^D, \ v(\langle a, c\rangle a) = v(c) = b\}.$$

We can generalize the above as follows. Let $Q \subseteq \mathbb{F}_2^D$ and $W \subseteq \mathbb{F}_2^\ell$ to be affine subspaces of dimension r. That is $Q = span(\{q_1, \ldots, q_r\}) + q_0$ and $W = span(\{w_1, \ldots, w_r\}) + w_0$ for some vectors $q_i \in \mathbb{F}_2^D$ and $w_i \in \mathbb{F}_2^\ell$. Define the corresponding r-Big set and r-Small set as follows.

$$\mathrm{BIG}_{Q,W} = \mathrm{BIG}_{q_0,w_0} \cap \mathrm{BIG}_{q_1+q_0,w_1+w_0} \cap \ldots \cap \mathrm{BIG}_{q_r+q_0,w_r+w_0}$$

Or equivalently,

$$\mathrm{BIG}_{Q,W} = \{v \in \mathcal{A}(D,\ell) : \ \Pi_W \circ v \circ \Pi_Q = v \circ \Pi_Q\},$$

and

$$\mathrm{SMALL}_{Q,W} = \{v \in \mathcal{A}(D,\ell) : \ \Pi_W \circ v \circ \Pi_Q = \Pi_W \circ v\},$$

where $\Pi_Q : \mathbb{F}_2^D \to \mathbb{F}_2^D$ and $\Pi_W : \mathbb{F}_2^\ell \to \mathbb{F}_2^\ell$ are the projections to subspaces Q and W respectively. Here $\circ$ denotes the composition of functions. ◊

> **N** Some remarks about the definition. Boaz used the notation $\mathbf{1}_Q$, $\mathbf{1}_Q$, and matrix multiplication, instead of Π_Q, Π_W, and $\circ$. Also in the original paper [DKK+16], they use zoom-ins and zoom-outs for small sets and big sets respectively.

The following proposition states some properties of the Big and Small sets. The proof will be postponed to later section.

Proposition 3.12. *Let $Q \subseteq \mathbb{F}_2^D$ and $W \subseteq \mathbb{F}_2^\ell$ be any subspaces of dimension r.*

1. *(geometric interpretation)*
   - *For any $v \in \mathrm{BIG}_{Q,W}$, we have $v(Q) \subseteq W$ where $v(Q) = \{v(q) : \ \forall q \in Q\}$.*
   - *For any $v \in \mathrm{SMALL}_{Q,W}$, we have $Q^\perp \subseteq ker(\Pi_W \circ v)$.*
2. *(non-expanding) We have*

$$\Pr_{\substack{v \sim \mathrm{BIG}_{Q,W} \\ e \sim \mathcal{R}^1(D,\ell)}}[v + e \in \mathrm{BIG}_{Q,W}] \geq 2^{-r},$$

$$\Pr_{\substack{v \sim \mathrm{SMALL}_{Q,W} \\ e \sim \mathcal{R}^1(D,\ell)}}[v + e \in \mathrm{SMALL}_{Q,W}] \geq 2^{-r}.$$

3. *(low density) We have*

$$\mu(\mathrm{BIG}_{Q,W}) \approx 2^{-\ell r}, \mu(\mathrm{SMALL}_{Q,W}) \approx 2^{-Dr},$$

where $\mu(S) = |S|/|\mathcal{A}(D,\ell)|$ for any subset $S \subseteq \mathcal{A}(D,\ell)$.

The proof of Proposition 3.12 is pretty straightforward but a little lengthy so we include it in subsection C.1 for completeness. Let us interpret the proposition. The first item tells us about the structure of $\text{BIG}_{Q,W}$ and $\text{SMALL}_{Q,W}$, which will help us to bound the size of the lists $\{L_i\}$ we construct. The second and third items together tell us that $\text{BIG}_{Q,W}$ and $\text{SMALL}_{Q,W}$ are *small non-expanding* sets, and we will use them to characterize typical sets which are also non-expanding.

3.2 What is the list?

Now that we have all the notation in place, let us define the list L_i of assignments we obtain from $\mathcal{G}_i$. Recall that this list would be a set of assignments for the original Label-Cover game (or Unique-Games). For now, let us forget about Small sets and only worry about Big sets. Observe that each big set $\text{BIG}_{Q,W}$ for some $Q \subseteq \mathbb{F}_2^D$ corresponds to a list of assignments Q.

First, we need to capture what kind of Big sets we are interested in. Intuitively, for an assignment F_i, we want a Big set B that has a large fraction of it having the same value under F_i. We call such Big set a *monochromatic set*.

Definition 3.13 (monochromatic set)**.** *Let $\tau \in [0,1]$, $F : \mathcal{A}(D,\ell)$ be an assignment and B be a Big set. We say B is a τ-monochromatic set with respect to F if there exists $\alpha \in \mathbb{F}_2^\ell$ such that*

$$|F^{-1}(\alpha) \cap B| \geq \tau |B|.$$

We call τ the goodness of B. ◊

Using this property, we define the list L_i as follows. First, choose some parameters (which will be set in the near future), let $\tau \in [0,1]$ be the goodness of Big sets and r be the dimension of Big sets. We define,

$$L_i = \{Q : \text{BIG}_{Q,W} \text{ is an } r\text{-}\textit{Big set} \text{ and } \text{BIG}_{Q,W} \text{ is } \tau(r)\text{-monochromatic.}\}$$

> **E** Verify that if F_i is a valid assignment, that is, if there exists $x \in \mathbb{F}_2^D$ such that $F_i(v) = v(x)$ for any $v \in \mathcal{A}(D,\ell)$, then $L_i = \{x\}$.

