January 2005

# The Unique Games Conjecture and Some of Its Implications on Inapproximability

Boulos Harb
*University of Pennsylvania*

## Recommended Citation

Boulos Harb, "The Unique Games Conjecture and Some of Its Implications on Inapproximability", . January 2005.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-05-18

## The Unique Games Conjecture and Some of Its Implications on Inapproximability

### Abstract

In this report, we study the Unique Games conjecture of Khot [32] and its implications on the hardness of approximating some important optimization problems. The conjecture states that it is NP-hard to determine whether the value of a unique 1-round game between two provers and a verifier is close to 1 or negligible. It gives rise to PCP systems where the verifier needs to query only 2 bits from the provers (in contrast, Håstad's verifier queries 3 bits [44]). We start by investigating the conjecture through the lens of Håstad's 3-bit PCP. We then discuss in detail two results that are consequences of the conjecture. The first states that Min-2SAT-Deletion is NP-hard to approximate within any constant factor [32]. The second result shows that minimum vertex cover is NP-hard to approximate within a factor of $2 - \varepsilon$ for every $\varepsilon > 0$ [34]. We display the use of Fourier techniques for analyzing the soundness of the PCP used to prove the first result, and we display the use of techniques from extremal combinatorics for analyzing the soundness of the PCP used to prove the second result. Finally, we present Khot's algorithm which shows that for the conjecture to be true, the domain of answers of the two provers must be large, and we survey some recent results examining the plausibility of the conjecture.

### Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-05-18

# The Unique Games Conjecture and some of its Implications on Inapproximability

Boulos Harb

May 9, 2005

## Abstract

In this report, we study the Unique Games conjecture of Khot [32] and its implications on the hardness of approximating some important optimization problems. The conjecture states that it is NP-hard to determine whether the value of a *unique* $1$-round game between two provers and a verifier is close to $1$ or negligible. It gives rise to PCP systems where the verifier needs to query only $2$ bits from the provers (in contrast, Håstad's verifier queries $3$ bits [44]). We start by investigating the conjecture through the lens of Håstad's 3-bit PCP. We then discuss in detail two results that are consequences of the conjecture. The first states that Min-2SAT-Deletion is NP-hard to approximate within any constant factor [32]. The second result shows that minimum vertex cover is NP-hard to approximate within a factor of $2 - \epsilon$ for every $\epsilon > 0$ [34]. We display the use of Fourier techniques for analyzing the soundness of the PCP used to prove the first result, and we display the use of techniques from extremal combinatorics for analyzing the soundness of the PCP used to prove the second result. Finally, we present Khot's algorithm which shows that for the conjecture to be true, the domain of answers of the two provers must be large, and we survey some recent results examining the plausibility of the conjecture.

# Contents

# 1 Introduction

Many important optimization problems are NP-hard to solve exactly in the worst case. When faced with such a problem, we have to settle for an approximate solution. An approximation algorithm for an NP-hard problem is a Turing Machine that produces a feasible solution to the given problem that is within a guaranteed factor of the optimum solution. Usually, this factor is taken to be greater than 1, so for a maximization problem, an approximation algorithm that achieves a factor $C$ produces a solution whose value is at least OPT$/C$, where OPT refers to the problem's global optimum solution. For a minimization problem, a factor $C$ approximation algorithm produces a solution whose value is at most $C$OPT. Finding an approximation algorithm is one aspect of studying the approximability of an NP-hard problem. The other aspect is proving, under certain assumptions, that the problem cannot be approximated within a certain factor. Such results that rule out the possibility of an approximation algorithm are referred to as *hardness* results or *inapproximability* results. Usually, they are based on the assumption that P$\neq$NP, thus ruling out the possibility of a polynomial time approximation algorithm.

Early inapproximability results are due to Garey and Johnson [23]. However, strong inapproximability results for many problems were not obtained until the connection between approximation hardness and multiprover interactive proofs was discovered by Feige *et al.* [19]. An interactive proof can be viewed as a game between a computationally unbounded prover, and a polynomial time algorithm called the verifier with access to random bits. The prover wants to convince the verifier of some fact, e.g. that a given string is in a language, and the verifier (probabilistically) decides whether to accept the fact or not by querying the prover. The Unique Games conjecture [32] is about a certain type of interactive proof with 2 provers. It implies strong inapproximability results showing for example that unless P $=$ NP, Min-2SAT-deletion cannot be approximated within any constant factor, and that under the same assumption vertex cover in $k$-uniform hypergraphs cannot be approximated within a factor of $k - \epsilon$ for every $k \geq 2$ and $\epsilon > 0$. In particular, this means that the conjecture would settle the vertex cover problem on graphs since there exists a factor 2 approximation algorithm for this problem. The conjecture would also settle the Max-Cut problem as it would imply that the approximation factor achieved by the algorithm of Goemans and Williamson [25] is the best possible [30, 39].

## 1.1 Brief History

Interactive proofs were introduced by Goldwasser, Micali and Rackof [26], and Babai [6]. Ben-Or *et al.* [9] defined the notion of multiprover interactive proofs where the verifier interacts with provers who cannot communicate with each other. Fortnow, Rompel and Sipser [21] showed that the class of languages that have multiprover interactive proofs equals the class of languages that have (in today's terms) probabilistically checkable proofs (PCP) with polynomial randomness and query complexity (the number of bits examined by verifier), i.e. MIP $=$ PCP$(\text{poly}, \text{poly})$. In a PCP, instead of interacting with the verifier, the provers write down the entire proof. The verifier decides whether to accept or reject the proof by checking a few randomly selected positions of the proof.

A breakthrough result of Lund *et al.* [37] demonstrated the power of interactive proof systems by using algebraic techniques to show that all co-NP statements have interactive proofs. Using these techniques, Shamir [43] showed that all decision problems which may be solved using a polynomial amount of memory have interactive proofs and vice versa, i.e. IP $=$ PSPACE. The result of Babai, Fortnow and Lund [7] showing that MIP $=$ NEXP further established the power of interactive proofs and enabled the connection with hardness of approximation. Feige *et al.* [19] made the connection by showing that NEXP $\subseteq$ PCP$(\text{poly}, \text{poly})$ implies that Max-Clique is hard to approximate unless EXPTIME $=$ NEXP. They achieved a hardness of approximation result under an assumption closer to P $\neq$ NP by showing that NP $\subseteq$ PCP$(\log n \log\log n, \log n \log\log n)$. This established that Max-Clique is hard to approximate within any constant factor unless problems in NP can be solved in $n^{O(\log\log n)}$ time.

Following the result of Feige *et al.* improved characterizations of NP were sought. Arora and Safra [5] formalized and named the class PCP. They introduced the idea of proof composition, which turned out to be fundamental in all subsequent developments, and showed that languages in NP have PCP verifiers that use logarithmic randomness and sub-logarithmic query complexity. Arora *et al.* [3] reduced the query complexity to constant, thus proving the celebrated PCP Theorem (Thm. 2.8). They also showed that Max-3SAT cannot be approximated within some constant factor. Bellare, Goldreich and Sudan [8] showed that this constant is $27/26$. Their result showed that in order to get strong hardness results, one needs to design PCP's with the specific application in mind.

Max-3SAT is a constraint satisfaction problem with three variables per constraint. Following the philosophy of [8], Håstad [44] proved that unless P $=$ NP, Max-3SAT cannot be approximated within a factor of $8/7$, which is a tight result, by constructing a PCP whose query complexity is 3, i.e. the verifier only needs to read 3 bits of the proof. The approach used to prove this result is similar to that of [8]. The starting point is a multiprover protocol, which comes from a combination of the PCP Theorem and Raz's parallel repetition theorem [40]. The protocol is transformed into a PCP by writing down the prover's answers in coded form. Håstad showed that the encoding introduced by [8] enables the verifier to check the proof by reading only 3 bits. The verifier in the multiprover system is known as the Raz Verifier, and the verifier that reads the encoded proof is called the inner verifier. Håstad's result also implies that Max-2SAT is NP-hard to approximate within any factor less than $22/21$. This factor is, however, not tight. Max-2SAT is a constraint satisfaction problem with two variables per constraint and we seem to have no techniques for constructing PCP's where the verifier can read only 2 bits. Khot [32] suggested the Unique Games conjecture as a possible direction for designing such PCP's. The Unique Games conjecture stipulates the existence of a verifier with stronger properties than the Raz Verifier. Having this powerful *outer* verifier enables the design of inner verifiers to prove strong inapproximability results for such problems as Max-2SAT and vertex cover. Nonetheless, even with such a powerful outer verifier, the inner verifiers are typically non-trivial relying on deep theorems in Fourier analysis.

## 1.2 Organization of the Report

The main focus of this report is understanding the Unique Games conjecture and presenting the results of [32, 34] based on it.

Section 2 defines some of the problems we consider and provides some necessary background. It ends with a description of 2-prover 1-round games and the Raz Verifier, thus setting up the stage for the discussion that follows.

We investigate the Unique Games conjecture by studying Håstad's 3-bit test. The Unique Games conjecture enables the design of a similar test that queries 2 bits of the witness proof. We will provide intuition behind the need for the third bit in Håstad's verifier and how the conjecture alleviates this need. This is done in Section 3. In Sec. 4 we show how the 2-bit test allows us to prove that it is NP-hard to distinguish between instances of Max-2SAT that are $(1-\epsilon)$-satisfiable and instances that are $(1-\epsilon^{\frac{1}{2}+o(1)})$-satisfiable for all sufficiently small $\epsilon > 0$. We also briefly present a verifier based on the Unique Games conjecture that shows a tight hardness result for coloring a 3-uniform hypergraph with 3 colors, and we state some recent results based on the conjecture.

Section 5 presents the result of Khot and Regev [34] which shows that vertex cover is hard to approximate within $2-\epsilon$ for any $\epsilon > 0$ assuming the Unique Games conjecture. The construction of the hard instance of vertex cover is very similar to that of [14] which shows that vertex cover in $k$-uniform hypergraphs is NP-hard to approximate within any constant factor smaller than $\lfloor k/2 \rfloor$, $k \geq 4$. In fact, the two constructions coincide when $k = 2$. Given this graph construction, we show how the proof of the latter result cannot be used to prove $2 - \epsilon$ hardness for vertex cover on graphs. We also discuss why the proof of [34] fails to give the desired result if the graph construction is based on the Raz Verfier. Assuming the construction is based on the Unique Games conjecture, we present the proof of [34].

Finally, the plausibility of the conjecture is discussed in Sec. 6. We end with a presentation of Khot's SDP based algorithm [32], which shows that for the conjecture to be true, the domain of answers of the provers must be large.

# 2 Background

## 2.1 Problems Considered

In this section, we define some of the problems we will be considering.

**Definition 2.1 (Max-$k$Lin-$p$).** *Let $p$ be a prime. Max-$k$Lin-$p$ is the problem of given a system of linear equations over the field $\mathbb{Z}_p$ with exactly $k$ variables in each equation, find the maximum number of equations that can be satisfied by any assignment.*

We will specifically be interested in the problems Max-3Lin-2 and Max-2Lin-2.

**Definition 2.2 (Max-$k$SAT).** *Max-$k$SAT is the problem of given a $k$-CNF formula (i.e. each clause contains exactly $k$ variables), find the maximum number of clauses that can be satisfied by any assignment.*

We will specifically be interested in Max-3SAT and Max-2SAT. The minimization version of Max-2SAT, where the objective is to find the minimum number of constraints that cannot be satisfied by any assignment, is called Min-2SAT-Deletion (or Min-2CNF-Deletion).

A $q$-uniform hypergraph $H = (V, E)$ consists of a set of vertices $V$ and a set of hyperedges $E$ where every hyperedge is a subset of vertices of size $q$. A hypergraph is said to be $k$-colorable if each of its vertices can be assigned a color from a set of $k$ colors such that no hyperedge is monochromatic, i.e. not all its vertices have the same color. A non-monochromatic hyperedge is said to be *correctly colored*. We will mostly be interested in the maximization version of hypergraph coloring defined below.

**Definition 2.3 (Hypergraph $k$-Coloring).** *Hypergraph $k$-Coloring is the problem of given a $q$-uniform hypergraph and $k$ different colors, find an assignment of colors to the vertices so as to maximize the number of correctly colored hyperedges.*

The minimization version is called Approximate Coloring, and it is the problem of given a $k$-colorable hypergraph, color it with as few colors as possible.

A *vertex cover* of a hypergraph $H$ is a subset of vertices $V' \subseteq V$ that contains at least one end point of each hyperedge $e \in E$, i.e. $e \cap V' \neq \emptyset$. The complement of a vertex cover is called an *independent set*, i.e. it is a subset of vertices that does not contain any hyperedge entirely within it.

**Definition 2.4 (E$k$-Vertex-Cover).** *E$k$-Vertex-Cover is the problem of given a hypergraph $H = (V, E)$, find a minimum size vertex cover in $H$.*

The problem E2-Vertex-Cover is simply the minimum vertex cover problem on graphs.

## 2.2 The Classes NP and PCP

Before defining the class of languages that have Probabilistically Checkable Proofs (PCP's) we recall the definition of the class NP in terms of the existence of a deterministic polynomial time verifier that can check language membership proofs.

**Definition 2.5 (NP).** *A language $L$ is in $NP$ if and only if there exists a deterministic polynomial time verifier $V$ such that given a string $x \in \{0,1\}^n$ it satisfies,*

- *Completeness: If $x \in L$, then there is a string $y$ with $|y| = n^{O(1)}$ such that $V(x,y) = 1$*
- *Soundness: If $x \notin L$, then for all $y$ with $|y| = n^{O(1)}$, $V(x,y) = 0$.*

The running time of $V$ is assumed to be polynomial in the length of $x$. We say that $V$ *accepts* $x$ when it outputs 1; otherwise, we say that it *rejects* $x$. We will refer to $y$ in the above definition as the *proof.*

A PCP is described by a probabilistic verifier that randomly examines a few bits of a written proof $y$. We say that the verifier $V$ has *oracle access* to $y$, and we write $V^y$ to indicate that $V$ does not receive $y$ explicitly. We are interested in two properties of $V$, namely, the number of coins $V$ flips and the number of bits of the proof it reads.

**Definition 2.6.** *A $(r(n), q(n))$-restricted verifier is a probabilistic polynomial time Turing machine such that given input $x$ of length $n$ and oracle access to proof $y$, it uses $r(n)$ random bits to list $q(n)$ positions of $y$, queries $y$ at these positions, and accepts or rejects $x$ based on the values it receives.*

The running time of $V$ is again assumed to be polynomial in the size of the input $x$. Note that $r(n)$ and $q(n)$ are bounded by the running time of $V$. Futhermore, $V$ is *non-adaptive* – it simultaneously decides which queries to make. The parameter $q(n)$ is called the *query complexity* of $V$. We can now define the class of languages PCP$_{c,s}[r(n), q(n)]$.

**Definition 2.7 (PCP).** *A language $L$ is in PCP$_{c,s}[r(n), q(n)]$ if there exists a $(r(n), q(n))$-restricted verifier $V$ such that given a string $x \in \{0,1\}^n$ it satisfies,*

- *Completeness: If $x \in L$, then there is a proof $y : \Pr[V^y(x) = 1] \geq c$ ;*
- *Soundness: If $x \notin L$, then for all $y$, $\Pr[V^y(x) = 1] < s$ ,*

*where the probabilities are taken over $V$'s choice of random bits and $0 \leq s < c \leq 1$. Furthermore, for any $y$, $|y| \leq q(n) \cdot 2^{r(n)}$.*

The bound on $|y|$ is determined by the number of possible positions of $y$ that $V$ can examine. All other bits of $y$ are irrelevant. If $c = 1$ we say that the verifier has *perfect completeness*, and if $c = 1 - o(1)$, we say it has *almost perfect completeness*. If $L \in$ PCP$_{c,s}[r,q]$, we say that $L$ has a PCP with parameters $(r,q)$.

## 2.3 The PCP Theorem

It is immediate from definition 2.7 that NP $=$ PCP$_{1,0}(0, \text{poly}(n))$. The PCP Theorem states the following suprising result:

**Theorem 2.8.** *[3, 5] NP $=$ PCP$_{1,\frac{1}{2}}[O(\log n), O(1)]$*

One side of this equality, $\text{PCP}_{1,\frac{1}{2}}[O(\log n), O(1)] \subseteq$ NP, is easy to see. Given a language $L \in \text{PCP}_{1,\frac{1}{2}}[O(\log n), O(1)]$ with verifier $V$, we can construct a deterministic verifier $V'$ that simulates $V$ on all $2^{O(\log n)} = \text{poly}(n)$ random coin flips and accepts if and only if $V$ accepts on all runs.

The PCP Theorem provides a "robust" characterization of the class NP in the sense that any proof of a false statement must be wrong almost everywhere since in order to reject with probability more than a half, it suffices for the verifier to check only a few a bits of a proof. As we will see below, this robustness allows us to reduce a languange $L \in$ NP to a 3SAT formula such that if $x \in L$, then the formaula is satisfiable, and if $x \notin L$, then no assignment can satisfy more than $1 - \epsilon$ fraction of the clauses of the formula. This shows the relationship between the PCP Theorem and the inapproximability of Max-3SAT, where the objective is to satisfy the maximum number of constraints in a given formula. The gap in the reduction implies that Max-3SAT does not have a $(1 + \epsilon)$-approximation unless $P = NP$. In fact, the PCP Theorem is equivalent to the inapproximability of Max-3SAT.