But this list as defined might be very long, possibly of size 2^{rD}. For this version of the notes though, we do not prove that the list size is bounded by a function of ℓ, ϵ. We conjecture that a pruning of this list by a careful procedure will give us a short list. We will refer to the pruned list as, $\text{Pruned}(L_i)$, i.e.

$$\text{Pruned}(L_i) = \text{Pruned}(\{Q : \text{BIG}_{Q,W} \text{ is an } r\text{-}\textit{Big set} \text{ and } \text{BIG}_{Q,W} \text{ is } \tau(r)\text{-monochromatic.}\})$$

> **N** In this version of the notes, we do not know how to prove that the list as defined above is short although we prove its correctness. We believe that this list or some modification of it can be proved to be short using an argument similar to [DKK+16].

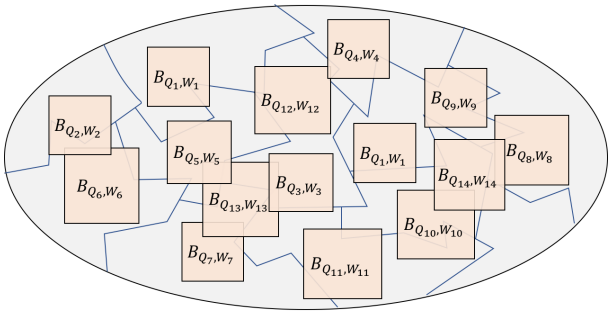


Figure 13: The Pruned list corresponds to a set of maximal Big sets which are monochromatic.

Now that we have a list-decoding, we need to prove two properties to complete the soundness analysis:

- List size bounded: Each list Pruned(L_i)is of size at most a constant which only depends on ϵ, τ, r and not on D.
- Correctness: The lists satisfy Equation 3.5 which says that for most constraints (i, j), there is some assignment $x \in L_i$ and $y \in L_j$ such that $y = f_{i,j}x$.

Item 1 combined with Item 2 imply that the original label cover game was sound.

> (S) Please make sure you understand the construction of lists and why the above two items suffice for the soundness theorem before moving on.

3.2.1 Bounding the list size

Conjecture 3.14. *There is an algorithm which prunes the lists $\{L_i\}$ for maximality, which ensures that the final list size is bounded, i.e.*

$$|\{Pruned(L_i)\}| \leq f(\epsilon, \tau, r, \ell).$$

Recall that a list L_i is a set of all r-BIG sets which are $\tau(r)$ monochromatic. This may possibly be a list of size 2^{rD} whereas we want the size to only depend on constants r, τ, ϵ, ℓ. The structure of the BIG sets will possibly allow us to prune the list and get Pruned(L_i) while retaining its properties.

Here is a vague outline of the proof. In Proposition 3.12 we proved that the BIG sets are non-expanding and of low density. Using this, prove that, if we have m BIG sets, then we can replace it by a single BIG set which contains all these sets and additionally, is also monochromatic. We continue this pruning until we are left with a list of small size. This argument is based on the [DKK+16] paper and the "sunflower argument" therein. A formal proof will be presented in a later version of the notes.

3.2.2 Proof that this list works

We will now assume that our lists $\{\text{Pruned}(L_i)\}$ have been generated from an assignment $\{F_i\}$ which has soundness $\geq \epsilon$, or equivalently,

$$\Pr_{\substack{i \sim j \\ (v,e)}} [F_j(v) = F_i\Big((v+e) \circ f_{i,j}\Big)] \geq \epsilon. \tag{3.15}$$

Using this we will prove that the original unique game was sound. Firstly, notice that an averaging argument over pairs $i \sim j$ gives us the following lemma.

Lemma 3.16. *Let $\{F_i\}$ be an assignment that has soundness at least $\epsilon > 0$. We have that, with probability $\epsilon/2$ over the choice of pairs $i \sim j$,*

$$\Pr_{(v,e)}[F_j(v) = F_i\Big((v+e) \circ f_{i,j}\Big) \geq \epsilon/2] \tag{3.17}$$

We will show that whenever a pair $i \sim j$ satisfy, condition Equation 3.17, we get that the lists $\text{Pruned}(L_i)$ and $\text{Pruned}(L_j)$ contain a common element. This proves the list-decoding theorem Lemma 3.4 which implies the soundness theorem.

From now on, we will fix such an $i \sim j$ pair, for which (3.17) holds. After fixing a pair i, j, we can permute the assignment F_i by $f_{i,j}$(replacing $F_i(v)$ by $F_i(v \circ f_{i,j})$), such that, now our verifier is checking equality between (the permuted assignment)F_i and F_j. That is, we have that,

$$\Pr_{(v,e)}[F_j(v) = F_i(v+e)] \geq \epsilon/2.$$

To do this, we first want to find a set T such that T is monochromatic both on assignments F_i and F_j. Using Lemma 3.7, we can prove that T is a non-expanding in the partition of F_i and F_j on the short code graph $\mathcal{G}$. The above intuition can be formulated into the following lemma.