**Theorem 2.9.** *[3] NP $=$ PCP$_{1,\frac{1}{2}}[O(\log n), O(1)]$ if and only if there is a constant $\epsilon > 0$ for which there exists a polynomial time reduction $f$ from any language $L \in$ NP to Max-3SAT such that*

- *If $x \in L$, OPT$(f(x)) = 1$,*
- *if $x \notin L$, OPT$(f(x)) < 1 - \epsilon$.*

Here, $\text{OPT}(f(x))$ refers to the maximum fraction of constraints of the formula $f(x)$ that any assignment can satisfy. In general, OPT will be clear from context. The reduction above is called a *gap-introducing reduction* as it introduces a gap of factor $1/(1-\epsilon)$ between the two classes of instances of Max-3SAT (those constructed from instances $x \in L$ and those constructed from instances $x \notin L$). As noted above, this gap establishes the approximation hardness of Max-3SAT. Suppose there is a $1/(1-\epsilon)$ factor approximation algorithm $\mathcal{A}$ for Max-3SAT. Then if $x \in L$, $\mathcal{A}(f(x)) \geq (1-\epsilon)\text{OPT}(f(x)) = 1 - \epsilon$, and if $x \notin L$, then $\mathcal{A} < 1 - \epsilon$. Hence, using $\mathcal{A}$ we can decide any NP language $L$; a contradiction unless P = NP. Even though the proof of the above theorem can be found in many places (see for example [2]), we will give the proof here as it shows that the choice of $1/2$ is arbitrary and can be replaced by any small constant, and it displays the importance of desinging PCPs that are very closely connected to the optimization problem whose hardness we are trying to prove.

*Proof of Theorem 2.9.* **(if)** Assume that $L \in$ NP and there is a gap-introducing reduction $f$ as in the statement of the theorem. Given input $x$, the PCP verifier we construct first runs $f$ to create a Max-3SAT formula $f(x)$. It then randomly selects a clause $C$ from $f(x)$ using its $O(\log n)$ random bits. Let a proof $y$ correspond to an assignment of the variables of $f(x)$. The verifier $V$ reads the values of the 3 variables in $C$ from $y$, and accepts if and only if the variables satisfy $C$. Hence, if $x \in L$, letting $y$ be a satisfying assignment to $f(x)$ we have $\Pr[V^y(x) = 1] = 1$. On the other hand, if $x \notin L$, then for any assignment $y$, $\Pr[V^y(x) = 1] < 1 - \epsilon$. Since $\epsilon$ is a constant, this probability can be reduced to $1/2$ (or any other small constant) at an exponential rate by a constant number of repetitions.

**(only if)** Assume that $L \in$ NP. By the PCP Theorem, $L \in \text{PCP}_{1,\frac{1}{2}}[c \log n, q]$ where $c$ and $q$ are constants. Let $V$ be its PCP verifier. Given input $x$ of length $n$, let $y$ be a proof to which $V$ has oracle access. For a random string of length $c \log n$, $V$ queries $q$ positions of the proof and decides to accept or reject based on the values it receives from $y$. We generate a boolean variable corresponding to each position in $y$ (so that $y$ corresponds to an assignment to those variables). Further, we generate a boolean function $f_r$ whose domain is $\{0,1\}^q$ for each random string $r$ of length $c \log n$. The function $f_r : \{0,1\}^q \to \{0,1\}$ takes as input the values assigned by $y$ to the $q$ variables that correspond to the $q$ positions $V$ queries given $r$. The output of $f_r$ is 1 if and only if $V$ accepts. By simulating $V$ on all $n^c$ random strings we get $n^c$ such boolean functions. The truth table of each boolean function can be represented by at most $2^q$ $q$-CNF clauses, and each such clause can be transformed to at most $(q-2)$ 3CNF clauses in the standard way. Hence, we end up with a 3CNF formula $\phi$ with $n^c 2^q (q-2)$ clauses.

Now if $x \in L$, then there exists a proof such that every test causes $V$ to accept; hence, the formula is satisfiable. If $x \notin L$, then any proof $y$ causes more than $n^c/2$ of the tests to reject. Hence, the fraction of unsatisfied clauses in our formula is $> (n^c/2)/(n^c 2^q (q-2)) = 1/(2^{q+1}(q-2)) = \epsilon$. That is OPT($\phi$) $< 1-\epsilon$ and $\epsilon$ is a constant. □

### 2.3.1 The Relationship to Constraint Satisfaction Problems

The gap in Thm. 2.9 is so small because the translation from boolean functions with domain $\{0,1\}^q$ to 3CNF clauses produced a large number of clauses. Intuitively, we can get a better approximation hardness result if we had a PCP verifier that needs to read a smaller number of bits. Max-3SAT is a constraint satisfaction problem on 3 variables. In fact, we now highlight the relationship between PCP's and the hardness of approximating constraint satisfaction problems on $k$ variables ($k$-CSP's) in general. In a $k$-CSP, we are given a set of variables and a set of constraints. Each constraint depends on exactly $k$ variables. The goal is to find an assignment to the variables that maximizes the number of satisfied constraints. Designing a specific verifier whose query complexity is $k$ implies a hardness result for a $k$-CSP. We let the positions of the proof be the variables of the problem and the verifier's possible tests (given its random bits) be the constraints. A proof defines an assignment to the variables. Hence the acceptance probability of the verifier equals the fraction of satisfied constraints, and the hardness factor is obtained from the ratio between the completeness and soundness of the constructed PCP system.

The next section shows a different characterization of NP that will allow us to design PCP's with lower query complexity.

## 2.4 2-Prover 1-Round Games

In order to design PCP's with low query complexity, we will need a detailed description of the queries made by the PCP verifier. We will design a new proof system with two provers and a simple probabilistic verifier. The system is best thought of as a game between the provers, $P_1$ and $P_2$ and the verifier $V$ where the provers are trying to convince the verifier of the validity of an NP statement of length $n$ (e.g. a formula that is claimed to be satisfiable). The two provers are cooperating and infinitely powerful. They can make any agreement before the start of the game, however, once the interaction with the verifier starts, they can no longer communicate. The verifier is allowed to ask each prover only one question; hence, the game is 1-round. It has access to $r(n)$ random bits, which it uses to generate two questions $q_1$ and $q_2$ without communicating with the provers. Note that this implies that the verifier is non-adaptive as it does not produce the second question based on the first answer it receives. The verifier simultaneously sends $q_1$ to $P_1$ and $q_2$ to $P_2$. Prover $P_1$ does not have access to $q_2$, and prover $P_2$ does not have access to $q_1$. The provers answer with $P_1(q_1)$ and $P_2(q_2)$. Since the verifier can ask the two provers for the same information, the provers' ability to cheat gets restricted. The verifier decides whether to accept or reject after receiving both answers $P_1(q_1)$ and $P_2(q_2)$. We now define the class of languages 2P1R$_{c,s}[r(n)]$ that are recognized by such verifiers.

**Definition 2.10.** *[44] A language $L$ is in 2P1R$_{c,s}[r(n)]$ if there exists a probabilistic polynomial time verifier $V$ that receives $r(n)$ random bits such that given a string $x \in \{0,1\}^n$ it produces two queries $q_1$ and $q_2$*

*based only on its random bits and $x$ and satisfies,*

- *Completeness: If $x \in L$, there exist two provers $P_1$ and $P_2$ whose answers $P_1(q_1)$ and $P_2(q_2)$ to queries $q_1$ and $q_2$ respectively cause $V$ to accept with probability at least $c$ ;*
- *Soundness: If $x \notin L$, then for any two provers $P_1$ and $P_2$, the probability that $V$ accepts based on the answers $P_1(q_1)$ and $P_2(q_2)$ is at most $s$ ,*

*where the probabilities are taken over $V$'s choice of random bits, and $0 \leq s < c \leq 1$.*

The *value* of a 2-prover 1-round game (2P1R) is the maximum acceptance probability of the verifier.

Note that the number of random bits available to $V$ limits the domain of questions $V$ can ask. This in turn limits the number of answers the provers need to prepare. We can thus turn the above game into a PCP simply by writing down each prover's answers indexed by the questions $V$ can ask the prover. It is noteworthy that if the game is not 1-round, then we cannot think of $P_1$ and $P_2$ as written proofs since the provers are infinitely powerful and hence can be considered adaptive. Next we construct a 2-prover 1-round PCP with logarithmic randomness that captures NP.

**A PCP with Low Query Complexity and Soundness close to $1$.** Given a language $L \in$ NP, we use the gap-introducing reduction $f$ given in Thm. 2.9 to transform any instance $x$ to a Max-3SAT formula $\phi_x$. Suppose $\phi_x$ has $n$ variables and $m$ clauses. Our 2P1R verifier works as follows. It assumes that prover $P_1$ is a string containing a truth assignment to the $n$ variables (i.e. each position takes one of two values). Furthermore, it assumes that prover $P_2$ is a string containing for each clause a *satisfying* assignment to its $3$ variables (i.e. each position takes one of 7 values). It uses its $O(\log n)$ random bits to pick a clause $C$ from $\phi_x$, and a random variable $z$ occurring in $C$. It queries $P_1$ at $x$ and $P_2$ at $C$. It receives a 1-bit answer $P_1(z)$ from the first prover and a 3-bit answer $P_2(C)$ from the second prover. Note that $P_2(C)$ implicitly contains an assignment to $z$. The verifier accepts if and only if $P_1(z)$ and the implicit assignment to $z$ in $P_2(C)$ are equal.

If $x \in L$ then $\phi_x$ is satisfiable and clearly there are proofs $P_1$ and $P_2$ that make the verifier accept with probability 1.

If $x \notin L$, then more than $\epsilon$ fraction of the clauses of $\phi_x$ are not satisfiable where $\epsilon$ is the constant in Thm. 2.9. Since $P_1$ is an assignment to the variables, more than $\epsilon m$ clauses are not satisfied by it. Suppose we pick an unsatisfiable clause $C$. This happens with probability $> \epsilon$. Since $P_2$ contains only satisfying assignments, its assignment to $C$ must differ from $P_2$ in at least one variable. The probability that we catch this inconsistency is at least $1/3$. Hence, the soundness of this 2P1R game is less than $1 - \epsilon/3$.

The above PCP is good in that $V$ queries only $4$ bits of the proof; however, its acceptance probability is always close to 1. As in the proof of Thm. 2.9, we would like to use a constant number of repetitions to reduce the acceptance probability in the soundness case. Repeating the above procedure $u$ independent times reduces the error probability to $(1 - \epsilon/3)^u$. If we do that, however, the game is no longer one round. We will use a different technique known as *parallel repetition*.

## 2.5 The Raz Verifier

Parallel repetition simply means that $V$ randomly chooses $u$ clauses $(C_i)_{i=1}^u$ and for each clause $C_i$ it chooses one variable $z_i$ at random. The verifier sends $q_1 = (z_i)_{i=1}^u$ to $P_1$ and $q_2 = (C_i)_{i=1}^u$ to $P_2$ all at once. It assumes that each position in $P_1$ is indexed by $u$ variables and contains an assignment to the $u$ variables. Thus, the length of $P_1$ is $n^u$ and each position takes one of $2^u$ values. Further, $V$ assumes each position in $P_2$ is indexed by $u$ clauses and contains a sequence of satisfying assignments to the $u$ clauses. Thus, the length of $P_2$ is $m^u$ and each position takes one of $7^u$ values. Verifier $V$ then receives the answers from $P_1$ and $P_2$ and accepts if all $u$ variable assignments it receives from $P_1$ are consistent with all $u$ clause assignments it receives from $P_2$.

Since the provers can see all answers, it is not clear that the error probability of this game is $(1-\epsilon/3)^u$ In fact, the error probability can be greater than that; however, in [40], Raz showed that the error probability indeed decreases exponentially with $u$.

**Theorem 2.11.** *[40] Given a 2-prover 1-round game with soundness $s < 1$ and answer size $d$, there exists $s' < 1$ that depends only on $s$ such that for all integers $u$ the soundness of $u$ parallel repetitions of the game is $(s')^{u/d}$.*

Hence, since the answer size of our original game is constant, by choosing $u$ to be a large enough constant, we can make the soundness arbitrarily small. However, the number of bits queried is now $u + 3u$. Note that the size of the domain of answers of the provers is a constant that depends on the soundness parameter. This 2P1R game with perfect completeness and arbitrarily low soundness is known as the *Raz Verifier*. In [44], Håstad uses the Raz Verifier to construct a 3-bit PCP. We will reserve the discussion of Håstad's PCP to the next section where we motivate the Unique Games conjecture.

# 3 The Unique Games Conjecture

The Unique Games conjecture (UGC) [32] is the following:

**Conjecture 3.1 (Unique Games Conjecture).** *For arbitrarily small constants $\zeta, \delta > 0$, there exists a constant $k = k(\zeta, \delta)$ such that it is NP-hard to determine whether a unique 2-prover 1-round game with answers from a domain of size $k$ has value at least $1 - \zeta$ or at most $\delta$.*

Why is the UGC stated as such? In this section we attempt to answer this question. We feel that the best way to provide intuition behind the conjecture is to describe Håstad's 3-bit PCP. We will also define *unique* games and describe a problem called Label Cover that is equivalent to a 2P1R game.

## 3.1 Overview of Håstad's PCP

Håstad's result is the following:

**Theorem 3.2.** *[44] For all $\epsilon, \eta > 0$,*

$$\mathit{NP} = \mathit{PCP}_{1-\epsilon, \frac{1}{2}+\eta}[O(\log n), 3] \enspace .$$

*Moreover, the acceptance condition of the verifiers is linear (i.e. if the three bits read from the proof are $b_1$, $b_2$ and $b_3$, the acceptance condition is either $b_1 + b_2 + b_3 = 0$ or $b_1 + b_2 + b_3 = 1$).*

The starting point of Håstad's PCP is the Raz Verifier described in Sec. 2.5. We will refer to the Raz Verifier as the *outer verifier*. Recall that this verifier has perfect completeness and arbitrarily low soundness. The problem, however, is that it reads answers from a large alphabet that is dependent on the soundness parameter. To achieve our goal, we will build a new verifier called the *inner verifier* that expects as a proof encodings of the provers' answers using a predefined encoding scheme. With a suitable encoding, the inner verifier can perform its test efficiently. A cheating prover, however, may not abide by the encoding. Hence, besides checking the consistency of the answers, the inner verifier must also check if the encodings of the answers are correct. Håstad's construction integrates these two tasks into a single test that reads only 3 bits. The test does not explicitly check that the encodings are correct. Instead, it is shown that if a proof makes the inner verifier accept with high probability, then there is a way to "decode" the proof and extract strategies for the provers that would make the outer verifier accept with probability greater than $\delta$. This leads to a contradiction if the soundness of the outer verifier is less than $\delta$. The strategies are extracted by analyzing the encoded answers using Fourier Analysis.

It suffices for our purposes to describe the 3-bit test. Even though we will not go over the soundness analysis of this test, the technique of using discrete Fourier transforms to extract strategies from encoded answers will be displayed when we analyze a different test in Sec. 4.2.3. We start by describing the encoding expected by the inner verifier.

## 3.2 The Long Code

The *long code* was introduced by [8]. The long code of an element $x \in \{0,1\}^u$ is a string of length $2^{2^u}$. It is a wasteful encoding; however, it is very useful for our purposes.

**Definition 3.3 (Long Code).** *Let $\mathcal{F}_M$ be the family of boolean functions $f : M \to \{0,1\}$. The long code of an element $x \in M$ is a map $A_x : \mathcal{F}_M \to \{0,1\}$ where $A_x(f) = f(x)$.*

The usefulness of the long code is apparent when we consider the type of questions the inner verifier should ask. In order not to waste any bits, the inner verifier will ask boolean questions. Suppose that the answer to the outer verifier's first query is $x$. The question the inner verifier will ask about $x$ is, "Does $x$ belong to the following set of values?" Since $x$ is a $u$-bit string, there are $2^u$ possible values for $x$, and hence $2^{2^u}$ possible subsets the inner verifier can inquire about. Note that $u$ in our case is a constant depending on the number of parallel repetitions the outer verifier performs, thus the inner verifier can ask such questions in constant time. The long code encodes the answer of the prover for every possible subset $S \subseteq \{0,1\}^u$. Hence, the long code for $x$ is a $2^{2^u}$ bit string where position $i = 1$ if $x \in S_i$, and $0$ otherwise (we use an arbitrary but fixed convention to order the subsets of $\{0,1\}^u$).

We can identify a set $S \subseteq \{0,1\}^u$ by a function $f_S : \{0,1\}^u \to \{0,1\}$. That is, $S = \{x \in \{0,1\}^u : f_S(x) = 1\}$. Asking if $x$ is in $S$ is equivalent to evaluating $f_S$ at $x$. Now since the set of all subsets of $\{0,1\}^u$ corresponds to the family of functions $f : \{0,1\}^u \to \{0,1\}$ we arrive at definition 3.3 above.

When working with long codes, it is sometimes more convenient to work with boolean variables from $\{1,-1\}$ rather than the standard $\{0,1\}$. We let $-1$ denote true so that multiplication represents the exclusive-or of two bits. The reason we use this multiplicative representation will become apparent when we utilize Fourier techniques to analyze the long codes. However, we use it below to define the mechanism of *folding* introduced by [8].