Lemma 3.18. *Let $i \sim j$ be a pair satisfies Equation 3.17. Then, there exists $\alpha, \beta \in \mathbb{F}_2^\ell$ such that, the set $T = F_i^{-1}(\alpha) \cap F_j^{-1}(\beta)$ is non-expanding in $\mathcal{G}$.*

Proof. We are unsure of the details of this proof, but we will include it in a later version of the notes. The intuition is that this lemma should follow from the properties of Cayley graphs. □

Now comes the role of the combinatorial hypothesis [DKK+16] that was proved in [BKS18]. It tells us that every non-expanding set of $\mathcal{G}$ has a non-trivial intersection with Big set and/or Small set. Equivalently, since most typical sets are non-expanding, this says that we can cover the graph $\mathcal{G}_i$ with monochromatic Big sets and Small sets.

Theorem 3.19 ([KMS18])**.** *For any $\epsilon > 0$ there exists $\tau > 0$ and integer $r > 0$ such that for any $A \subseteq \mathcal{A}(D, \ell)$, if*

$$\mathop{\mathbf{P}}_{\substack{v \in A \\ e \in \mathcal{R}^1(D,\ell)}} [v + e \in A] \geq \epsilon.$$

Then, there exists r_1-Big set B and r_2-Small set S such that

$$|(B \cap S) \cap A| \geq \tau \cdot |B \cap S|, \tag{3.20}$$

where $r_1 + r_2 \leq r$.

The above lemma completes the soundness analysis and it's only 3 lines from here! For simplicity, here we assume a stronger version of Theorem 3.19 where $|(B \cap S) \cap A| \geq \delta \cdot |B|$ instead of $\geq |B \cap S|$.

> **N** Note that the assumption that $|(B \cap S) \cap A| \geq \delta \cdot |B|$ is not generally true. In fact, this corresponds to Boaz's example where monochromatic Big sets can cover the graph $\mathcal{G}_i$. In the next lecture, we will see how to handle the most general case where this is not necessarily true and small sets are also involved.

Lemma 3.21. *For any $\epsilon > 0$ there exists $r > 0$ such that the following holds: Let $i \sim j$ be a pair satisfying Equation 3.17, then there exists an r-Big set B such that $Q_B \in L_i, L_j$ and additionally Pruned$(L_i) \cap$ Pruned$(L_j) \neq \phi$.*

Proof Sketch. Let us first prove that there is a Big set B which is included in both lists L_i and L_j. We will look into the pruned case after that. Since T is a non-expanding set, theorem 3.19 gives us that there is some big set B such that $|B \cap T| \geq \tau|B|$. Recall that $T = F_i^{-1}(\alpha) \cap F_j^{-1}(\beta)$, which means that B is monochromatic with respect to both F_i and F_j. The way the lists L_i, L_j were constructed we get that $Q_B \in L_i, L_j$ and so $L_i \cap L_j \neq \phi$.

> **N** We haven't concretely defined a pruning procedure, but we conjecture that for some definition, the following holds.

In the case that we have pruned lists $\text{Pruned}(L_i), \text{Pruned}(L_j)$, we cannot say that the Big set B we get above has been included even after pruning. But by the properties that the pruned lists would satisfy, we can say that there exist big sets B_i and B_j where $Q_{B_i} \in \text{Pruned}(L_i), Q_{B_j} \in \text{Pruned}(L_j)$ such that $Q_B \subseteq Q_{B_i}, Q_{B_j}$, which implies that $Q_{B_i} \cap Q_{B_j} \neq \phi$. This means that the pruned lists have a common element, which is what we had set out to do. □

This lemma completes the soundness analysis assuming the case that Big sets are enough to cover the graph.

> **E** Convince yourself that this completes the analysis and proves Lemma 3.4.

4 Discussion

4.1 Relation among parameters

There are plenty of parameters show up in the reduction. In this subsection, we are going to discuss the relation among them.

$$\delta \Rightarrow r, \tau \Rightarrow \ell \Rightarrow |L| \Rightarrow \epsilon \Rightarrow d \Rightarrow \mu \Rightarrow D. \tag{4.1}$$

In the whole reduction, we first pick the soundness $\delta > 0$, and in the end we hope for some small D, which correspond to the *blowup* of the reduction. Specifically, it is of interest to explicitly upper bound $D(\delta)$. After we pick δ, the dimension and goodness of monochromatic set are then decided by Theorem 3.19. The dimension ℓ of the tiny alphabet set will also be determined as well as the size of the list. With these parameters, we can then upper bound the soundness ϵ of the Label-Cover we want to reduce from. Finally, the logarithmic alphabet ratio d and the smoothness μ are settled down and in the end we get to know how large D should be.

References

[ABS10] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 563–572. IEEE, https://www.dsteurer.org/paper/subexpug.pdf, 2010.

[BGH+15] Boaz Barak, Parikshit Gopalan, Johan Håstad, Raghu Meka, Prasad Raghavendra, and David Steurer. Making the long code shorter. *SIAM Journal on Computing*, 44(5):1287–1324, 2015.

[BKS18] Boaz Barak, Pravesh Kothari, and David Steurer. Small-set expansion in short-code graph and the 2-to-1 conjecture. In *Unpublished draft*, 2018.

[DKK+16] Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 23, page 198. https://eccc.weizmann.ac.il/report/2016/198/, 2016.

[DKK+17] Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in grassmann graphs. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 24, page 94. https://eccc.weizmann.ac.il/report/2017/094/, 2017.

[GW95] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[Hås01] Johan Håstad. Some optimal inapproximability results. In *Journal of the ACM (JACM)*, volume 48(4), pages 798–859. http://www.cs.umd.edu/~gasarch/BLOGPAPERS/max3satl.pdf, 2001.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[Kho02] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775. ACM, https://dl.acm.org/citation.cfm?id=510017, 2002.

[Kho10] Subhash Khot. Inapproximability of np-complete problems, discrete fourier analysis, and geometry. In *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pages 2676–2697. World Scientific, 2010.

[KMS17] Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and grassmann graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 576–589. ACM, 2017.

[KMS18] Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. In *Electronic Colloquium on Computational Complexity (ECCC)*. https://eccc.weizmann.ac.il/report/2018/006/, 2018.

[Ste10] David Steurer. Subexponential algorithms for d-to-1 two-prover games and for certifying almost perfect expansion. *http://www.dsteurer.org/paper/dgames/*, 1:2–1, 2010.

[SW06] Amir Shpilka and Avi Wigderson. Derandomizing homomorphism testing in general groups. *SIAM Journal on Computing*, 36(4):1215–1230, 2006.

A Boaz's dictionary

The high level points I'd like to both understand myself and convey to the audience are:

1. In what way is the DKKMS reduction similar to previous results like Hastad's 3XOR and others that used the long code or other gadgets on top of the label cover problem.

2. In what way is it different, and in particular why is the need for smoothness in the underlying label cover and hence a polynomial blowup.

3. What are the barriers to transforming this into a proof of the unique games conjecture - do we expect that there would simply be a better gadget?

I have some understanding of 1 and 2, not real understanding of 3 yet but some thoughts on it. One intuition I have is that previous reductions typically relied on an error correcting code with some local test, and had a theorem of the form

"If some string F (e.g., a "putative codeword" that we run the test on) passes the test with probability larger than some constant $\epsilon > 0$, then there is a constant sized list $F_1, ..., F_t$ of codewords such that a constant fraction of coordinates of F agrees with at least some F_i in the list"

Sometimes the list was not exactly codewords but some generalization of it. For example Hastad's test used the longcode where the codewords are dictatorships, and the theorem said that any function passing the test with constant probability must be correlated with some list of juntas.

I think DKKMS use a similar high level structure, but their list could be very large - of super constant size. What saves them is that the list is "structured" in some way (this is where their zoom in and zoom out comes into play).

I am going to use somewhat different language than they do. To translate between the two you can use the following "dictionary" (below I try to be as consistent as possible with the notation and variables used in DKKMS - in my talk I will probably use different notation and letters). This dictionary should also be part of the scribe notes.

A.1 Difference 1: subspaces vs affine functions/matrices

Think of $n \gg \ell$.

I will think of the code where we map a string g in $GF(2)^n$ into a function F that maps every affine function $v : GF(2)^n \to GF(2)^\ell$ into $v(g)$.

That is, the error correcting code I will talk about maps every string in $GF(2)^n$ into a string over alphabet $GF(2)^\ell$ of length roughly $2^{\ell n+\ell}$ with coordinate corresponding to every affine function from $GF(2)^n$ to $GF(2)^\ell$.

DKKMS consider the code where we map a string g in $GF(2)^n$ (which we think of also as a linear function from $GF(2)^n$ to $GF(2)$) into the function F that on input a subspace $L \subseteq GF(2)^n$ of dimension ℓ, $F(g)$ is the restriction of the function g to the subspace L. (Note that this restriction can be represented using ℓ bits).

So their code maps every string in $GF(2)^n$ into a string of alphabet $GF(2)^\ell$ of length roughly $2^{\ell(n-\ell)}$ (roughly the number of ℓ dimensional subspaces of $GF(2)^n$). Note that in both cases the dominant term is $2^{\ell n}$ so the codes are rather similar.

A.2 Difference 2: the test

I will think of the test where given some putative codeword F, we choose a random affine function $g : GF(2)^n \to GF(2)^\ell$ and a random rank one linear function $e : GF(2)^n \to GF(2)^n$ (that is a function of the form $e(x) = \langle x, u \rangle v$ where u, v are vectors in $GF(2)^n$), and check if $F(g) = F(g + e)$. This is a 1-to-1 test (since we check equality between two coordinates of the putative codeword) that a correct codeword passes with probability 1/2.

One can also think of the test where we check whether either $F(g) = F(g + e)$ or $F(g) = F(g + e) + v$ where v is the vector above. This would be a 2-to-1 test that a correct codeword passes with probability 1.

DKKMS think of the test (see Test 1 in their paper) where given a putative codeword F, they choose two ℓ dimensional subspaces L_1, L_2 whose intersection L' has $\ell - 1$ dimension and they accept if the restriction of $F(L_1)$ to L' is equal to the restriction of $F(L_1)$ to L'. This is a 2-to-2 test that a correct codeword passes with probability 1. (The difference between 2 to 2 and 2 to 1 is immaterial). They could also have defined a version of this test that would be 1-to-1 and will pass with probability 1/2.