**Definition 3.4 (Folding).** *A function $A : \mathcal{F}_M \to \{1,-1\}$ is folded if for all $f \in \mathcal{F}$, $-A(f) = A(-f)$.*

A correct long code is clearly folded, since for $S \subseteq M$, $x \in S$ iff $x \notin \bar{S}$. In order to implicitly ensure that a long code written by a prover is folded, we store (in an arbitrary but fixed mannar) for each pair of functions $(f, -f)$ one representative. When we want to access the other function, we negate the result we read. Suppose $f$ is chosen for example. Then, if we want to evaluate $f(x)$, we simply read $A_x(f)$ where $A_x$ is the long code for $x$. If we wish to evaluate $-f(x)$, then we read $A_x(f)$ and negate the result.

The fact that the number of parallel repetitions executed by the Raz Verifier is constant allowed the inner verifier to use the long code to encode the provers' answers. Another important property of the Raz Verifier is described below and it will allow us to design the 3-bit test. A modification of this property given by the Unique Games conjecture allows us to design a 2-bit test.

### 3.3 The Raz Verifier's Projection Property

Let us a look at a concrete example of a round of interaction between the Raz Verifier and the provers. Let the number of parallel repetitions be $u = 2$. Suppose the verifier $V$ randomly picks the clauses $(C_1, C_2) = (x_1 \vee \bar{x_2} \vee x_3, \bar{x_1} \vee x_4 \vee x_5)$ and the variables $(z_1, z_2) = (x_1, x_5)$. The verifier then probes the provers and receives answers $P_1(z_1, z_2)$ and $P_2(C_1, C_2)$. Note that given the set of probes and $P_2$'s answer, there is a unique answer of $P_1$ that would make $V$ accept. In our example, if $V$ receives $(110, 010)$ from $P_2$, then the only answer received from $P_1$ that would make $V$ accept is $(1, 0)$. That is, the accepted answer of $P_1$ is the *projection* of the answer of $P_2$ at $(x_1, x_5)$. This implies that for every possible pair of questions $q_1$, $q_2$ to provers $P_1$ and $P_2$, there is a projection $\pi_{q_1,q_2} : [7^u] \rightarrow [2^u]$ such that $V$ accepts if and only if the answers $P_1(q_1)$, $P_2(q_2)$ satisfy $\pi_{q_1,q_2}(P_2(q_2)) = P_1(q_1)$. This projection property is almost all we need to design the inner verifier.

Let $R_Y, R_X$ be the sets of possible answers the outer verifier can receive from provers $P_1$ and $P_2$ respectively. That is, $|R_Y| = 2^u$, and $|R_X| = 7^u$ where $u$ is the number of parallel repetitions. Suppose the outer verifier receives answer $a_1$ (resp. $a_2$) from $P_1$ (resp. $P_2$) in response to question $q_1$ (resp. $q_2$). We are trying to verify if the two answers are consistent, i.e. if they satisfy $\pi_{q1,q2}(a_2) = a_1$. For ease of notation, define $\pi := \pi_{q1,q2}$. The inner verifier will pick a random set $F$ from the range $R_Y$ of the projection $\pi$, and it will ask $P_1$ if $a_1$ belongs to $F$. If $a_1 \in F$, then the set of consistent answers received from $P_2$ is limited to $F' = \pi^{-1}(F) \subseteq R_X$. The inner verifier will accept if and only if $a_1 \in F$ and $a_2 \in F'$, or $a_1 \notin F$ and $a_2 \notin F'$ resulting in a 2-bit test (i.e. the exclusive-or of the two provers' answers is 0). Note that since the long codes are folded, the provers cannot always pass the test by simply answering 1 to every query. Nonetheless, this test does not work as it can disclose the set of variables in $q_1$ to $P_2$. Intuitively, this is because even though $F$ is random, the values in $F'$ are correlated allowing $P_2$ to infer $F$ and $q_1$ [18]. Knowing $q_1$ enables $P_2$ to pick an assignment to the clauses in $q_2$ that is consistent with $P_1$'s assignment to the variables in $q_1$, thus ensuring that the outer verifier accepts. Recall that the provers can make any agreement before the start of the interaction with the outer verifier, and specifically, they can agree on $P_1$'s assignment. Note that this does not contradict the soundness of the outer verifier as its soundness depends on the fact that each prover does not know the question directed to the other prover (i.e. $q_1$ is hidden from $P_2$ and vice versa). Going back to the pair of clauses in our example above, suppose for simplicity that $F' = \{(10\mathbf{0}, 0\mathbf{0}1), (11\mathbf{0}, 0\mathbf{0}1), (01\mathbf{0}, 1\mathbf{0}0), (01\mathbf{0}, 1\mathbf{0}1)\}$. Then, it is easy to see that $q_1 = (x_3, x_4)$ and $F = \{(0, 0)\}$.

**Håstad's Inner Verifier.** In order to overcome this difficulty, the inner verifier asks $P_2$ two questions. It picks a random set $G$ from the domain $R_X$ and asks $P_2$ if $a_2$ belongs to $G$ and if $a_2$ belongs to the exclusive-or of $G$ and $F'$ denoted $G \oplus F'$. The two sets now appear random to $P_2$ and do not enable it to infer $F$. In terms of long codes, the test is as follows. Let $A$ be the long code of $a_1$ and $B$ be the long code of $a_2$. The inner verifier picks a random function $f : R_Y \rightarrow \{0, 1\}$ and a random function $g : R_X \rightarrow \{0, 1\}$. The function $f$ corresponds to our set $F$ above, and $g$ corresponds to set $G$. Note that the function $f \circ \pi$, where $(f \circ \pi)(x) = f(\pi(x))$ for $x \in R_X$ corresponds to the set $\pi^{-1}(F) = F'$. Let $h : R_X \rightarrow \{0, 1\}$ be a function such that $h = g \oplus (f \circ \pi)$. The verifier reads the bits $A(f)$, $B(g)$ and $B(h)$, and accepts if and only if,

$$A(f) \oplus B(g) \oplus B(h) = 0 \ \ ,$$

which is a linear 3-bit test. There is a crucial part of the test that we have omitted. The function $h$ is in fact defined as $h = g \oplus (f \circ \pi) \oplus \mu$ where $\mu : R_X \to \{0, 1\}$ is chosen by setting $\mu(x) = 1$ with probability $\epsilon$ and $\mu(x) = 0$ with probability $1 - \epsilon$ independantly for each $x \in R_X$. If $h$ is not defined with $\mu$, then it can be shown (see for example [33, p. 33]) the test would accept with probability 1 even if $B$ is not a long code. This reduces the completeness of the verifier to $1 - \epsilon$, but this is all that we can hope for since perfect completeness would have implied that $P = NP$. Recall the relationship between PCP's and CSP's given in Sec. 2.3.1. Håstad's PCP implies that it is NP-hard to determine if the maximum fraction of clauses that can be satisfied in a Max-3Lin-2 instance is at least $1 - \epsilon$ or at most $1/2 + \eta$ for all $\epsilon, \eta > 0$. If we had perfect completeness, then the set of linear equations in the resulting Max-3Lin-2 instance could be solvable and using Gaussian Elimination we can determine in polynomial time if a system of linear equations over a field ($\mathbb{Z}_2$ in this case) is solvable.

## 3.4 Unique Games and Khot's Conjecture

A 2P1R game is called *unique* (e.g. see [36, 17]) if the answer of one prover uniquely determines the answer of the other prover *and vice versa*. Suppose that the Raz Verifier were a unique 2P1R game with almost perfect completeness and arbitrarily low soundness. (We say *almost* perfect completeness because it is trivial to determine if a unique game has value 1 as shown in Thm. 3.7 below.) Having the uniqueness property means that that for every possible pair of questions $q_1$, $q_2$ to provers $P_1$ and $P_2$, there is a *bijection* $\pi_{q_1,q_2} : R \to R$ such that $V$ accepts if and only if the answers $P_1(q_1)$, $P_2(q_2)$ satisfy $\pi_{q_1,q_2}(P_2(q_2)) = P_1(q_1)$. Note that the two provers provide answers from the same domain $R$. Intuitively, having these bijections would eliminate the need for the inner verifier we describe above to make the third query since the pre-image of a random set under a bijection is simply a permutation of the set and is also random. The Unique Games conjecture stipulates the existence of such a powerful outer verifier that would allow us to construct boolean 2-query PCP's. We will see a 2-bit test based on our discussion in Sec. 4.2.1.

## 3.5 Label Cover

A 2P1R game with the property that the answer of the second prover uniquely determines the answer of the first prover is equivalent to a problem called Label Cover first defined in [45]. We will use the definition of a weighted Label Cover from [32].

**Definition 3.5.** *A weighted Label Cover $\mathcal{L}(G(Y, X), R_Y, R_X, \{\pi_{yx}\}, \{p_{yx}\})$ consists of a complete bipartite graph $G$ with bipartition $Y$, $X$. Each edge $(y, x)$ has a weight $p_{yx}$ with $\sum_{y,x} p_{yx} = 1$. Every vertex in $Y$ is supposed to get a label from $R_Y$, and every vertex in $X$ is supposed to get a label from $R_X$. With every edge $(y, x)$ there is an associated projection $\pi_{yx} : R_X \to R_Y$. The goal is to find a labeling of the vertices, that is find functions $L_Y : Y \to R_Y$ and $L_X : X \to R_X$, that maximizes the weight of satisfied edges. An edge $(y, x)$ is satisfied if $\pi_{yx}(L_X(x)) = L_Y(y)$. $OPT(\mathcal{L})$ is defined to be the maximum weight of edges satisfied by any labeling. A Label Cover is* unique *if $R_X = R_Y = R$ and every function $\pi_{yx} : R \to R$ is a bijection.*

It is clear how a label cover problem is the same as a 2P1R game. Let $Y$, $X$ be the sets of questions the verifier can ask the two provers, and $R_Y$, $R_X$, respectively, be the set of their possible answers. Hence, the Unique Games conjecture can be stated as follows:

**Conjecture 3.6 (Unique Games conjecture).** *[32] For arbitrarily small constants $\zeta, \delta > 0$, there exists a constant $k = k(\zeta, \delta)$ such that it is NP-hard to determine whether a unique Label Cover instance with label sets of size $k$ (i.e. $|R| = k$) has optimum at least $1 - \zeta$ or at most $\delta$.*

The following theorem shows why the completeness parameter of the UGC is not 1.

**Theorem 3.7.** *Deciding if a unique Label Cover has optimum equal to $1$ is in $P$.*

*Proof.* Given a unique Label Cover instance $\mathcal{L}$ defined as in Def. 3.5, the following simple algorithm finds a labeling that satisfies all the edges if one exists. First, we get rid of edges with weight $0$ as they do not contribute to the optimum. For every connected component of the resulting graph, we do the following. Mark all the component's vertices False. Pick an arbitrary vertex $x_0 \in X$ and assign it an arbitrary label $L(x_0)$ from $R$. Now do the following:

- For every labeled vertex $v$ marked False,
- If $v \in X$, then assign each unlabeled $y$ in the neighborhood of $v$ the label $\pi_{yv}(L(v))$. If some $y$ was already labeled with $L(y)$, then check if $L(y) = \pi_{yv}(L(v))$. Mark $v$ True if all tests pass; otherwise, start over with a different label for $x_0$.
- If $v \in Y$, then assign each unlabeled $x$ in the neighborhood of $v$ the label $(\pi_{vx})^{-1}(L(v))$. If some $x$ was already labeled with $L(x)$, then check if $L(x) = (\pi_{vx})^{-1}(L(v))$. Mark $v$ True if all tests pass; otherwise, start over with a different label for $x_0$.

If some label to $x_0$ causes all the vertices in the connected component to be marked True, then the component is satisfied. If we can satisfy all components then $OPT(\mathcal{L}) = 1$. Otherwise, there is no labeling that has value 1. □

# 4 Hardness Results based on the Unique Games Conjecture

This section is mainly dedicated to showing that Min-2SAT-Deletion is NP-hard to approximate to within any constant factor [32]. The proof of Min-2SAT-Deletion closely follows that in [32]. We include it here as it displays the powerful technique developed by Håstad ([44, 27]) of analyzing the tests of an inner verifier using Fourier analysis. We also present the inner verifier and the test used to prove that 3-uniform hypergraph 3-coloring is hard to approximate within any factor less than $\frac{9}{8}$. We omit the test's soundness anlysis, however. Finally, we cite other results announced in [32] and point out some exciting more recent results by Khot *et al.* [30], Chwala *et al.* [12] and Dinur *et al.* [15]. All these PCP constructions essentially start with the Unique Games conjecture as the outer verifier and construct suitable inner verifiers to prove the hardness of the considered problems.

## 4.1 Fourier Analysis

The soundness proof of the inner verifier for Min-2SAT-Deletion depends heavily on Fourier analysis. Let $\mathcal{F}$ be the family of functions $f : M \to \{1, -1\}$. For $\beta \subseteq M$, the basis functions $\chi_\beta : \mathcal{F} \to \{1, -1\}$ used to define the Fourier transforms are

$$\chi_\beta(f) = \prod_{y \in \beta} f(y) \ ,$$

Note that $\chi_\beta$ is the point-wise product of long codes, and when $|\beta| = 1$, $\chi_\beta$ is just the long code of the element in $\beta$.

**Proposition 4.1.** *[28] For any $\beta \subseteq M$,*

$$\sum_{f \in \mathcal{F}} \chi_\beta = \begin{cases} 2^{|M|} & \textit{if } \beta = \emptyset \\ 0 & \textit{otherwise} \end{cases}$$

For functions $A, B$ mapping $\mathcal{F}$ to $\Re$ define their inner product as,

$$\langle A, B \rangle = 2^{-|M|} \sum_{f \in \mathcal{F}} A(f)B(f) = E_f[A(f)B(f)] \ .$$

Under this inner product, the Fourier basis form a complete orthonormal system since their number is $2^{|M|}$ and for any $\alpha, \beta \subseteq M$,

$$\begin{aligned} \langle \chi_\alpha, \chi_\beta \rangle &= E_f[\chi_\alpha(f)\chi_\beta(f)] \\ &= E_f[\prod_{x \in \alpha} f(x) \prod_{y \in \beta} f(y)] \\ &= E_f[\prod_{x \in \alpha \Delta \beta} f(x)] \\ &= 1 \text{ if } \alpha = \beta \text{ and } 0 \text{ otherwise,} \end{aligned}$$

where $\Delta$ denotes the symmetric difference of two sets. The third equality follows from the fact that if $x \in \alpha \cap \beta$, then $f(x)f(x) = 1$, and the last equality follows from Prop. 4.1. Hence, any function $A : \mathcal{F} \rightarrow \Re$ can be written as a linear combination of the basis functions,

$$A(f) = \sum_{\beta \subseteq M} \hat{A}_\beta \chi_\beta(f) \ , \tag{1}$$

where $\hat{A}_\beta = \langle A, \chi_\beta \rangle = E_f[A(f)\chi_\beta(f)]$. Equation (1) is the Fourier inversion formula, and $\hat{A}_\beta$ is called the Fourier coefficient of $A$ *at set* $\beta$.

**Theorem 4.2 (Parseval's identity).** *For any function $A : \mathcal{F} \rightarrow \Re$,*

$$\sum_{\beta \subseteq M} \hat{A}_\beta^2 = 2^{-|M|} \sum_{f \in \mathcal{F}} A^2(f) \ .$$

A proof of Parseval's identity can be found in [28] for example. Specifically, when $A$ has range $\{1, -1\}$, i.e. $A : \mathcal{F} \rightarrow \{1, -1\}$, Parseval's identity says that $\sum_{\beta \subseteq M} \hat{A}_\beta^2 = 1$. Furthermore, $\hat{A}_\beta$ is a measure of the correlation of $A$ with $\chi_\beta$. For any $\beta \subseteq M$ and $f \in \mathcal{F}$, let $1_{\{A=\chi_\beta\}}(f)$ indicate if $A(f) = \chi_\beta(f)$. Clearly, for any $f \in \mathcal{F}$, $1_{\{A=\chi_\beta\}}(f) = (A(f)\chi_\beta(f) + 1)/2$. Taking expectations we have,

$$\Pr_f[A(f) = \chi_\beta(f)] = \frac{E_f[A(f)\chi_\beta(f)] + 1}{2} = \frac{\hat{A}_\beta + 1}{2} \ . \tag{2}$$

This implies that if $A$ is the long code of some $x \in M$, then $\hat{A}_{\{x\}} = 1$ and by Parseval's identity all other Fourier coefficients are $0$.

The next lemma shows the effect of folding (see Def. 3.4) on the Fourier coefficients of a long code.

**Lemma 4.3.** *[44] If $A$ is folded, then for all $\beta \in M$, if $\hat{A}_\beta \neq 0$ then $|\beta|$ is odd (and in particular $\beta$ is not empty).*

Let $\pi : M \rightarrow M$ be a permutation. The following proposition relates the Fourier basis function of $f \circ \pi$ to that of $f$.

**Proposition 4.4.** *[32] $\chi_\beta(f \circ \pi) = \chi_{\pi(\beta)}(f)$.*

*Proof.* Since $\pi$ is a bijection, we have

$$\chi_\beta(f \circ \pi) = \prod_{x \in \beta} f(\pi(x)) = \prod_{y \in \pi(\beta)} f(y) = \chi_{\pi(\beta)}(f) \ .$$

□

## 4.2 Hardness of Approximating Min-2SAT-Deletion

In this section we outline the proof of the following theorem.