So the tests are very similar as well.

A.3 Difference 3: the "Nice" sets.

I think of a "d level nice big set" parameterized by a d-tuple W of vectors in $GF(2)^n$ and a d-tuple W' of vectors in $GF(2)^\ell$ as the set $BIG_{W,W'}$ of all affine functions $g : GF(2)^n \to GF(2)^\ell$ such that $g(W_i) = W'_i$ for $i = 1, \ldots, d$. Note that each such set constitutes about a $2^{-d\ell}$ fraction of all functions.

I will think of a "d level nice small set" parameterized by a pair of affine functions $Q : GF(2)^n \to GF(2)^d$, and $Q' : GF(2)^\ell \to GF(2)^\ell$ as the set $SMALL_{Q,Q'}$ of all affine functions $g : GF(2)^n \to GF(2)^\ell$ such that the function $x \mapsto Q'(g(x))$ equals to Q. Note that each such set constitutes about a 2^{-dn} fraction of all functions.

DKKMS think of a "d zoom out" parameterized by a co-dimension d subspace W as the set of all subspaces L such that $L \subseteq W$. Note that this is about a $2^{-d\ell}$ fraction of subspaces.

They think of a "d zoom in" parameterized by a dimension d subspace Q as the set of all subspaces L such that $Q \subseteq L$. Note this is about a 2^{-dn} fraction of subspaces.

A.4 Difference 4: abstraction

Another difference between my presentation and the DKKMS is that I will try to abstract the particular construction label cover instance.

I will think of a label cover instance which contains constraints of the form $y_j = f_{i,j}(x_i)$ for affine functions $f_{i,j} : GF(2)^{2k} \to GF(2)^{2k-\beta k}$. (Think of $\beta = O(\log k/k)$ so $\beta k \sim \log k$.)

Another way to think of it is as a game where the verifier sends i to prover one and gets back $x_i \in GF(2)^{2k}$, j to prover two and gets back $y_j \in GF(2)^{2k-\beta k}$, and checks whether $y_j = f_{i,j}(x_i)$.

They use a particular construction of such an instance which is obtained by staring with an underlying 3XOR instance. The verifier picks k random equations $e_1, \ldots, e_k$ (very likely to touch $3k$ distinct variables) and sends them to the first prover, who sends back an assignment to the $3k$ variables that satisfies these. (This can be described as a $2k$ dimensional vector since that is the dimension of the affine subspace of satisfying assignments.) The verifier sends to the second prover $k - \beta k$ of the equations plus one variable out of each of the remaining variables. He gets back a satisfying assignment for the equations and an assignment for the isolated variables. This can be described in $2(k - \beta k) + \beta k = 2k - \beta k$ bits. The verifier checks that the two assignments are consistent, which is an affine check.

B Scriber's dictionary

In this notes, many of the presentations are also different from what Boaz and DKKMS used. Here we try to list all the difference and we hope to minimize the confusion of readers.

Notation

- DKKMS and us use $\mathbb{F}_2$ while Boaz uses $GF(2)$. The two are the same.
- We use Π_Q to denote the projection operator to subspace Π_Q while Boaz uses $\mathbf{1}_Q$.
- We use $\mathcal{A}(D, \ell)$ to denote the space of all affine functions from $\mathbb{F}_2^D$ to $\mathbb{F}_2^\ell$ while Boaz does not explicitly give this set a name.
- We use ϵ, δ to denote the soundness of the original game (smooth Label-Cover) and the reduced game (Unique-Games) respectively while Boaz uses in a reverse way.
- Most of the time we use function instead of matrix. As a result, the operations are denoted in *function composition* $\circ$ while Boaz uses matrix multiplication most of the time.
- The way we doing list-decoding is different from that of Boaz. We directly pick all the $\tau(r)$-monochromatic set while Boaz carefully makes sure all the chosen $\tau(r)$-monochromatic sets are disjoint.

Folding and randomized query In the protocol we used in this notes, see Figure 10, we adopt the *folding* notation. In Boaz's presentation, sometimes he used a random query instead. See Figure 14.

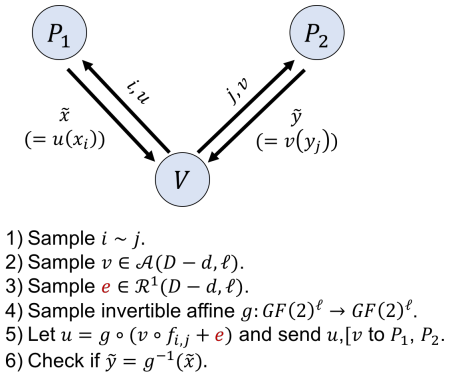


Figure 14: The noisy protocol using randomized query instead of folding.

It is a good exercise to see why the two are equivalent up to some constant factor.

> **E** Show that the noisy protocol with folding is sound[a] if and only if the noisy protocol with randomized query is sound. Hint: Averaging argument.
>
> [a] That is, the soundness is independent to n.

C Some missing proofs

C.1 Proof of the properties of Big and Small sets

Proof of Proposition 3.12.

1. This is left as an exercise for the reader.

2. Fix $v \in \mathrm{BIG}(D, \ell)$. Sampling e from $\mathcal{R}^1(D, \ell)$, is equivalent to sampling a and b uniformly at random in $\mathbb{F}_2^\ell$ and $\mathbb{F}_2^D$ respectively and setting $e' = ab^\top$[14]. Let $\{q_1, q_2, \ldots, q_r\}$ be a basis of Q, we have

$$\begin{aligned}
\Pr_{e \sim \mathcal{R}^1(D,\ell)}[v + e \in \mathrm{BIG}_{Q,W}] &= \Pr_{e \sim \mathcal{R}^1(D,\ell)}[(v+e)(Q) \subseteq W] \\
&\geq \Pr_{e \sim \mathcal{R}^1(D,\ell)}[e(Q) = \mathbf{0}] \\
&= \Pr_{a \sim \mathbb{F}_2^\ell,\ b \sim \mathbb{F}_2^D}[ab^\top q_i = \mathbf{0},\ \forall i \in [r]] \\
&= 2^{-r},
\end{aligned}$$

[14] Note that with this sampling there's a small probability $(2^{-(D+\ell)})$ of e' being a zero matrix, which has rank zero and thus $e \notin \mathcal{R}^1(D, \ell)$. However, for the simplicity of analysis, we use this sampling and omit the probability of degenerating.

where the second step follows from linearity of the functions and item 1. Similarly,

$$\begin{aligned}\Pr_{e\sim\mathcal{R}^1(D,\ell)}[v+e\in \text{SMALL}_{Q,W}] &= \Pr_{e\sim\mathcal{R}^1(D,\ell)}[Q^\perp \subseteq ker(\Pi_W\circ(v+e))]\\ &\geq \Pr_{e\sim\mathcal{R}^1(D,\ell)}[e(Q)=\mathbf{0}]\\ &= 2^{-r}.\end{aligned}$$

3. Note that $\mu(\text{BIG}_{Q,W}) = \mathbf{P}_{v\in\mathcal{A}(D,\ell)}[v\in\text{BIG}_{Q,W}]$. Recall that from item 1, $v\in\text{BIG}_{Q,W}$ if $v(Q)\subseteq W$. Let $\{q_1,\ldots,q_r\}$ be a basis for Q. Observe that for any $i\in[r]$

$$\Pr_{v\sim\mathcal{A}(D,\ell)}[v(q_i)\subseteq W] = \frac{|W|}{|\mathbb{F}_2^\ell|} = 2^{r-\ell}.$$

Also, since $\{q_i\}$ is a basis, the probabilities for distinct q_i, q_j are independent. That is,

$$\Pr_{v\sim\mathcal{A}(D,\ell)}[\forall i\in[r],\ v(q_i)\subseteq W] = 2^{r(r-\ell)}.$$

Finally, note that $v(Q)\subseteq W$ if and only if $v(q_i)\in W$ for all $i\in[r]$. As a result,

$$\mu(\text{BIG}_{Q,W}) = \Pr_{v\sim\mathcal{A}(D,\ell)}[v\in\text{BIG}_{Q,W}] = 2^{r^2-r\ell}\approx 2^{-r\ell},$$

where we think of r being much smaller than ℓ.

The density of $\text{SMALL}_{Q,W}$ can be approximated in a similar way by using a basis for $Q^\perp$.

□

C.2 Proof of typical sets are non-expanding lemma

Proof of Lemma 3.7. Let us restate Lemma 3.7 here.

Lemma C.1. *Suppose there exists an assignment $F_1, F_2, \ldots, F_n$ where $F_i : \mathcal{A}(D,\ell)\to\mathbb{F}_2^\ell$ satisfying*

$$\Pr_{\substack{i\sim j\\ v\sim\mathcal{A}(D,\ell)\\ e\in\mathcal{R}^1(D,\ell)\\ u=v\circ f_{i,j}+e}}[F_j(v)=F_i(u)]\geq\epsilon. \tag{C.2}$$

Then with probability at least $\epsilon/2$ over the choice of constraints $i\sim j$, for all typical sets $F_i^{-1}(\alpha)$ of F_i, where $\alpha\in\mathbb{F}_2^\ell$, and all typical sets $F_j^{-1}(\alpha)$ of F_j, we have that,

$$\Pr_{\substack{v\sim F_i^{-1}(\alpha)\\ e\in\mathcal{R}^1(D,\ell)}}[F_i(v)=F_i(v+e)=\alpha]\geq\epsilon', \tag{C.3}$$

and

$$\Pr_{\substack{v\sim F_j^{-1}(\alpha)\\ e\in\mathcal{R}^1(D,\ell)}}[F_j(v)=F_j(v+e)=\alpha]\geq\epsilon', \tag{C.4}$$

where $\epsilon'=\Omega(\epsilon^3)$.