**Theorem 4.5.** *[32] The Unique Games Conjecture implies that for every $\frac{1}{2} < t < 1$ and for any sufficiently small constant $\epsilon > 0$, it is NP-hard to distinguish between the instances of Min-2Lin-2 where the fraction of satisfied equations is at least $1 - \epsilon$ and those where it is at most $1 - \epsilon^t$.*

Theorem 4.5 implies the same gap for Max-2SAT using the following simple reduction. We transform an equation of the form $x + y \equiv_2 0$ to the two clauses $x \vee \bar{y}$ and $\bar{x} \vee y$, and we transform equations of the form $x + y \equiv_2 1$ to $x \vee y$ and $\bar{x} \vee \bar{y}$. If an equation is satisfied, then the two corresponding clauses are satisfied; otherwise, exactly one clause is not satisfied. Hence, if there are $n$ equations in a Max-2Lin-2 instance and $\epsilon n$ are not satisfied, there will be $2n$ clauses in the constructed Max-2SAT instance and $\epsilon/2$ fraction will not be satisfied. It immediately follows that it is NP-hard to distinguish between the instances of Min-2SAT-Deletion where the fraction of unsatisfied clauses is at most $\epsilon$ and those where it is at least $\epsilon^t$ for any $\frac{1}{2} < t < 1$. Hence, Min-2SAT-Deletion cannot be approximated within any constant factor.

**Related Algorithmic Results.** The gap of $(1 - \epsilon, 1 - \epsilon^{\frac{1}{2}+o(1)})$ for Min-2SAT-Deletion is tight since, on an instance whose optimum is $1 - \epsilon$, the algorithm of Goemans and Williamson [25] produces a solution with value $1 - O(\sqrt{\epsilon})$. Zwick's algorithm [48] for Max-2SAT finds a $(1 - O(\epsilon^{1/3}))$-satisfying assignment when given a $(1 - \epsilon)$-satisfiable 2CNF formula. And Agrawala *et al.* [1] recently gave an $O(\sqrt{\log n})$-approximation algorithm for Min-2SAT-Deletion.

### 4.2.1 The PCP

The PCP we construct will be composed from an outer verifier and an inner verifier. The unique Label Cover instance $\mathcal{L}(G(Y, X), R, \{\pi_{yx}\}, \{p_{yx}\})$ guaranteed by Conjecture 3.6 serves as our PCP outer verifier. The inner verifier expects the proof to contain the long codes of the labels applied to every $y \in Y$ and $x \in X$. The long codes are assumed to be folded.

The inner verifier will pick an edge $(y, x)$ and check if labels of $y$ and $x$ satisfy $\pi_{yx}$. Let $p_y = \sum_{x \in X} p_{yx}$ and define $q_y : X \rightarrow [0, 1]$ as $q_y(x) = p_{yx}/p_y$. The verifier will pick an edge by first picking a vertex $y \in Y$ with probability $p_y$, and then picking a vertex $x \in X$ with probability $q_y(x)$. That is, it will choose $x$ conditioned on the fact that it already chose $y$. The full test is as follows:

1. Pick $y \in Y$ with probability $p_y$. Let $A$ be the supposed long code of the label of $y$. Recall that $A$ is indexed by all functions $h : R \to \{1, -1\}$.

2. Pick a random function $f : R \to \{1, -1\}$.

3. Pick a function $\mu : R \to \{1, -1\}$ by defining independently for each label $a \in R$

$$\mu(a) = \begin{cases} 1 & \text{with probability } 1 - \epsilon \\ -1 & \text{with probability } \epsilon \end{cases}$$

4. With probability $\frac{1}{2}$ select one of the following actions:

   (a) (Codeword test) Accept if and only if $A(f) = A(f\mu)$

   (b) (Consistency test) Pick a vertex $x \in X$ with probability $q_y(x)$. Let $B$ be the supposed long code of the label of $x$, and let $\pi = \pi_{yx}$.
   Accept if and only if $A(f) = B(f \circ \pi)$.

It would seem at first glance that the "perturbation" function should be added to the consistency test, i.e. $A(f) = B((f \circ \pi)\mu)$, for otherwise the test can always be made to accept with probability 1 even if $B$ is not a long code. Consider the following example.[1] Let $A$ be the long code of some $a \in R$, and let $B = \chi_\beta$ for some $\beta \subseteq R$ where $|\beta|$ is odd, $|\beta| > 1$, and $\pi(b) = a$ for all $b \in \beta$. Then,

$$A(f) = f(a) = \prod_{b \in \beta} f(\pi(b)) = \chi_\beta(f \circ \pi) = B(f \circ \pi) \ .$$

The second equality follows from the fact that the cardinality of $\beta$ is odd. Nonetheless, the function $\mu$ ensures that the verifier rejects codes whose Fourier spectrum depends significantly on sets of large size, and we will see in the soundness analysis that it suffices to include it only in the codeword test. For intuition, let $A$, for example, be the product of long codes, i.e. let $A = \chi_\alpha$ for some $\alpha \subseteq R$ with $|\alpha| > 1$, then the probability the codeword test accepts is

$$\frac{E_{f,\mu}[A(f)A(f\mu)] + 1}{2} = \frac{E_{f,\mu}[\prod_{a \in \alpha} f^2(a)\mu(a)]}{2} = \frac{(1 - 2\epsilon)^{|\alpha|} + 1}{2} \ ,$$

which decreases as $|\alpha|$ increases. Recall that if $A$ is the long code of some $a \in R$, then all its Fourier coefficients are 0 except for $\hat{A}_{\{a\}}$ which is 1

### 4.2.2 Completeness

The completeness of the verifier is $1 - \frac{\epsilon + \zeta}{2}$. In a correct proof, $A$ and $B$ are the long codes of some $a, b \in R$ where $a$ (resp. $b$) is the label of the vertex $y \in Y$ (resp. $x \in X$) that we picked. The verifier selects a test with probability $1/2$. Now the codeword test fails when $\mu(a) = -1$, which happens with probability $\epsilon$. The consistency test, on the other hand, fails when we pick an unsatisfied edge in the unique Label Cover instance $\mathcal{L}$, which happens with probability $\zeta$. When we pick a satisfied edge, the consistency test succeeds since $f(a) = A(f) = B(f \circ \pi) = f(\pi(b)) = f(a)$. Note that by the Unique Games conjecture, we can assume $\zeta$ to be arbitrarily small. The completeness of the verifier follows.

[1] This example is given in [33] to show how, without $\mu$, Håstad's 3-bit can always fail.

### 4.2.3 Soundness

We will show that the soundness of the verifier is at most $1 - \frac{1}{8}c_t\epsilon^t$ for any $\frac{1}{2} < t < 1$ where $c_t$ is a constant dependent on $t$ (from Thm. 4.6), and where $\epsilon$ is that of the "perturbation" function $\mu$. We will use Fourier analysis to show that if the verifier accepts with probability greater than $1 - \frac{1}{8}c_t\epsilon^t$, then we can extract a (probabilistic) labeling of reasonable weight using the Fourier coefficients of the codes provided. Since OPT($\mathcal{L}$) $\leq \delta$, this would lead to a contradiction provided we choose $\delta$ to be small enough. The analysis uses the following result of Bourgain [11] as stated in [32],

**Theorem 4.6 (Bourgain).** *Let $A$ be any boolean function (for instance a supposed long code) and $k > 0$ an integer. Then for every $\frac{1}{2} < t < 1$, there exists a constant $c_t > 0$ such that,*

$$\textit{If} \sum_{\alpha\,:\,|\alpha|>k} \hat{A}_\alpha^2 < c_t k^{-t} \quad \textit{then} \sum_{\alpha\,:\,|\bar{A}_\alpha|\leq\frac{1}{10}4^{-k^2}} \hat{A}_\alpha^2 < \frac{1}{100} \ .$$

The probability of acceptance of the inner verifier is,

$$\Pr[\text{Accept}] = \frac{1}{2}\left[E_{y,f,\mu}\left[\frac{1 + A(f)A(f\mu)}{2}\right] + E_{y,x,f}\left[\frac{1 + A(f)B(f\circ\pi)}{2}\right]\right] \ .$$

This can be shown, for example, by the indicator method as we did for Eq. (2) in Sec. 4.1.

Using the Fourier transform we have,

$$E_{f,\mu}[A(f)A(f\mu)] = E_{f,\mu}[\sum_{\alpha_1,\alpha_2} \hat{A}_{\alpha_1}\hat{A}_{\alpha_2}\chi_{\alpha_1}(f)\chi_{\alpha_2}(f)\chi_{\alpha_2}(\mu)]; \text{ and,} \tag{3}$$

$$E_f[A(f)B(f\circ\pi)] = E_f[\sum_{\alpha,\beta} \hat{A}_\alpha\hat{B}_\beta\chi_\alpha(f)\chi_\beta(f\circ\pi)] \ . \tag{4}$$

Now, $\chi_{\alpha_1}(f)\chi_{\alpha_2}(f) = \chi_{\alpha_1\Delta\alpha_2}$ and as shown in Sec. 4.1 its expectation over $f$ is 1 if $\alpha_1\Delta\alpha_2 = \emptyset$ and 0 otherwise. Hence, (3) is non-zero only if $\alpha_1 = \alpha_2 = \alpha$. Since $E_\mu[\chi_\alpha(\mu)] = (1-2\epsilon)^{|\alpha|}$, we have that,

$$E_{f,\mu}[A(f)A(f\mu)] = \sum_\alpha \hat{A}_\alpha^2(1-2\epsilon)^{|\alpha|} \ . \tag{5}$$

Using Prop. 4.4, we similarly see that (4) is non-zero only if $\alpha = \pi(\beta)$. Since $\pi$ is a bijection, $\beta = \pi^{-1}(\alpha)$ and we have that,

$$E_f[A(f)B(f\circ\pi)] = \sum_\alpha \hat{A}_\alpha\hat{B}_{\pi^{-1}(\alpha)} \ . \tag{6}$$

The probability of acceptance becomes,

$$\begin{aligned}\Pr[\text{Accept}] &= \frac{1}{2} + \frac{1}{4}E_y\left[\sum_\alpha \hat{A}_\alpha^2(1-2\epsilon)^{|\alpha|} + \sum_\alpha \hat{A}_\alpha E_x\left[\hat{B}_{\pi^{-1}(\alpha)}\right]\right] \\ &= \frac{1}{2} + \frac{1}{4}E_y[D_y + C_y] \ .\end{aligned}$$

Suppose this probability is greater than $1 - \frac{1}{8}c_t\epsilon^t$ where $t$ and $c_t$ are from Bourgain's Theorem. Then we have $E_y[D_y + C_y] \geq 2 - \frac{1}{2}c_t\epsilon^t$, which implies by Markov's inequality that over the choice of $y$, with probability at least $\frac{1}{2}$, $D_y + C_y \geq 2 - c_t\epsilon^t$. Fix any such "good" $y$. By Parseval's identity, $D_y \leq 1$ and we have that $C_y \geq 1 - c_t\epsilon^t > \frac{1}{2}$ where the last inequality follows by choosing $\epsilon$ small enough. Similarly, by Cauchy-Schwarz and Parseval's identity $C_y \leq 1$ and we have that $D_y \geq 1 - c_t\epsilon^t$. This last fact combined with the term $(1-2\epsilon)^{|\alpha|}$ introduced by $\mu$ allows us to show that

$$\sum_{\alpha\,:\,|\alpha|>\epsilon^{-1}} \hat{A}_\alpha^2 < c_t\epsilon^t \ , \tag{7}$$

and applying Bourgain's Theorem, we get,

$$\sum_{\alpha\,:\,|\hat{A}_\alpha|\leq\frac{1}{10}4^{-\epsilon^{-2}}} \hat{A}_\alpha^2 < \frac{1}{100} \ . \tag{8}$$

Equation (7) says that for the given code its Fourier coefficients at sets of large size are insignificant, while Eq. (8) says that the code is determined by a few coefficients. Ideally, in a correct proof, a long code is determined by only one coefficient at a set of size one (see Sec. 4.1).

We summarize the rest of the argument. Call $\alpha$ "good" if $\alpha$ is nonempty, $|\alpha| \leq \epsilon^{-1}$, and $|\hat{A}_\alpha| \geq \frac{1}{10}4^{-\epsilon^{-2}}$. All other $\alpha \subseteq M$ are "bad". It is shown that the contribution of bad $\alpha$'s to the term $C_y$ introduced by the consistency test is small. First, if $\alpha = \emptyset$, then by lemma 4.3, $\hat{A}_\alpha = 0$. Next, if $|\alpha| > \epsilon^{-1}$, then (7) is used to show that $C_y < \sqrt{c_t\epsilon^t}$. Finally, if $|\hat{A}_\alpha| < \frac{1}{10}4^{-\epsilon^{-2}}$, then (8) is used to show that $C_y < 1/10$. Combined with the fact that $C_y > 1/2$, the above implies that when restricted to good $\alpha$'s, $C_y$ remains at least $1/4$.

Hence, if the acceptance probability of the inner verifier is greater than $1 - \frac{1}{8}c_t\epsilon^t$, the codes provided by the provers must be "close" to long codes in a sense that they are determined by a few coefficients at sets of small size, namely those coefficients with good $\alpha$'s. We will depend on those coefficients to define a (probabilistic) labeling for the edges in $\mathcal{L}$ of total weight $\Omega(\epsilon 4^{2\epsilon^{-2}})$. This will contradict the fact that OPT($\mathcal{L}$) $< \delta$ if $\delta$ is chosen to be sufficiently small implying that the acceptance probability of the verifier is at most $1 - \frac{1}{8}c_t\epsilon^t$. Note that by the Unique Games conjecture, we can assume $\delta$ to be arbitrarily small.

The labeling we define is as follows. For a good vertex $y \in Y$, pick $\alpha$ with probability $\hat{A}_\alpha^2$. Pick a random element of $\alpha$ and define it to be the label of $y$. For any $x \in X$, pick $\beta$ with probability $\hat{B}_\beta^2$. Pick a random element of $\beta$ and define it to be the label of $x$.

Now, let $(y, x)$ be an edge with a good $y$ and let $a$ and $b$ be the labels of $y$ and $x$ respectively. The probability that we pick a certain $\alpha_i$ and $\beta_i = \pi^{-1}(\alpha_i)$ is $\hat{A}_{\alpha_i}^2\hat{B}_{\pi^{-1}(\alpha_i)}^2$. Given this event, the probability that $\pi(b) = a$ is $1/|\alpha_i|$ since $a$ and $b$ are randomly chosen elements of $\alpha_i$ and $\beta_i$ respectively, and since $|\alpha_i| = |\beta_i|$. Therefore, with probability $\sum_\alpha \hat{A}_\alpha^2\hat{B}_{\pi^{-1}(\alpha)}^2\frac{1}{|\alpha|}$ the edge $(y, x)$ is satisfied. Let $p_x = \sum_{y\in Y} p_{yx}$, i.e. if an edge is picked with probability equal to its weight, $p_x$ is the probability that the right end point is $x$. The expected weight of satisfied edges is then,

$$\sum_{y,x} p_{yx} \sum_\alpha \hat{A}_\alpha^2\hat{B}_{\pi^{-1}(\alpha)}^2\frac{1}{|\alpha|} = E_x\left[\sum_\alpha \hat{A}_\alpha^2\hat{B}_{\pi^{-1}(\alpha)}^2\frac{1}{|\alpha|}\right] \geq \epsilon E_x\left[\sum_{\alpha \text{ good}} \hat{A}_\alpha^2\hat{B}_{\pi^{-1}(\alpha)}^2\right] \ . \tag{9}$$

Note that we are assuming that $y$ is good, which happens with probability at least $1/2$. The above expression is shown to be $\Omega(\epsilon 4^{2\epsilon^{-2}})$ by the properties of good $\alpha$'s and the fact that $C_y \geq 1/4$ even when restricted to good $\alpha$'s. This concludes the soundness analysis.

### 4.3 Hardness of Coloring $3$-uniform Hypergraphs with $3$ Colors

In this section we will outline the PCP used to prove the following theorem.

**Theorem 4.7.** *[32] The Unique Games conjecture implies that given a $3$-uniform hypergraph and $3$ colors, it is NP-hard to determine whether there exists a coloring of the vertices that correctly colors $1 - \epsilon$ fraction of the hyperedges or any coloring correctly colors at most $\frac{8}{9} + \epsilon$ fraction of the hyperedges, where $\epsilon > 0$ is an arbitrarily small constant.*

The problem of coloring a $q$-uniform hypergraph with $k$ colors can be thought of as a constraint satisfaction problem. The vertices of the graph are the variables of the CSP, and the edges are its constraints. Since each edge is a set of $q$ vertices, each constraint has exactly $q$ variables. The $k$ colors correspond to a domain of size $k$ from which we will assign values to the variables. A constraint is satisfied if not all $q$ variables in the constraint have the same value. Hence, satisfying a constraint is equivalent to correctly coloring the corresponding edge. The optimum of the CSP is the maximum fraction of constraints that can be satisfied by any assignment. As in [31], we will call this CSP NAE$_{q,k}$.