We will first prove that,

$$\Pr_{\substack{v\sim\mathcal{A}(D,\ell)\\ e\in\mathcal{R}^1(D,\ell)}}[F_i(v)=F_i(v+e)]\geq\epsilon'. \tag{C.5}$$

Note that here the probability is over all vertices $v \in \mathcal{A}(D,\ell)$. It is enough to prove this because by the symmetry of the short code graph this implies that, for all typical sets,

$$\Pr_{\substack{v\sim F_i^{-1}(\alpha)\\ e\in\mathcal{R}^1(D,\ell)}}[F_i(v) = F_i(v+e) = \alpha] \geq \epsilon'. \tag{C.6}$$

Let us start with some notations. In this proof, we use the matrix representation of $\mathcal{A}(D,\ell)$. That is, each element is represented by a $\ell \times D$ boolean matrix. The test in the protocol becomes checking the consistency between a matrix A and A plus a rank one matrix. Formally, we can define the underlying testing graph as $G_{D,\ell} = (\mathbb{F}_2^{\ell\times D}, E_{D,\ell})$ where

$$E_{D,\ell} = \{(A,B) :\ A,B \in \mathbb{F}_2^{\ell\times D},\ A - B = ab^T,\ a \in \mathbb{F}_2^{\ell},\ b \in \mathbb{F}_2^{D}\}.$$

Now, denote the *normalized* adjacency matrix of $G_{D,\ell}$ as $M_{D,\ell}$. That is, the ℓ_1 norm of each column of $M_{D,\ell}$ is 1. Then $M_{D,\ell}$ captures the random walk on $G_{D,\ell}$. Specifically, for any $S \subseteq \mathbb{F}_2^{\ell\times D}$, let $\mathbf{1}_S$ be the indicator of S. $\mathbf{1}_S^\top M_{D,\ell}\mathbf{1}_S/|S|$ is the probability of beginning in S and staying in S after one step. That is, $\mathbf{1}_S^\top M_{D,\ell}\mathbf{1}_S/|S|$ refers to the non-expandingness of S.

> Ⓢ Please make sure you understand the matrix representation of $\mathcal{A}(D,\ell)$, the testing graph $G_{D,\ell}$, and how these connect to the soundness lemma and the non-expanding property.

Now, let us go back to the starting point of the soundness lemma. Consider a typical set defined by $S_{i,\alpha} = \{v \in \mathbb{F}_2^{\ell\times D} :\ F_i(v) = \alpha\}$. Equation C.2 tells us that $S_{i,\alpha}$ is non-expanding with respect to $M_{D,\ell}^2$. Concretely, we have the following lemma.

Lemma C.7 (typical set is non-expanding w.r.t. two-step random walk)**.** *Given Equation C.2, with probability at least $\epsilon/2$ over the choice of constraints (i,j),*

$$\Pr_{\substack{v\sim\mathcal{A}(D,\ell)\\ e,e'\in\mathcal{R}^1(D,\ell)}}[F_i(v) = F_i(v+e+e')] \geq \epsilon^3/64,$$

Proof of Lemma C.7. In this proof, we think of $f_{i,j}$ as identity function for simplicity. First, by averaging argument, with probability at least $\epsilon/2$ over the choice of pair $(i,j) \in [n]\times[n]$,

$$\Pr_{\substack{v\sim\mathcal{A}(D,\ell)\\ e\in\mathcal{R}^1(D,\ell)}}[F_j(v) = F_i(v+e)] \geq \epsilon/2.$$

We call an element $v \in \mathcal{A}(D,\ell)$ *good* if $F_j(v) = F_i(v+e)$ with probability at least $\epsilon/4$ over the choice of e. Formally, define

$$GOOD_{i,j} = \{v :\ \Pr_{e\in\mathcal{R}^1(D,\ell)}[F_j(v) = F_i(v+e)] \geq \epsilon/4\}.$$

By averaging argument, we have $\Pr_{v\in\mathcal{A}(D,\ell)}[v \in GOOD_{i,j}] \geq \epsilon/4$. Now, let us look at the two-step non-expanding probability as follows.

Claim C.8.

$$\Pr_{\substack{v\in\mathcal{A}(D,\ell)\\ e,e'\in\mathcal{R}^1(D,\ell)}}[F_i(v) = F_i(v+e+e')] \geq \frac{\epsilon^3}{64}.$$

Proof of Claim C.8.

$$\begin{aligned}
&\Pr_{\substack{v\in\mathcal{A}(D,\ell)\\ e,e'\in\mathcal{R}^1(D,\ell)}}[F_i(v)=F_i(v+e+e')]\\
\geq&\Pr_{\substack{v\in\mathcal{A}(D,\ell)\\ e,e'\in\mathcal{R}^1(D,\ell)}}[F_i(v)=F_j(v+e)=F_i(v+e+e')]\\
\geq&\Pr_{v\in\mathcal{A}(D,\ell)}[v\in GOOD_{i,j}]\cdot\Pr_{\substack{v\in GOOD_{i,j}\\ e,e'\in\mathcal{R}^1(D,\ell)}}[F_i(v)=F_j(v+e)=F_i(v+e+e')] &\text{(C.9)}\\
\geq&\frac{\epsilon}{4}\cdot\Pr_{\substack{v\in GOOD_{i,j}\\ e,e'\in\mathcal{R}^1(D,\ell)}}[F_j(v)=F_i(v+e)\text{ and }F_j(v)=F_i(v+e')] &\text{(C.10)}\\
\geq&\frac{\epsilon}{4}\cdot\sum_{v\in GOOD_{i,j}}\Pr_{e,e'\in\mathcal{R}^1(D,\ell)}[F_j(v)=F_i(v+e)\text{ and }F_j(v)=F_i(v+e')] &\text{(C.11)}\\
\geq&\frac{\epsilon}{4}\cdot\sum_{v\in GOOD_{i,j}}\Pr_{e\in\mathcal{R}^1(D,\ell)}[F_j(v)=F_i(v+e)]\cdot\Pr_{e'\in\mathcal{R}^1(D,\ell)}[F_j(v)=F_i(v+e')] &\text{(C.12)}\\
\geq&\frac{\epsilon^3}{64}. &\text{(C.13)}
\end{aligned}$$

□

Equation C.9 conditions on the good elements in $\mathcal{A}(D,\ell)$. Equation C.10 relabel $(v, v+e, v+e+e')$ to $(v+e, v, v+e'')$. Note that the distributions of the two labeling are the same. Equation C.11 expand the probability of good event so that in the next step we can separate the probability by the independence of e and e'. Equation C.12 is due to the independence of e and e'' and Equation C.13 is due to the definition of $GOOD_{i,j}$. □

Equivalently, the result of Lemma C.7 can be written as the matrix form $\mathbf{1}^\top M_{D,\ell}^2\mathbf{1}/|\mathcal{A}(D,\ell)|\geq\epsilon'$. Recall that our goal is to show that $S_{i,\alpha}$ is non-expanding with respect to one step of random walk. That is, we hope for $\mathbf{1}^\top M_{D,\ell}\mathbf{1}/|\mathcal{A}(D,\ell)|\geq\epsilon''$ for some constant ϵ''. In the following, we achieve this goal by showing $M_{D,\ell}$ is *positive semidefinite*. Before we prove that $M_{D,\ell}$ is positive semidefinite, let us first see why this implies the our goal.

Note that the absolute value of the eigenvalues of $M_{D,\ell}$ is at most 1 due to the normalization. If $M_{\ell,D}$ is positive semidefinite, it immediately implies that $x^\top M_{D,\ell}x\geq x^\top M_{D,\ell}^2x$ for any vector x. That is,

$$\Pr_{\substack{v\in\mathcal{A}(D,\ell)\\ e\in\mathcal{R}^1(D,\ell)}}[F_i(v)=F_i(v+e)]=\frac{\mathbf{1}^\top M_{D,\ell}\mathbf{1}}{|\mathcal{A}(D,\ell)|}\geq\frac{\mathbf{1}^\top M_{D,\ell}^2\mathbf{1}}{|\mathcal{A}(D,\ell)|}\geq\epsilon'.$$

$M_{D,\ell}$ is positive semidefinite To show that $M_{D,\ell}$ is positive semidefinite, it suffices to show that the adjacency matrix of $G_{D,\ell}$ is positive semidefinite. Observe that $G_{D,\ell}$ is a *Cayley graph* on group $\mathbb{F}_2^{\ell\times D}$ with generator $S=\{ab^\top:\ a\in\mathbb{F}_2^\ell,\ b\in\mathbb{F}_2^D\}$. As a result, the eigenvalues of the adjacency matrix of $G_{D,\ell}$ can be characterized by the *characters* of $\mathbb{F}_2^{\ell\times D}$.

The characters of a group G is a mapping χ from G to $\mathbb{C}\backslash\{0\}$ satisfying the property $\chi(g\cdot g')=\chi(g)\cdot\chi(g')$ for any $g,g'\in G$. It can be easily shown that the characters of the group $\mathbb{F}_2^{\ell\times D}$ is of the form $\chi_H(A)=(-1)^{tr(H^\top A)}$ where $H\in\mathbb{F}_2^{\ell\times D}$. It is also well-known that the eigenvalues of the adjacency matrix of the Cayley graph is of the following form. For any $H\in\mathbb{F}_2^{\ell\times D}$,

$$\lambda(H)=\frac{1}{|S|}\sum_{A\in S}\chi_H(A)=\frac{1}{|S|}\sum_{a\in\mathbb{F}_2^\ell}\sum_{b\in\mathbb{F}_2^D}\chi_H(ab^\top).$$

Let us rewrite the summands and have

$$\chi_H(ab^\top) = (-1)^{tr(H^\top ab^\top)} = (-1)^{a^\top Hb}.$$

When we fix a $b \in \mathbb{F}_2^D$, if $Hb = \mathbf{0}$, then for all $a \in \mathbb{F}_2^\ell$ $a^\top Hb = 0$ and thus $\sum_{a\in\mathbb{F}_2^\ell}(-1)^{a^\top Hb} = 1$. If $Hb \neq \mathbf{1}$, then half of the $a \in \mathbb{F}_2^\ell$ has inner product 1 with Hb and half of them has inner product 0. As a result, $\sum_{a\in\mathbb{F}_2^\ell}(-1)^{a^\top Hb} = 0$. The above discussion implies that

$$\lambda(H) = \frac{1}{|S|}\sum_{a\in\mathbb{F}_2^\ell}\sum_{b\in\mathbb{F}_2^D}\chi_H(ab^\top) \geq 0.$$

We conclude that $M_{D,\ell}$ is positive semidefinite. □