**Definition 4.8.** *[31] The problem NAE$_{q,k}$ is said to have the* Random Threshold Property *if it is NP-hard to do strictly better than a random assignment. That is, it is NP-hard to distinguish whether the optimum is at least $1 - \epsilon$ or at most $1 - \frac{1}{k^{q-1}} + \epsilon$ for arbitrarily small $\epsilon > 0$.*

Hence, Conjecture. 4.7 asserts that NAE$_{3,3}$ has the random threshold property. In [31], Khot proves this result (with gap $(1, \frac{8}{9} + \epsilon)$) *unconditionally*. In fact, it is shown that NAE$_{3,k}$ for every $k \geq 3$ has the random threshold property. Recall that in a unique Label Cover instance, the maps $\pi_{yx} : R \rightarrow R$ are bijections. The main technique in [31] is to obtain a weaker notion of this property. Khot (see also [33, Thm. 4.2.2]) shows the hardness of Label Cover when the maps $\pi_{yx} : R_X \rightarrow R_Y$ satisfy the following *smoothness* property: For every $x \in X$ and every pair of distinct labels $a, a' \in R_X$,

$$\Pr_y[\pi_{yx}(a) \neq \pi_{yx}(a')] \approx 1 \ . \tag{10}$$

This property is combined with the *multi-layered* version of Label Cover from [14] to prove the result. Note that for the only other case, namely NAE$_{3,2}$, Zwick's algorithm [47] performs strictly better than a random assignment.

### 4.3.1 The PCP

The unique Label Cover instance $\mathcal{L}(G(Y,X), R, \{\pi_{yx}\}, \{p_{yx}\})$ guaranteed by the Unique Games conjecture again serves as the outer verifier in the PCP we construct. Hence, the construction of the PCP again reduces to the construction of a suitable inner verifier. The inner verifier expects the proof to contain the long codes of the labels of all the vertices in $\mathcal{L}$. Let $\mathcal{F}_R^3$ be the family of functions $f : R \rightarrow \{1, \omega, \omega^2\}$. The long code $A$ of a label $a \in R$ is indexed by all functions $f \in \mathcal{F}_M^3$ and is defined as $A(f) = f(a)$. The verifier will read three symbols from the proof and accept if and only if the three symbols are not all equal. The verifier's test is as follows:

1. Pick $y \in Y$ with probability $p_y$.
2. Pick three vertices $x_1$, $x_2$ and $x_3$ with probabilities $q_y(x_1)$, $q_y(x_2)$ and $q_y(x_3)$ respectively. Let $A$, $B$, and $C$ be the supposed long codes of $x_1$, $x_2$ and $x_3$ respectively.
3. Pick two random functions $f, g \in \mathcal{F}_M^3$. Let $h = \bar{f} \cdot \bar{g}$ where $\bar{f}(a)$ is the complex conjugate of $f(a)$.
4. Pick a function $\mu : R \rightarrow \{\omega, \omega^2\}$ by defining for each $a \in R$, $\mu(a) = \omega$ with probability $\frac{1}{2}$ and $\mu(a) = \omega^2$ with probability $\frac{1}{2}$.

5. Accept if and only if not all of $A(f \circ \pi_{yx_1})$, $B(g \circ \pi_{yx_2})$, and $C((h \circ \pi_{yx_3}) \cdot \mu)$ are equal.

The completeness of the verifier is $1 - 3\zeta$. The verifier picks 3 edges from the unique Label Cover instance and each can be unsatisfied with probability $\zeta$. Suppose all edges are satisfied. In a correct proof $A$, $B$ and $C$ are the long codes of some $a, b, c \in R$ respectively where $a, b$ and $c$ are the labels of $x_1, x_2$ and $x_3$ respectively. Furthermore, $\pi_{yx_1}(a) = \pi_{yx_2}(b) = \pi_{yx_3}(c) = d$ for some $d \in R$ ($d$ is the label of $y$). Suppose $A(f \circ \pi_{yx_1}) = f(d) = g(d) = B(g \circ \pi_{yx_2})$. Then $C((h \circ \pi_{yx_3}) \cdot \mu) = \overline{f(d)g(d)}\mu(c) = \overline{f(d)^2}\mu(c) = f(d)\mu(c) \neq f(d)$ since $\mu(c) \in \{\omega, \omega^2\}$. Hence, if the edges are all satisfied, not all three symbols read can be equal.

The soundness of the test is shown to be $\frac{8}{9} + \epsilon$ for arbitrarily small $\epsilon > 0$ by showing that if the inner verifier accepts the not-all-equal test with a probability bounded away from $0$, then it is possible to define a labeling for the vertices of $\mathcal{L}$ of reasonable weight. Since OPT$(\mathcal{L}) \leq \delta$, this leads to a contradiction provided we choose $\delta$ small enough.

## 4.4 Other Hardness Results

**Minimum Multicut and Sparsest Cut.** Chawla *et al.* [12] note that, as implied by the approximation-preserving reduction from Min-2SAT-Deletion to Minimum Multicut of [35], Thm. 4.5 also shows that Minimum Multicut is hard to approximate within any constant factor. Recall that Minimum Multicut is the problem of given a graph $G$ and $k$ pairs of vertices $\{(s_i, t_i)\}_{i=1}^{k}$, find a minimum-size subset of edges whose removal disconnects every $(s_i, t_i)$ pair. Sparsest Cut is the problem of given a graph $G$, find the cut with smallest edge expansion where the edge expansion of a cut $(S, \bar{S})$ is defined as $|E(S, \bar{S})| / \min\{S, \bar{S}\}$. Using a stronger version of the Unique Games conjecture, [12, Cor. 1.4] shows for some fixed constant $c > 0$ that it is NP-hard to approximate Min-2SAT-Deletion, Minimum Multicut and Sparsest Cut to within factor $c \log\log n$. The stronger version of the conjecture requires that the parameters $\zeta, \delta$ and the answer domain $k = k(\zeta, \delta)$ satisfy $\max\{\zeta, \delta\} \leq 1/(\log n)^{\Omega(1)}$ and $k = O(\log n)$. (We will show in Sec. 6 that for the UGC to hold, $k$ must be at least $\max\{\frac{1}{\zeta^{(1/10)}}, \frac{1}{\delta}\}$, which does not exclude the parameters required by the stronger version).

On the algorithmic side, the algorithm of [24] approximates Minimum Multicut to within a factor of $O(\log k)$, and [4] give a $O(\sqrt{\log n})$-approximation algorithm for Sparsest Cut.

**Max-Cut.** In [30], Khot *et al.* show that assuming the Unique Games conjecture, it is NP-hard to approximate Max-Cut to within any factor greater than $\frac{3}{4} + \frac{1}{2\pi}$ ($\approx .909155$). [2] If they further assume a conjecture they refer to as the *Majority is Stablest* conjecture together with the UGC, then they show that it is NP-hard to approximate Max-Cut to within a factor $\alpha_{GW} + \epsilon$ for all $\epsilon > 0$. Here, $\alpha_{GW} = \min_{0 \leq \theta \leq \pi} \frac{\theta/\pi}{(1-\cos\theta)/2}$, which is exactly the approximation factor of the Goemans-Williamson algorithm [25]. The currently best known gap for Max-Cut is $(1, \frac{16}{17} + \epsilon)$ due to Håstad [44].

A generalization of the Majority is Stablest conjecture was recently confirmed by Mossel *et al.* [39, Thm. 4.4]. Besides implying that based on the UGC the Goemans-Willimason .878-approximation algorithm is the best possible for Max-Cut, their theorem also implies that based on the UGC, Max-2Lin-2 and Max-2SAT have a gap of $(1 - \epsilon, 1 - O(\sqrt{\epsilon}))$. Furthermore,

**Theorem 4.9.** *[39, Thm. 2.12] UGC implies that for each $\epsilon > 0$ there exists $q = q(\epsilon)$ such that given an instance of Max-2Lin-q it is NP-hard to distinguish between the case where it is $(1 - \epsilon)$-satisfiable and $\epsilon$-hardness. Indeed, this statement is equivalent to UGC.*

[2] For consistency with the cited work, the approximation factors are $< 1$.

**Approximate Coloring.** The Approximate Coloring problem (cf. Sec. 2.1) can be stated as follows: Given a graph $G$ and a pair $(q, Q)$, decide if the chromatic number of $G$, $\chi(G) \leq q$ or $\chi(G) \geq Q$. When $q = 3$, the best known polynomial time algorithm solves the problem for $Q = \tilde{O}(n^{3/14})$ where $n$ is the number of vertices of the given graph [10]. The strongest hardness result, on the other hand, is due to Khanna *et al.* [29] and shows that the problem is NP-hard for $Q = 5$. Most recently, assuming a variant of the Unique Games conjecture (called the $\ltimes$ Conjecture), Dinur *et al.* [15] show that the problem is hard for any constant $Q > 3$. They also show that for any $q \geq 4$ and any constant $Q > 0$, the problem is hard based on Khot's [32] 2-to-1 conjecture.

We end this section with Khot's $d$-to-1 conjecture. A 2P1R game has the "$d$-to-1" property if the answer of the second prover uniquely determines the answer of the first prover and for every answer of the first prover, there are at most $d$ answers for the second prover that would make the verifier accept.

**Conjecture 4.10 ($d$-to-1).** *[32] Let $\delta > 0$ be an arbitrarily small constant, then there exists a constant $k = k(\delta)$ such that it is NP-hard to determine whether a 2P1R game with $d$-to-1 property and answers from a domain of size at most $k$ has value $1$ or at most $\delta$.*

Khot states that the 2-to-1 conjecture implies a $\sqrt{2} - \epsilon$ hardness for Vertex Cover. In [34], however, Khot and Regev show that the Unique Games conjecture implies vertex cover is hard to approximate to within $2 - \epsilon$. This result is the topic of Sec. 5.

# 5 Hardness of Approximating Vertex Cover

In this section, we present the following result due to Khot and Regev [34]:

**Theorem 5.1.** *[34] Assuming the Unique Games conjecture, E$k$-Vertex-Cover is NP-hard to approximate within factor $k - \epsilon$ for every $k \geq 2$ and arbitrarily small $\epsilon > 0$.*

In [14], Dinur *et al.* show an inapproximability factor of $\lfloor \frac{k}{2} \rfloor - \epsilon$ for E$k$-Vertex-Cover based on the Raz Verifier using a construction similar to that of [34]. We will compare the two techniques and show how the Unique Games conjecture is used to prove the stronger result.

One way of reducing the Raz Verifier to E$k$-Vertex-Cover is by introducing a block of vertices for each variable in $X$ and $Y$ (representing their long codes) and emulating each constraint $\pi_{yx}$ by a set of hyperedges consisting of both $x$-vertices (vertices introduced by $X$) and $y$-vertices (vertices introduced by $Y$). However, this reduction has a basic "bipartiteness" flaw: The underlying constraint graph being bipartite (i.e. the Label Cover instance) has a vertex cover whose size is at most half the number vertices. This translates to a vertex cover in the hypergraph regardless of whether the PCP instance used to construct the graph is satisfiable or not.

Dinur *et al.* [14] overcome this bipartiteness flaw by introducing a multi-layered PCP. Instead of two "layers" $X$ and $Y$ the multi-layered PCP has $\ell$ layers $X_1, X_2, \ldots, X_\ell$. Each pair of layers represents an instance of the Raz Verifier. In this PCP, it is NP-hard to distinguish between the case where there exists an assignment that satisfies all the constraints, and the case where for every pair of layers $X_i$ and $X_j$ there is no assignment that satisfies an $\epsilon$ fraction of the constraints between $X_i$ and $X_j$. Using this multilayered PCP and the *biased* long code introduced in [16], Dinur *et al.* show that E$k$-Vertex-Cover is NP-hard to approximate within a factor of $(k - 1 - \epsilon)$ for all $k \geq 3$ where $\epsilon > 0$ is any arbitrary constant. We introduce the biased long code in the next section as we will also be using it for our construction.

Another way to reduce the Raz Verifier to E$k$-Vertex-Cover is to construct the hypergraph only from the variables in $X$. We introduce a $2^{|R_X|}$ block of vertices for each $x \in X$ representing its long code, and hyperedges connect vertices from $x_1$'s block to vertices from $x_2$'s block only if there exists $y \in Y$ such that $\pi_{yx_1}$ and $\pi_{yx_2}$ are constraints in the system. This construction is used both in [14] for showing the $\lfloor \frac{k}{2} \rfloor - \epsilon$ result, and in [34] for showing the $k - \epsilon$ result but starting from the Unique Games conjecture instead of the Raz Verifier. A stronger form of the UGC called the *Strong Unique Label Cover*, or Strong LC for short, is needed in [34].

A Strong LC $\mathcal{L}(Y, X, E, R, \Pi)$ is defined as follows. We are given a bipartite graph $(Y, X, E)$ possibly with parallel edges in which all the degrees of the vertices in $X$ are equal to some constant $d$. Each vertex in $Y$ and $X$ is supposed to get a label from $R$. With every edge $(y, x) \in E$ there is an associated bijection $\pi_{yx} : R \to R$, $\pi_{yx} \in \Pi$. An assignment of labels to vertices $L : Y \cup X \to R$ is said to satisfy edge $(y, x) \in E$ if $\pi_{yx}(L(x)) = L(y)$.

**Theorem 5.2.** *[34, Thm. 3.2] Assuming the Unique Games conjecture, for any $\zeta, \gamma > 0$ there exists constants $k$, $d$ such that the following is NP-hard. Given a Strong LC $\mathcal{L}(Y, X, E, R, \Pi)$ with $|R| = k$ and the degree of every vertex in $X$ is $d$, distinguish between the case where there exists as assignment in which at least $1 - \zeta$ fraction of the $X$ vertices have all their edges satisfied and the case where no assignment satisfies more than $\gamma$ of the edges.*

This form of the Unique Games conjecture shares two key properties with the Raz Verifier that the employed techniques use to prove hardness in our hypergraph construction:

1. (Regularity) The layer in the underlying constraint graph used to create the vertices of the hypergraph is regular.

2. (Strong Completeness) The Raz Verifier has perfect completeness, i.e. the provers have a strategy such that with probability 1, after fixing the question to the second prover, the verifier accepts for *every* question to the first prover. In the Strong LC, this happens with probability very close to 1. (Recall that the original form of the UGC simply states that the provers have a strategy that convinces the verifier with probability very close to 1).

To reiterate, in [14], the hardness achieved is $\lfloor \frac{k}{2} \rfloor - \epsilon$, and in [34] it is $k - \epsilon$. Both results use the same type of hypergraph construction, which we briefly mentioned but will shortly formalize, for constructing a hard instance of E$k$-Vertex-Cover. Their underlying constraint graphs are made essentially similar by Thm. 5.2. They differ in their proof techniques and in that the constraints are projections in one and bijections in the other. We will investigate why the proofs fail to give the tight hardness result when the constraints are projections. Since the two constructions coincide exactly when $k = 2$, we will proceed by constructing a graph from the Raz Verifier first, and switching to the Unique Games conjecture to show the strong hardness result.

## 5.1 Preliminaries

The following definitions are from [14] and [34]. We include them here for easy reference.

**Definition 5.3.** *For a bias parameter $0 < p < 1$ and a ground set $R$, the weight of a set $F \subseteq R$ is*

$$\mu_p^R(F) \overset{def}{=} p^{|F|} \cdot (1 - p)^{|R \setminus F|} \ .$$

Hence, the weight of a subset is the probability of obtaining this subset when each element in $R$ is picked with probability $p$. The weight of a family of subsets $\mathcal{F} \subseteq 2^R$ is,

$$\mu_p^R(\mathcal{F}) \overset{def}{=} \sum_{F \in \mathcal{F}} \mu_p^R(F) \ .$$

Note that since $\mu_p^R(2^R) = 1$ the bias parameter defines a distribution on $2^R$. We denote this distribution $\mu_p^R$.

We will use a combinatorial view of the long code, and we define the biased long code next.

**Definition 5.4 ($p$-biased Long Code).** *Let $p < 0 < 1$ be a bias parameter. A $p$-biased long code over a domain $R$ for an element $\sigma \in R$ is a $2^{|R|}$ bit string indexed by all subsets $F \subseteq R$. The bit indexed by $F$ has a weight $\mu_p^R(F)$ attached to it and its value is $1$ if $\sigma \in F$ and $0$ otherwise.*

The only difference between this definition and Def. 3.3 is the weight attached to each bit in the long code.

**Definition 5.5 (Influence).** *For a family $\mathcal{F} \subseteq 2^R$, an element $\sigma \in R$, and a bias parameter $p$, the influence of the element on the family is defined as,*

$$\textit{Influence}_p^R(\mathcal{F}, \sigma) \overset{def}{=} \Pr_{F \in \mu_p^R}[\textit{exactly one of } \ F \cup \{\sigma\}, F \setminus \{\sigma\} \ \textit{ is in } \mathcal{F}] \ .$$

The *average sensitivity* of a family is defined as the sum of influences of all the elements:

$$\mathrm{as}_p^R(\mathcal{F}) \overset{def}{=} \sum_{\sigma \in R} \mathrm{Influence}_p^R(\mathcal{F}, \sigma) \ .$$

**Definition 5.6 (Monotone Family).** *A family $\mathcal{F} \subseteq 2^R$ is called monotone if $F \in \mathcal{F}$ and $F \subseteq F'$ implies $F' \in \mathcal{F}$.*

**Definition 5.7 (Core-Family).** *A family $\mathcal{F} \subseteq 2^R$ is called a core-family with a core $C \subseteq R$ if there exists a family $\mathcal{F}_C \subseteq 2^C$ such that,*

$$\forall F \in 2^M, \quad F \in \mathcal{F} \ \textit{ if and only if } \ F \cap C \in \mathcal{F}_C \ .$$

Finally, we define the notion of an $s$-wise $t$-intersecting family. Denote $[n] = \{1, 2, \ldots, n\}$ and $2^{[n]} = \{F : F \subseteq [n]\}$.

**Definition 5.8.** *A family $\mathcal{F} \subseteq 2^{[n]}$ is called $s$-wise $t$-intersecting if for every $s$ sets $F_1, F_2, \ldots, F_s \in \mathcal{F}$, we have $|F_1 \cap F_2 \cap \cdots \cap F_s| \geq t$.*

## 5.2 The Construction

We start with an instance of the Raz Verifier, or the equivalent Label Cover instance, call it $\mathcal{L}$, where $Y$ (resp. $X$) corresponds to the set of questions the verifier can ask prover $P_1$ (resp. $P_2$), and $R_Y$ (resp. $R_X$) corresponds to the set of its possible answers. Let $\{\pi_{yx}\}$ be the set of projection constraints. We will construct a weighted graph $G = (V, E)$ as follows. The set of vertices of the graph will correspond to the bits of the long codes of the labels assigned to the vertices of $X$. Namely, the set of vertices is defined to be,

$$V \overset{def}{=} X \times 2^{R_X} \ .$$

For each $x \in X$, we define the *block of vertex* $x$, $B[x]$, as the set of vertices corresponding to $x$. That is,

$$B[x] \overset{def}{=} \{\langle x, F\rangle : F \subseteq R_X\} \ .$$

The weight of each vertex is

$$\text{weight}(\langle x, F\rangle) \overset{def}{=} \frac{1}{|X|}\mu_p^{R_X}(F) \ ,$$

where $0 < p < 1$ is a bias parameter.

The edges are defined as follows. For every pair of constraints $\pi_{yx_1}$ and $\pi_{yx_2}$ sharing a common variable $y \in Y$, we add the following edges between vertices in $B[x_1]$ and $B[x_2]$,

$$\{\{\langle x_1, F\rangle, \langle x_2, G\rangle\} : \pi_{yx_1}(F) \cap \pi_{yx_2}(G) = \emptyset\} \ .$$

That is, there is no $r_1 \in F$ and $r_2 \in G$ such that $\pi_{yx_1}(r_1) = \pi_{yx_2}(r_2)$. The intuition behind the construction of the edges comes from the completeness proof. Essentially, when the Label Cover instance is satisfiable, we want $G$ to have a large independent set.

## 5.3 Completeness

Assume $\mathcal{L}$ has an assignment $A$ that satisfies all the constraints. The following is an independent set in $G$:

$$\text{IS} = \{\langle x, F\rangle : x \in X, A(x) \in F\} \ .$$

That is, the vertices of $G$ corresponding to the '1' bits of the long codes of the labels assigned to the vertices of $X$ form an independent set. Consider an edge $e = \{\langle x_1, F\rangle, \langle x_2, G\rangle\}$ and suppose both $\langle x_1, F\rangle$ and $\langle x_2, G\rangle$ are in IS. Then we know that $A(x_1) \in F$ and $A(x_2) \in G$. But since edges $(y, x_1)$ and $(y, x_2)$ for some $y \in Y$ in the label cover instance are satisfied, we have $\pi_{yx_1}(A(x_1)) = A(y)$ and $\pi_{yx_2}(A(x_2)) = A(y)$. Hence, $A(y) \in \pi_{yx_1}(F) \cap \pi_{yx_2}(G)$ and we reach a contradiction by recalling the construction of the edges. Now,

$$\text{weight}(\text{IS}) = \sum_{x \in X} \text{weight}(\text{IS} \cap B[x]) = \frac{1}{|X|}\sum_{x \in X} \Pr_{D \in \mu_p^{R_X}}\left[D \in \{F \in 2^{R_X} : A(x) \in F\}\right] = p \ .$$

The desired completeness is achieved by setting $p = \frac{1}{2} - \epsilon$ where $\epsilon$ is arbitrarily small. Now if starting from the Raz Verifier, we could show in the soundness case that no independent set in $G$ has weight $\delta$ where $\delta$ is arbitrarily small, then we would obtain the desired hardness result for vertex cover. This is because we would have shown that we cannot differentiate between graphs whose minimum vertex cover has size $\leq \frac{1}{2} - \epsilon$ and graphs whose minimum vertex cover has size $\geq 1 - \delta$.

## 5.4 Soundness

Assume that there is no assignment that satisfies even a $\gamma$ fraction of the constraints of our Label Cover instance $\mathcal{L}$. Following the usual paradigm, we will assume towards contradiction that the graph $G$ contains an independent set IS of size $\delta$. We would like to show that in such a case, it is possible to "decode" the long codes and define a labeling that satisfies a $\gamma$ fraction of the constraints of $\mathcal{L}$. The proofs in [14] and [34] employ results from extremal combinatorics and sensitivity analysis of Boolean functions to do the decoding. We will investigate why the proofs fail to produce the desired $2 - \epsilon$ hardness when the constraints are projections (i.e. when starting with the Raz Verifier as we did). We will then switch to the Unique Games conjecture to show the desired result.

For every $x \in X$ let,

$$\mathcal{F}[x] = \{F : F \subseteq R_X, \langle x, F\rangle \in \text{IS}\} \ .$$

Let $X^*$ be the set of vertices $x$ such that $\mu_p^{R_X}(\mathcal{F}[x]) \geq \delta/2$, i.e. weight$(\mathcal{F}[x]) \geq \frac{\delta}{2}$weight$(B[x])$. Using this, we have,

$$\begin{aligned} \text{weight}(IS) &= \sum_{x \in X^*} \text{weight}(\mathcal{F}[x]) + \sum_{x \notin X^*} \text{weight}(\mathcal{F}[x]) \\ \Rightarrow \qquad \delta &\leq \frac{|X^*|}{|X|} + \frac{\delta}{2}\frac{|X| - |X^*|}{|X|} \\ \Rightarrow \qquad |X^*| &\geq \frac{\delta}{2}|X| \end{aligned}$$

The crux of the argument lies in being able to associate a small set of labels $L[x] \subseteq R_X$ with every $x \in X^*$, i.e. any $x$ such that the intersection of IS with $B[x]$ is large. We will try to satisfy only those constraints that are incident on $X^*$. This is a $\delta/2$ fraction of all constraints since the $X$ side of the underlying bipartite constraint graph is regular and $|X^*| \geq \frac{\delta}{2}|X|$. Let $Y^*$ be the set of variables of $Y$ that share a constraint with some variable in $X^*$. In order to be able to satisfy the constraints incident on $X^*$, we would like to have,

$$\pi_{yx}(L[x]) \cap L[y] \neq \emptyset \ , \tag{11}$$

for every constraint $\pi_{yx}$ with $x \in X^*$ and $y \in Y^*$, where $L[y]$ is a small set of labels for $y$. We will depend on the fact that the intersection of the IS with $B[x]$ for $x \in X^*$ is large ($\mu_p^{R_X}(\mathcal{F}[x]) \geq \delta/2$) to infer $L[x]$. We do not, however, have such a direct handle on the variables $y \in Y^*$. Notice though if we ensure that,

$$\pi_{yx_1}(L[x_1]) \cap \pi_{yx_2}(L[x_2]) \neq \emptyset \ , \tag{12}$$

for every two constraints $\pi_{yx_1}, \pi_{yx_2}$ with $x_1, x_2 \in X^*$, $y \in Y^*$, and if we let,

$$L[y] \overset{def}{=} \pi_{yx(y)}(L[x(y)]) \ ,$$

where $x(y)$ is some $x \in X^*$ with which $y$ has a constraint, then condition (11) will be satisfied. Here is how we would use condition (11) to define a labeling. Suppose there is a constant $h$ such that $|L[x]| \leq h$ for all $x \in X^*$. The following probabilistic labeling completes the argument. For $x \in X^*$ (resp. $y \in Y^*$) let $A(x)$ (resp. $A(y)$) be a randomly chosen element of $L[x]$ (resp. $L[y]$). For each test $\pi_{yx}$ with $x \in X^*, y \in Y^*$, $\pi_{yx}(L[x])$ and $L[y]$ intersect and both have size at most $h$. Hence, with probability at least $1/h^2$ we have that $\pi_{yx}(A(x)) = A(y)$, which implies that the expected fraction of satisfied constraints is at least $\frac{\delta}{2h^2}$. Therefore, there exists an assignment that satisfies at least this many constraints and setting $\gamma < \frac{\delta}{2h^2}$ would give the contradiction.

But we need to show (12) provided that the sizes of the label sets are upper bounded by a constant. As in [14], let's try to use the following lemma about $s$-wise $t$-intersecting families.

**Lemma 5.9.** *[14] For arbitrary $\epsilon, \delta > 0$ and integer $s \geq 2$ with $p = 1 - \frac{1}{s} - \epsilon$, there exists $t = t(\epsilon, \delta, s)$ such that for any $s$-wise $t$-intersecting family $\mathcal{F} \subseteq 2^{[n]}$, $\mu_p(\mathcal{F}) < \delta$. Moreover, it is enough to choose $t = \Omega(\frac{1}{\epsilon^2}(\log \frac{1}{\delta} + \log(1 + \frac{1}{s\epsilon^2})))$.*

Consider any $x \in X^*$. Since $p = \frac{1}{2} - \epsilon$ and $\mu_p^{R_X}(\mathcal{F}[x]) \geq \delta/2$, by the above lemma, there exists $t = t(\epsilon, \frac{\delta}{2}, 2)$ and sets $F_1^x, F_2^x \in \mathcal{F}[x]$ such that $|F_1^x \cap F_2^x| < t$, i.e. $\mathcal{F}[x]$ is not 2-wise $t$-intersecting. Let $L[x] = F_1^x \cap F_2^x$.

Notice, however, we are *not* guaranteed for all $x_1, x_2 \in X^*$ and $y \in Y^*$ with constraints $\pi_{yx_1}, \pi_{yx_2}$ that $\pi_{yx_1}(L[x_1]) \cap \pi_{yx_2}(L[x_2]) \neq \emptyset$, even though $\langle x_1, F_1^{x_1}\rangle, \langle x_1, F_2^{x_1}\rangle, \langle x_2, F_1^{x_2}\rangle$, and $\langle x_2, F_2^{x_2}\rangle$ are in IS.[3] Consider the following simple example,

$$\begin{aligned} P_1 &:= \pi_{yx_1}(F_1^{x_1}) = \{1,2,3,7,8\} \\ P_2 &:= \pi_{yx_1}(F_2^{x_1}) = \{4,5,6,7,8\} \\ Q_1 &:= \pi_{yx_2}(F_1^{x_2}) = \{3,6,9,10\} \\ Q_2 &:= \pi_{yx_2}(F_2^{x_2}) = \{1,4,9,10\} \end{aligned}$$

Note that none of $P_i \cap Q_j,\ i,j = 1,2$ is empty since otherwise we would have an edge in IS. Also, $\pi_{yx_1}(L[x_1]) \subseteq P_1 \cap P_2 = \{7,8\}$, and $\pi_{yx_2}(L[x_2]) \subseteq Q_1 \cap Q_2 = \{9,10\}$, which implies that $\pi_{yx_1}(L[x_1]) \cap \pi_{yx_2}(L[x_2]) = \emptyset$.

This example shows a problem even if the constraints were bijections. Therefore, we need to explore different avenues for constructing the small label sets and we turn to the techniques of [34].

We construct the labels in [34] partly from the next lemma. This lemma is obtained by combining the fact that each family $\mathcal{F}[x]$ is a monotone family with the Russo-Margulis identity [42, 38] and Friedgut's Theorem [22]. It is easy to see why each $\mathcal{F}[x]$ is a monotone family. Let $F \in \mathcal{F}[x]$. This means that $\forall\langle x', G\rangle \in$ IS, $\pi_{yx}(F) \cap \pi_{yx'}(G) \neq \emptyset$. But then, any $F' : F' \supseteq F$ must satisfy the same constraints. Hence, there is no edge between $\langle x, F'\rangle$ and any other vertex in IS implying that $F' \in \mathcal{F}[x]$. Now, the Russo-Margulis identity guarantees that a monotone family will have low average sensitivity, and Friedgut's Theorem says that a family with low average sensitivity can be well approximated by a core-family with a "small" core. We will use these cores as part of our label sets. Let $\eta > 0$ be a sufficiently small "accuracy" parameter:

**Lemma 5.10.** *[34, Lemma 4.2] For every variable $x \in X^*$, there exists a real number $p[x] \in (1 - \frac{1}{2} - \epsilon, 1 - \frac{1}{2} - \frac{\epsilon}{2})$ and a core-family $\widehat{\mathcal{F}}[x] \subseteq 2^{R_X}$ with core $C[x]$ such that,*

- *The average sensitivity $as_{p[x]}^{R_X}(\mathcal{F}[x]) \leq \frac{2}{\epsilon}$.*
- *The size of $C[x]$ is at most $h_0$, which is a constant depending only on $\epsilon, \delta, \eta$.*
- *$\mu_{p[x]}^{R_X}(\mathcal{F}[x] \,\Delta\, \widehat{\mathcal{F}}[x]) < \eta$, and in particular $\mu_{p[x]}^{R_X}(\widehat{\mathcal{F}}[x]) \geq \delta/4$ provided $\eta < \delta/4$.*

We fatten each core $C[x]$ with the following set to facilitate the analysis:

$$\text{Infl}[x] = \{\sigma \in R_X \setminus C[x] : \text{Influence}_{p[x]}^{R_X}(\mathcal{F}[x], \sigma) \geq \eta'\} \ ,$$

where $\eta' > 0$ is another accuracy parameter. Now,

$$\eta' |\text{Infl}[x]| \leq \sum_{\sigma \in R_X} \text{Influence}_{p[x]}^{R_X}(\mathcal{F}[x], \sigma) \leq \frac{2}{\epsilon} \ ,$$

[3] In the construction of [14], $k = 4$ and $\{\langle x_1, F_1\rangle, \langle x_1, F_2\rangle, \langle x_2, G_1\rangle, \langle x_2, G_2\rangle\}$ is an edge if $\pi_{yx_1}(F_1 \cap F_2) \cap \pi_{yx_2}(G_1 \cap G_2) = \emptyset$. Hence, it must be the case that $\pi_{yx_1}(L[x_1]) \cap \pi_{yx_2}(L[x_2]) \neq \emptyset$ for otherwise $\{\langle x_1, F_1^{x_1}\rangle, \langle x_1, F_2^{x_1}\rangle, \langle x_2, F_1^{x_2}\rangle, \langle x_2, F_2^{x_2}\rangle\}$ would be a hyperedge in IS.

by recalling the definition of the average sensitivity and using Lemma 5.10. We define the sets of labels of each $x \in X^*$ as

$$L[x] \overset{def}{=} C[x] \cup \text{Infl}[x] \ .$$

The size of each $L[x]$ is at most $h_0 + \frac{2}{\epsilon\eta'} \overset{def}{=} h$, which is a constant.

It remains to show condition (12) for every $\pi_{yx_1}$, $\pi_{yx_2}$ sharing the same $y \in Y^*$. Can we assume as in [34] that $\pi_{yx_1}(L[x_1]) \cap \pi_{yx_2}(L[x_2]) = \emptyset$ and hope to reach a contradiction? The difficulty, both for us and for [34], is that $L[x_1]$ and $L[x_2]$ are not necessarily in $\mathcal{F}[x_1]$ and $\mathcal{F}[x_2]$ respectively. If they were, we would immediately reach a contradiction as we would have that the edge $\{\langle x_1, L[x_1]\rangle, \langle x_2, L[x_2]\rangle\}$ is contained in IS. In [34], based on the definitions of $L[x_1]$ and $L[x_2]$ and the assumption above, the existence of two other sets $F_1 \in \mathcal{F}[x_1]$ and $F_2 \in \mathcal{F}[x_2]$ is exhibited such that $\pi_{yx_1}(F_1) \cap \pi_{yx_2}(F_2) = \emptyset$, which leads to the desired contradiction. In our case, however, it is possible that the image of say $\mathcal{F}[x_1]$ has low weight in the projected space, i.e. $\mu_p^{R_Y}(\{\pi_{yx}(F) : F \in \mathcal{F}[x_1]\}) < \delta/2$. Consider the following extreme example. Suppose for simplicity $|R_X| = 2^{3u}$. Let $\mathcal{F}[x_1]$ be all non-empty subsets of the first $2^{3u} - 2^u + 2$ elements of $R_X$. Suppose $\pi_{yx_1}$ maps those elements to two elements in $R_Y$ and maps the $2^u - 2$ remaining elements of $R_X$ to distinct elements of $R_Y$. Hence, $\mu_p^{R_X}(\mathcal{F}[x_1]) > 3/2^{2^u} + \epsilon$ and the weight of the image of $\mathcal{F}[x_1]$ under $\mu_p^{R_Y}$ is $< 3/2^{2^u} + \epsilon$. Note that these are all constants since $u$, the number of repetitions, is constant. This is a problem since it causes the proof of the analogue of Lemma 5.13 in the projected space to break down. Such a situation does not occur when the constraints are bijections.

We continue by assuming that the underlying constraint graph used in the construction is based on the Unique Games conjecture. We start with two general lemmas needed for the proof.

**Lemma 5.11.** *[34, Lemma 2.2] If $\mathcal{F} \subseteq 2^R$ is monotone and $p \geq q$, then $\mu_p^R(\mathcal{F}) \geq \mu_q^R(\mathcal{F})$.*

**Lemma 5.12.** *[34, Lemma 2.5] Let $\epsilon > 0$ be an arbitrarily small constant and define $p = 1 - \frac{1}{k} - \epsilon$ to be the bias parameter. Then, for a sufficiently large universe $R$, the following holds. For any $\mathcal{F} \subseteq 2^R$ such that $\mu_p^R(\mathcal{F}) \geq 1 - \frac{1}{k}$ there exist $k$ sets in the family $\mathcal{F}$ whose intersection is empty.*

The proof of the first lemma can be found in [16, Prop. 3.3] for example, and the proof of the second can be found in [13, Lemma A.4].

The proof towards contradiction is continued in [34] assuming $\pi_{yx_1}$ and $\pi_{yx_2}$ are identities. This is indeed without loss of generality, and we shortly elaborate on this. With identities, the assumption $\pi_{yx_1}(L[x_1]) \cap \pi_{yx_2}(L[x_2]) = \emptyset$ implies $L[x_1] \cap L[x_2] = \emptyset$, which in turn implies the following: There exists a subset $U_0 \subseteq C[x_1]$ with a large family of extensions $H$ that do not include elements from the cores $C[x_1], C[x_2]$ such that $H \cup U_0 \in \mathcal{F}[x_1]$. This statement is formalized in Lemma 5.13 below. A similar lemma shows the existence of a subset $V_0 \subseteq C[x_2]$ with a large family of extensions $H_2$ such that $H_2 \cup V_0 \in \mathcal{F}[x_2]$. Since both families are "large", Lemma 5.12 gives us two sets $H_1, H_2$ in their intersection such that $H_1 \cap H_2 = \emptyset$ allowing us to complete the proof.

We will use the core-families given by Lemma 5.10 with their core fattenings to prove Lemma 5.13. Note that now $R_X = R_Y = R$.

**Lemma 5.13.** *[34, Lemma 4.3] There exists $U_0 \subseteq C[x_1]$ such that defining*

$$R' \overset{def}{=} R \setminus (C[x_1] \cup C[x_2]); \textit{ and,}$$
$$\mathcal{H}[x_1] \overset{def}{=} \{H : H \in 2^{R'}, H \cup U_0 \in \mathcal{F}[x_1]\} \ ,$$

*we have $\mu_{p[x_1]}^{R'}(\mathcal{H}[x_1]) \geq 1 - 8\eta/\delta$.*

*Proof.* The idea of the proof is the following. If we choose $U_0 \subseteq C[x_1]$ and $U_0 \in \widehat{\mathcal{F}}[x_1]$, then roughly speaking, for any $G \subseteq R \setminus C[x_1]$ there is a good chance that $U_0 \cup G \in \mathcal{F}[x_1]$. This will follow from the fact that $\widehat{\mathcal{F}}[x_1]$ is a core-family (hence, $U_0 \cup G \in \widehat{\mathcal{F}}[x_1]$) that well-approximates $\mathcal{F}[x_1]$ (as $\mu^R_{p[x_1]}(\mathcal{F}[x_1] \,\Delta\, \widehat{\mathcal{F}}[x_1]) < \eta$), and that has significant weight (as $\mu^R_{p[x_1]}(\widehat{\mathcal{F}}[x_1]) \geq \delta/4$).

Let us formalize this intuition. We are saying that if $\mathcal{G} = \{G \subseteq R \setminus C[x_1] : G \cup U_0 \in \mathcal{F}[x_1]\}$, then $\mu^{R\setminus C[x_1]}_{p[x_1]}(\mathcal{G})$ is large. The set $\mathcal{G}$ looks very similar to $\mathcal{H}[x_1]$ except that sets in the latter family do not include any element from $C[x_2]$. Suppose we can ensure that

$$G \in \mathcal{G} \iff (G \cup U_0) \setminus C[x_2] \in \mathcal{F}[x_1] \ , \tag{13}$$

then we would have that

$$G \in \mathcal{G} \iff (G \setminus C[x_2]) \cup U_0 \in \mathcal{F}[x_1] \iff G \setminus C[x_2] \in \mathcal{H}[x_1] \ , \tag{14}$$

which follows since $U_0 \cap C[x_2] = \emptyset$ by our assumption that $L[x_1] \cap L[x_2] = \emptyset$ and from the definition of $\mathcal{H}[x_1]$. We can ensure (13) by slightly modifying the definition of $\mathcal{G}$. Let,

$$\mathcal{G} = \{G \subseteq R \setminus C[x_1] : G \cup U_0 \in \mathcal{F}'[x_1]\}; \text{ where,}$$
$$\mathcal{F}'[x_1] = \{F \in \mathcal{F}[x_1] : F \setminus C[x_2] \in \mathcal{F}[x_1]\} \ .$$

With this new definition of $\mathcal{G}$, (14) implies that,

$$\mu^{R\setminus C[x_1]}_{p[x_1]}(\mathcal{G}) = \mu^{R'}_{p[x_1]}(\mathcal{H}[x_1]) \ . \tag{15}$$

However, we have a relationship between the core-family $\widehat{\mathcal{F}}[x_1]$ and $\mathcal{F}[x_1]$, whereas we now need a relationship between $\widehat{\mathcal{F}}[x_1]$ and $\mathcal{F}'[x_1]$. This is where the set $\mathrm{Infl}[x_1]$ comes in. Again, by the assumption $L[x_1] \cap L[x_2] = \emptyset$ we have $C[x_2] \cap \mathrm{Infl}[x_1] = \emptyset$, which implies that the elements in $C[x_2]$ have influence at most $\eta'$ on $\mathcal{F}[x_1]$. A technical lemma in [34, Lemma 2.4] gives

$$\mu^R_{p[x_1]}(\mathcal{F}[x_1] \,\Delta\, \mathcal{F}'[x_1]) \leq \eta \ ,$$

by setting $\eta'$ to be a small enough constant. Hence,

$$\mu^R_{p[x_1]}(\widehat{\mathcal{F}}[x_1] \,\Delta\, \mathcal{F}'[x_1]) \leq \mu^R_{p[x_1]}(\widehat{\mathcal{F}}[x_1] \,\Delta\, \mathcal{F}[x_1]) + \mu^R_{p[x_1]}(\mathcal{F}[x_1] \,\Delta\, \mathcal{F}'[x_1]) < 2\eta \ .$$

It remains to show $\exists U_0 \subseteq C[x_1], U_0 \in \widehat{\mathcal{F}}[x_1]$ such that $\mu^{R\setminus C[x_1]}_{p[x_1]}(\mathcal{G}) \geq 1 - 8\eta/\delta$, or equivalently,

$$\Pr_{G \in \mu^{R\setminus C[x_1]}_{p[x_1]}} \left[G \cup U_0 \notin \mathcal{F}'[x_1]\right] \leq 8\eta/\delta \ . \tag{16}$$

Now,

$$2\eta > \mu^R_{p[x_1]}(\widehat{\mathcal{F}}[x_1] \,\Delta\, \mathcal{F}'[x_1]) \geq \Pr_{G \in \mu^R_{p[x_1]}} \left[G \in \widehat{\mathcal{F}}[x_1] \text{ and } G \notin \mathcal{F}'[x_1]\right] \tag{17a}$$

$$\geq \Pr_{G \in \mu^R_{p[x_1]}} \left[G \notin \mathcal{F}'[x_1] \mid G \in \widehat{\mathcal{F}}[x_1]\right] \cdot \frac{\delta}{4} \tag{17b}$$

$$= \frac{\delta}{4} \sum_{U \subseteq C[x_1]} \Pr_{G \in \mu^{R\setminus C[x_1]}_{p[x_1]}} \left[G \cup U \notin \mathcal{F}'[x_1] \mid G \cup U \in \widehat{\mathcal{F}}[x_1]\right] \tag{17c}$$

where (17b) follows from the fact that $\mu^R_{p[x_1]}(\widehat{\mathcal{F}}[x_1]) \geq \delta/4$ in Lemma 5.10. Setting $\eta < \delta/4$, the fact that $\mu^R_{p[x_1]}(\mathcal{F}[x_1]) \geq \mu^R_p(\mathcal{F}[x_1]) \geq \delta/2$ (by Lemma 5.11 and the monotonicity of $\mathcal{F}[x_1]$), combined with Freidgut's Theorem, which says $\mu^R_{p[x_1]}(\mathcal{F}[x_1] \,\Delta\, \widehat{\mathcal{F}}[x_1]) < \eta$, guarantee this lower bound on the weight of $\widehat{\mathcal{F}}[x_1]$. Now, for our choices of $G$ and $U$ in (17c),

$$G \cup U \in \widehat{\mathcal{F}}[x_1] \iff (G \cup U) \cap C[x_1] \in \widehat{\mathcal{F}}_C[x_1] \iff U \in \widehat{\mathcal{F}}_C[x_1] \iff U \in \widehat{\mathcal{F}}[x_1] \ .$$

Therefore,

$$\frac{8\eta}{\delta} > \sum_{U \subseteq C[x_1], U \in \widehat{\mathcal{F}}[x_1]} \Pr_{G \in \mu^{R \setminus C[x_1]}_{p[x_1]}} \left[ G \cup U \notin \mathcal{F}'[x_1] \right] \ ,$$

implying that $\exists U_0 \subseteq C[x_1], U_0 \in \widehat{\mathcal{F}}[x_1]$ satisfying (16) and completing the proof. □

The bijections simply rename the space $R$, and with bijections (as opposed to identities) the lemma would be stated as: (For ease of notation, we write $\pi_{yx_j}(S) := \pi_j S$)

**Lemma 5.14.** *There exists $\pi_1 U_0 \subseteq \pi_1 C[x_1]$ such that defining*

$$R' \overset{def}{=} R \setminus (\pi_1 C[x_1] \cup \pi_2 C[x_2]); \text{ and,}$$
$$\mathcal{H}[x_1] \overset{def}{=} \{H : H \in 2^{R'}, H \cup \pi_1 U_0 \in \pi_1 \mathcal{F}[x_1]\} \ ,$$

*we have $\mu^{R'}_{p[x_1]}(\mathcal{H}[x_1]) \geq 1 - 8\eta/\delta$, where $\pi_1 \mathcal{F}[x_1] = \{\pi_1 F : F \in \mathcal{F}[x_1]\}$.*

Analogously, we have a lemma stating the existence of $\pi_2 V_0 \subseteq \pi_2 C[x_2]$ with the family $\mathcal{H}[x_2]$ of extensions such that $\mu^{R'}_{p[x_2]}(\mathcal{H}[x_2]) \geq 1 - 8\eta/\delta$. The proofs of the two lemmas are similar to the proof of Lemma 5.13 modulo the renaming of variables. Armed with these two lemmas, we can complete the proof. Let $p^* = \frac{1}{2} - \frac{\epsilon}{2}$. Note that $\mathcal{H}[x_1]$ and $\mathcal{H}[x_2]$ are both monotone subfamilies of $2^{R'}$. Therefore by Lemma 5.11, $\mu^{R'}_{p^*}(\mathcal{H}[x_1]) \geq \mu^{R'}_{p[x_1]}(\mathcal{H}[x_1]) \geq 1 - 8\eta/\delta$ and similarly for $\mathcal{H}[x_2]$. Hence, the intersection of the two families satisfies,

$$\mu^{R'}_{p^*}(\mathcal{H}[x_1] \cap \mathcal{H}[x_2]) \geq 1 - \frac{16\eta}{\delta} > \frac{1}{2} \ ,$$

by choosing $\eta < \delta/32$. Therefore, setting $k = 2$ in Lemma 5.12 implies that there exist sets $H_1, H_2 \in \mathcal{H}[x_1] \cap \mathcal{H}[x_2]$ such that $H_1 \cap H_2 = \emptyset$. Now define $G_1 = \pi_1 U_0 \cup H_1$ and $G_2 = \pi_2 V_0 \cup H_2$. By the definition of $\mathcal{H}[x_1]$ and $\mathcal{H}[x_2]$, we have $G_1 \in \pi_1 \mathcal{F}[x_1]$ and $G_2 \in \pi_2 \mathcal{F}[x_2]$. Furthermore,

$$F_1 = (\pi_{yx_1})^{-1}(G_1) \in \mathcal{F}[x_1]; \text{ and,}$$
$$F_2 = (\pi_{yx_2})^{-1}(G_2) \in \mathcal{F}[x_2],$$

by the definition of $\pi_1 \mathcal{F}[x_1]$ and $\pi_2 \mathcal{F}[x_2]$. Thus, $\langle x_1, F_1 \rangle$ and $\langle x_2, F_2 \rangle$ are vertices in the supposed IS and they form an edge since,

$$\begin{aligned} \pi_1 F_1 \cap \pi_2 F_2 &= G_1 \cap G_2 \\ &= (\pi_1 U_0 \cap \pi_2 V_0) \cup (\pi_1 U_0 \cap H_2) \cup (\pi_2 V_0 \cap H_1) \cup (H_1 \cap H_2) \\ &= \emptyset \ . \end{aligned}$$

The last line follows since

- $(\pi_1 U_0 \cap \pi_2 V_0) = \emptyset$ by the assumption that $\pi_1 L[x_1] \cap \pi_2 L[x_2] = \emptyset$,
- $(\pi_1 U_0 \cap H_2) = (\pi_2 V_0 \cap H_1) = \emptyset$ by recalling that $H_1, H_2 \subseteq R' = R \setminus (\pi_1 C[x_1] \cup \pi_2 C[x_2])$; and,
- $H_1 \cap H_2 = \emptyset$ by Lemma 5.12.

This completes our investigation.

In summary, assuming the Unique Games conjecture, we obtain the tightest possible hardness of approximation result for vertex cover. On the other hand, the best known inapproximation result for vertex cover starting with the Raz Verifier and utilizing similar tools (including biased long codes, Friedgut's Theorem, and theorems from extremal combinatorics) is by Dinur and Safra [16] who were able to show a $1.36$ hardness factor. Their result comes about five years after Håstad's $\frac{7}{6}$ hardness result [44].

# 6 The Plausibility of the Unique Games Conjecture

In light of its consequences, it is important to investigate the plausibility of the Unique Games conjecture. One important aspect of the conjecture is the domain size $k = k(\zeta, \delta)$ of the provers' answers. For example, it is easy to see that $k$ must be at least $1/\delta$. By choosing their answers uniformly at random from the domain of possible answers, the provers can make the verifier always accept with probability $1/k$. Hence, $1/k \leq \delta$. Khot [32] also relates the domain size to $\zeta$ through the following theorem,

**Theorem 6.1.** *[32, Thm. 1] There exists a polynomial time algorithm such that given a unique 2-prover game with value $1-\epsilon$ and answers from a domain of size $k$, it finds prover strategies that make the verifier accept with probability $1 - O(k^2 \epsilon^{1/5} \sqrt{\log\left(\frac{1}{\epsilon}\right)})$.*

We present the algorithm of Khot that gives this theorem below. The theorem implies that for the Unique Games conjecture to hold, it must be the case that,

$$1 - ck^2 \zeta^{1/5} \sqrt{\log\left(\frac{1}{\zeta}\right)} \leq \delta \ , \tag{18}$$

where $c$ is some constant; for otherwise, we would be able to distinguish between instances of unique games whose value is at least $1-\zeta$ and instances whose value is at most $\delta$. Expression (18) implies that $k \geq 1/\left(\zeta^{1/10}\sqrt{c}(\log(1/\zeta))^{1/4}\right)$. [4] Khot notes that disproving the conjecture may require an algorithm that gives a theorem similar to Thm. 6.1, but whose performance is independent of $k$. Indeed, Trevisan [46] provides such an algorithm that disproves a stronger version of the conjecture. Namely, for a constant $c$ and for every $\epsilon > 0$, the Unique Games conjecture with completeness $1 - c(\epsilon^3/(\log|\Pi|)^3)$ and soundness $1-\epsilon$ is false. Here, $\Pi$ is the set of constraints in the game, i.e. the completeness parameter $\zeta$ is not a constant, but is dependent on the input.

Nonetheless, a weaker version of the conjecture was recently confirmed by Feige and Reichman [20]. In [32], Khot raised the question of whether the value of a unique 2P1R game with domain size $k$ is hard to approximate within factor $f(k)$ where $f(k) \to \infty$ as $k \to \infty$. Feige and Reichman answer this question positively:

**Theorem 6.2.** *[41, 20] There is some $\gamma > 0$ such that for every prime $p$ it is NP-hard to approximate the value of a unique 2P1R game with answers from a domain of size $p$ to within a factor smaller than $p^\gamma$.*

[4] In [32], it is stated that $k$ must be at least $1/\zeta^{1/10}$. It seems though that $k$ need only be greater than the expression here.

For arbitrarily small $\delta > 0$, we can choose $p$ large enough so that $p^{\gamma} \geq 1/\delta$. Hence, Thm. 6.2 can be restated as follows,

**Corollary 6.3.** *[20, Cor. 2] For any arbitrarily small constant $\delta > 0$, there exists a constant $0 < \zeta < 1$ and a prime $p$ such that it is NP-hard to determine whether a unique 2P1R game with answers from a domain of size $p$ has value at least $1 - \zeta$ or at most $\delta(1 - \zeta)$.*

Technically, this result is weaker than the Unique Games conjecture since in the latter, both constants $\delta$ and $\zeta$ are arbitrary. Furthermore, the result does not provide the desired gap provided by the UGC since, as noted in [46, 12], the value of the instances produced by the proof is very small. Prior to this result, it was only known that approximating the value of unique games within *some* constant factor is NP-hard. This comes from the fact that Max-2Lin-2 is NP-hard to approximate within a factor $\frac{12}{11} - \epsilon$ [44], and via a reduction that shows given an instance of Max-2Lin-2 that is $\mu$-satisfiable, we can transform it to a unique game whose value is $\frac{1+\mu}{2}$ (see, e.g., [41]).

Tackling Max-2Lin-p seems to be a fruitful approach for proving the Unique Games conjecture. Feige and Reichman state Thm. 6.2 [20, Thm. 4] for a variant of Max-2Lin-p, they call *proper Max-2Lin-p*, which they show is equivalent to a unique game. Also, as stated in Sec. 4.4, [39] show that if for all $\epsilon > 0$ there is a $p = p(\epsilon)$ such that Max-2Lin-p has $(1 - \epsilon, \epsilon)$-hardness, then the Unique Games conjecture is true.

On the other hand, Khot *et al.* [30] show that the Unique Games problem is formally easier than improving the approximation guarantee for Max-Cut, which may provide encouragement for attacking unique games algorithmically. The SDP of Khot, and Trevisan's algorithms are steps in this direction.

Instead of attacking the Unique Games conjecture, another interesting avenue of exploration would be to prove at least some of the results it implies using Khot's [31] Smooth Label Cover or the multi-latyered version of it. The maps in a Smooth Label Cover have the smoothness property (see Sec. 4.3, Eq. (10)), which is a weaker analogue of the bijection property of the maps in a Unique Label Cover. Khot used the multi-layered version of smooth label cover to confirm Conjecture 4.7 about the hardness of approximating NAE$_{3,3}$, which is based on Unique Label Cover. Most interesting would be to achieve a hardness factor better than $1.36$ for Vertex Cover.

**Proof of Theorem 6.1.** We now present the semidefinite programming based algorithm of [32] that gives Thm. 6.1 and we provide a sketch of its proof. The full details can be found in [32].

Assume we are given a weighted unique Label Cover instance $\mathcal{L}(X, [k], \{\pi_{uv}\}, \{w_{uv}\})$ where the constraint graph need not be bipartite. Namely, $X$ is a set of $n$ variables which take values from the domain $[k]$. For every pair $(u, v)$, there is a constraint which is a bijection $\pi_{uv} : [k] \to [k]$ with weight $w_{uv}$ where $\sum_{u,v} w_{uv} = 1$. A constraint $\pi_{uv}$ is said to be satisfied by an assignment $A : X \to [k]$ if $\pi_{uv}(A(u)) = A(v)$. The goal is to find an assignment that maximizes the weight of satisfied constraints.

We first formulate this problem as a strict quadratic program (where each monomial in the objective function and the constraints have degree 2 or 0), and then relax the program to a vector program. For each $u \in X$ we create $k$ new variables $u_1, \ldots, u_k$ where $u_i$ indicates if variable $u$ is assigned the value $i \in [k]$. Clearly, if $u$ is assigned $i_0 \in [k]$ we wish to have $u_{i_0} = 1$ and $u_i = 0$ for all $i \neq i_0$. This is achieved by the following constraints:

$$u_1^2 + u_2^2 + \cdots + u_k^2 = 1 \quad \forall u \in X \tag{19a}$$

$$u_i u_j = 0 \quad \forall u \in X \text{ and } \forall i \neq j \tag{19b}$$

We wish to maximize the following objective function

$$\sum_{u,v} w_{uv} \left( u_1 v_{\pi_{uv}(1)} + u_2 v_{\pi_{uv}(2)} + \cdots + u_k v_{\pi_{uv}(k)} \right) \quad . \tag{20}$$

Before relaxing the program, we need to make sure that the solution to the vector program has the same type of symmetries as the solution to the quadratic program. Therefore, we add the next two constraints which are implied by constraints (19)

$$u_i v_j \geq 0 \quad \forall u, v \in X \text{ and } \forall i, j \tag{21a}$$

$$\sum_{1 \leq i,j \leq k} u_i v_j = 1 \tag{21b}$$

The relaxation is standard. Each auxiliary variable $u_i$ will be replaced with a vector $\vec{u}_i$ in $\Re^{kn}$, and each degree 2 term in the objective function and the constraints will be replaced by the corresponding inner product. The complete program is,

$$\begin{aligned}
\text{maximize} \quad & \sum_{u,v} w_{uv} \left( \vec{u}_1 \cdot \vec{v}_{\pi_{uv}(1)} + \cdots + \vec{u}_k \cdot \vec{v}_{\pi_{uv}(k)} \right) && && \text{(22a)} \\
\text{subject to} \quad & \vec{u}_1 \cdot \vec{u}_1 + \vec{u}_2 \cdot \vec{u}_2 + \cdots + \vec{u}_k \cdot \vec{u}_k = 1 && \forall u \in X && \text{(22b)} \\
& \vec{u}_i \cdot \vec{u}_j = 0 && \forall u \in X \quad \forall i \neq j && \text{(22c)} \\
& \vec{u}_i \cdot \vec{v}_j \geq 0 && \forall u, v \in X \quad \forall i, j && \text{(22d)} \\
& \sum_{1 \leq i,j \leq k} \vec{u}_i \cdot \vec{v}_j = 1 && && \text{(22e)}
\end{aligned}$$

Clearly, any feasible solution to the quadratic program yields a solution to the vector program having the same objective function value by setting $\vec{u}_i = (u_i, 0, \ldots, 0)$.

For all $u \in X$, let $\vec{u} = \sum_{i=1}^{k} \vec{u}_i$. In any feasible solution of the SDP and for any two variables $u, v$, constraint (22e) implies that $\vec{u} \cdot \vec{v} = 1$, and constraints (22b) and (22c) imply that $\|\vec{u}\| = \|\vec{v}\|$. Hence for all $u, v \in X$, $\vec{u} = \vec{v}$. Denote $\vec{s} = \vec{u}$ which is the same for all variables $u$.

The following algorithm produces an assignment whose expected weight is $1 - O(k^2 \epsilon^{1/5} \sqrt{\log\left(\frac{1}{\epsilon}\right)})$, which is sufficient to prove Thm. 6.1 as it shows that there exists an assignment with this weight. Recall that the weight of an assignment is the total weight of edges it satisfies. In a 2-prover game an assignment corresponds to a strategy of the provers and its weight is equal to the probability of acceptance of the verifier given this strategy.

**Algorithm 1:**

1. Solve vector program (22a)

2. Pick $\vec{r}$ to be a uniformly distributed vector on the $nk$-dimensional unit sphere.
   Assume $\vec{r} \cdot \vec{s} \geq 0$ by replacing $\vec{r}$ with $-\vec{r}$ if needed.

3. Construct the following assignment $A$. For every $u \in X$, let

$$A(u) = i_0 \quad \text{where} \quad i_0 = \arg\max_{1 \leq i \leq k} (\vec{r} \cdot \vec{u}_i) \quad .$$

Let $\alpha_{uv} = \sum_i \vec{u}_i \cdot \vec{v}_{\pi_{uv}(i)}$, i.e. $\alpha_{uv}$ is the part if the SDP objective function corresponding to the constraint $(u, v)$. We are given that $\sum_{u,v} w_{uv}\alpha_{uv} \geq 1 - \epsilon$. A simple calculation shows

$$\sum_{\alpha_{uv} \geq 1 - \frac{1}{2}\epsilon^{4/5}} w_{uv} \geq 1 - 2\epsilon^{1/5} \ . \tag{23}$$

Hence, if with probability $p$ we can satisfy the pairs $(u, v)$ for which $\alpha_{uv}$ is close to 1, then the expected weight of the assignment produced will be at least $p(1 - 2\epsilon^{1/5})$. Specifically, the proof shows that for any $\alpha_{uv} \geq 1 - \frac{1}{2}\epsilon^{4/5}$, $\Pr[\pi_{uv}(A(u)) = A(v)] = 1 - O(k^2\epsilon^{1/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)})$; therefore, the expected weight of the assignment produced is $1 - O(k^2\epsilon^{1/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)})$ as desired. In the remaining part of the argument, let $\pi := \pi_{uv}$.

The case when $\alpha_{uv} = 1$ provides some intuition behind the proof. When $\alpha_{uv} = 1$, we have,

$$2\alpha_{uv} = 2\sum_i \vec{u}_i \cdot \vec{v}_{\pi(i)} = \sum_i \vec{u}_i \cdot \vec{u}_i + \sum_i \vec{v}_{\pi(i)} \cdot \vec{v}_{\pi(i)} \ , \tag{24}$$

since constraint (22b) ensures each sum on the r.h.s. is 1. Expression (24) can be rewritten as,

$$\begin{aligned}
&\textstyle\sum_i \|\vec{u}_i\|^2 + \|\vec{v}_{\pi(i)}\|^2 - 2\vec{u}_i \cdot \vec{v}_{\pi(i)} = 0 \\
\Rightarrow &\forall i \quad \|\vec{u}_i\|^2 + \|\vec{v}_{\pi(i)}\|^2 - 2\vec{u}_i \cdot \vec{v}_{\pi(i)} = 0 \\
\Rightarrow &\forall i \quad \|\vec{u}_i\|^2 + \|\vec{v}_{\pi(i)}\|^2 - 2\|\vec{u}_i\|\|\vec{v}_{\pi(i)}\| = 0 \\
\Rightarrow &\forall i \quad \|\vec{u}_i - \vec{v}_{\pi(i)}\| = 0
\end{aligned}$$

The second and third lines follow from the fact that for all $i$, $\|\vec{u}_i\|^2 + \|\vec{v}_{\pi(i)}\|^2 \geq 2\|\vec{u}_i\|\|\vec{v}_{\pi(i)}\| \geq 2\vec{u}_i \cdot \vec{v}_{\pi(i)}$. Hence, for all $i$, $\vec{u}_i = \vec{v}_{\pi(i)}$ and for any vector $\vec{r}$, if $i_0$ is the index that maximizes $\vec{r} \cdot \vec{u}_i$, then $\vec{r} \cdot \vec{v}_{\pi(j)}$ is maximized at index $\pi(i_0)$. The algorithm thus assigns $A(u) = i_0$ and $A(v) = \pi(i_0)$ satisfying the constraint. When $\alpha_{uv}$ is close to 1, however, a similar calculation to the one above only guarantees that for all $i$, $\|\vec{u}_i - \vec{v}_{\pi(i)}\| \leq \epsilon^{2/5}$.

Again, let $i_0 \in [k]$ be the index that maximizes $\vec{r} \cdot \vec{u}_i$. The proof shows that with probability $1 - O(k^2\epsilon^{1/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)})$ we have,

$$\forall i \neq i_0, \qquad |\vec{r} \cdot (\vec{u}_{i_0} - \vec{u}_i)| \geq 5\epsilon^{2/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)} \ , \tag{25}$$

and, using the fact that $\|\vec{u}_i - \vec{v}_{\pi(i)}\| \leq \epsilon^{2/5}$, we have,

$$\forall i, \qquad |\vec{r} \cdot (\vec{u}_i - \vec{v}_{\pi(i)})| \leq \epsilon^{2/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)} \ . \tag{26}$$

Expressions (25) and (26) imply that if $i \neq i_0$,

$$\vec{r} \cdot \vec{u}_{i_0} - \vec{r} \cdot \vec{v}_{\pi(i)} \geq 4\epsilon^{2/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)} \ .$$

But expression (26) says $\vec{r} \cdot \vec{u}_{i_0} - \vec{r} \cdot \vec{v}_{\pi(i_0)} \leq \epsilon^{2/5}\sqrt{\log\left(\frac{1}{\epsilon}\right)}$. Therefore, $\pi(i_0)$ is the index that maximizes $\vec{r} \cdot \vec{v}_i$, and the algorithm sets $A(v) = \pi(i_0) = \pi(A(u))$.

## References

[1] Amit Agrawal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. $O(\sqrt{\log n})$ approximation algorithms for Min UnCut, Min 2CNF Deletion, and Directed Cut Problems. In *STOC '05: Proceedings of the thirty-sixth ACM Symposium on Theory of Computing*, Baltimore, MD, USA, May 2005.

[2] Sanjeev Arora. *Probabilistic checking of proofs and hardness of approximation problems*. PhD thesis, University of California at Berkeley, Berkeley, CA, USA, 1995.

[3] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[4] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 222–231, New York, NY, USA, 2004. ACM Press.

[5] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of np. *J. ACM*, 45(1):70–122, 1998.

[6] L Babai. Trading group theory for randomness. In *STOC '85: Proceedings of the seventeenth annual ACM symposium on Theory of computing*, pages 421–429, New York, NY, USA, 1985. ACM Press.

[7] L Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. In *IEEE Symposium on Foundations of Computer Science*, pages 16–25, 1990.

[8] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, pcps, and nonapproximability—towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998.

[9] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Widgerson. Multi-prover interactive proofs: how to remove intractability. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 113–131, New York, NY, USA, 1988. ACM Press.

[10] Avrim Blum and David Karger. An $\tilde{O}(n^{3/14})$-coloring algorithm for 3-colorable graphs. *Inf. Process. Lett.*, 61(1):49–53, 1997.

[11] J. Bourgain. On the distribution of the fourier spectrum of boolean functions. In *Israel Journal of Mathematics*, volume 131, pages 269–276, 2002.

[12] Shuchi Chawla, Robert Krauthgamer, Ravi Kumar, Yuval Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. In *IEEE Conference on Computational Complexity*, June 2005. To appear.

[13] Irit Dinur, Venkatesan Guruswami, and Subhash Khot. Vertex cover on k-uniform hypergraphs is hard to approximate within factor $(k-3-epsilon)$. *Electronic Colloquium on Computational Complexity (ECCC)*, (027), 2002.

[14] Irit Dinur, Venkatesan Guruswami, Subhash Khot, and Oded Regev. A new multilayered pcp and the hardness of hypergraph vertex cover. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 595–601, New York, NY, USA, 2003. ACM Press.

[15] Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring, April 13 2005.

[16] Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover, 2004.

[17] Uriel Feige. Error reduction - the state of the art. Technical report, Weizmann Institute of Technology, 1995.

[18] Uriel Feige. Lecture notes for a course on probabilistically checkable proofs and hardness of approximation. http://www.wisdom.weizmann.ac.il/ feige/pcp.html, 2003.

[19] Uriel Feige, Shafi Goldwasser, Laszlo Lov&#225;sz, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, 1996.

[20] Uriel Feige and Daniel Reichman. On systems of linear equations with two variables per equation. In *APPROX-RANDOM '04*, pages 117–127, 2004.

[21] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theor. Comput. Sci.*, 134(2):545–557, 1994.

[22] E. Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–35, 1998.

[23] Michael Garey and David Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Company, 1979.

[24] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM J. Comput.*, 25(2):235–251, 1996.

[25] Michel X. Goemans and David P. Williamson. Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[26] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[27] Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. In *ECCCTR: Electronic Colloquium on Computational Complexity, technical reports*, 1997.

[28] Jonas Holmerin. *On Probabilistic Proof Systems and Hardness of Approximation*. PhD thesis, Stockholm University, Stockholm, Sweden, 2002.

[29] Sanjeev Khanna, Nathan Linial, and Shmuel Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000.

[30] Khot, Kindler, Mossel, and O'Donnell. Optimal inapproximability results for max-cut and other 2-variable CSPs? In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

[31] Subhash Khot. Hardness results for coloring 3 -colorable 3 -uniform hypergraphs. In *FOCS '02: Proceedings of the 43rd Symposium on Foundations of Computer Science*, pages 23–32, Washington, DC, USA, 2002. IEEE Computer Society.

[32] Subhash Khot. On the power of unique 2-prover 1-round games. In *IEEE Conference on Computational Complexity*, page 25, 2002.

[33] Subhash Khot. *New Techniques for Probabilistically Checkable Proofs and Inapproximability Results*. PhD thesis, Princeton University, Princeton, New Jersey, 2003.

[34] Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within $2-\varepsilon$. In *IEEE Conference on Computational Complexity*, pages 379–, 2003.

[35] P. Klein, A. Agrawal, R. Ravi, and S. Rao. Approximation through multicommodity flow. In *Proceedings: 31st Annual Symposium on Foundations of Computer Science*, volume 2, pages 726–737. IEEE Computer Society Press, October 1990.

[36] Lszl Lovsz and Uriel Feige. Two-prover one-round proof systems: Their power and their problems (extended abstract). In *STOC*, pages 733–744, 1992.

[37] Carsten Lund, Lance Fortnow, Howard Karloff, and Naom Nissan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[38] G. Margulis. Probabilistic characteristics of graphs with large connectivity (in russian). *Probl. Pered. Inform.*, 10(2):101–108, 1974.

[39] Elchanan Mossel, RyanODonnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality, March 13 2005.

[40] Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998.

[41] Daniel Reichman. Master's thesis, The Weizmann Institute, 2004.

[42] L. Russo. An approximate zero-one law. *Z. Wahrsch. Verw. Gebiete*, 61(1):129–139, 1982.

[43] Adi Shamir. Ip = pspace. *J. ACM*, 39(4):869–877, 1992.

[44] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[45] Jacques Stern, Lszl Babai, Sanjeev Arora, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.*, 54:317–331, 1997.

[46] Luca Trevisan. Approximation algorithms for unique games. *Electronic Colloquium on Computational Complexity (ECCC)*, 2004.

[47] Uri Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *SODA '98: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 201–210, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[48] Uri Zwick. Finding almost-satisfying assignments. In *STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 551–560, New York, NY, USA, 1998. ACM Press.