Electronic Colloquium on Computational Complexity, Report No. 119 (2011)

$2^{\log^{1-\varepsilon} n}$ Hardness for Closest Vector Problem with Preprocessing

Subhash A. Khot
New York University
New York, NY, USA.
khot@cims.nyu.edu

Preyas Popat
New York University
New York, NY, USA.
popat@cs.nyu.edu

Nisheeth K. Vishnoi
Microsoft Research
Bangalore, India
nisheeth.vishnoi@gmail.com

Abstract

We prove that for an arbitrarily small constant $\varepsilon > 0$, assuming NP$\not\subseteq$DTIME$(2^{\log^{O(1/\varepsilon)} n})$, the preprocessing versions of the closest vector problem and the nearest codeword problem are hard to approximate within a factor better than $2^{\log^{1-\varepsilon} n}$. This improves upon the previous hardness factor of $(\log n)^{\delta}$ for some $\delta > 0$ due to [AKKV05].

1 Introduction

Given an integer lattice B and a target vector t in $\mathbb{Z}^m$, the CLOSEST VECTOR PROBLEM (CVP) asks for the vector in B nearest to t under the l_p norm. All $p \geq 1$ are interesting although, $p = 2$ case has received the most attention. An integer lattice is a set of vectors $\{\sum_{i=1}^{n} \alpha_i b_i \mid \alpha_i \in \mathbb{Z}\}$, where $b_1, b_2, \ldots, b_n \in \mathbb{Z}^m$ is a set of linearly independent vectors, called the *basis* of the lattice. An important variation of CVP is the pre-processing version of the problem where the lattice B is known in advance and the algorithm is allowed arbitrary pre-processing on B before the input t is revealed. This is known as the CLOSEST VECTOR PROBLEM WITH PRE-PROCESSING (CVPP). A related problem is the NEAREST CODEWORD PROBLEM (NCP) where the input is a generator matrix C of a linear code over $\mathbb{F}_2$ and a target vector t. The goal is to find the codeword nearest to t in Hamming distance. Again, if C is known in advance and arbitrary pre-computation is allowed on it, the problem is known as the NEAREST CODEWORD PROBLEM WITH PRE-PROCESSING (NCPP).

Pre-processing problems arise in cryptography and coding theory where, typically, a publicly known lattice (or a linear error-correcting code) is being used to transmit messages across a faulty channel. The decrypting or decoding of the received word is equivalent to solving an instance of CVP for the lattice being used in the protocol. The basis of the lattice being known beforehand, it becomes imperative to understand if the performance of the decoding algorithm can be improved, or if the security of the cryptographic protocol can be compromised (see [FM04, Reg04] for more details).

Potentially, this pre-computated information could make CVPP easier than CVP. Indeed, using the so-called Korkine-Zolotarev basis, Lagarias *et al.* [LLS90] constructed an $O(n^{1.5})$ factor approximation algorithm for CVPP, which is significantly better than the best known almost-exponential

ISSN 1433-8092

$2^{O(n \log\log n/\log n)}$ approximation factor known for CVP, see [MV10, Sch87]. This $n^{1.5}$ factor was improved to n by Regev [Reg04], and subsequently to $O(\sqrt{n/\log n})$ by Aharonov and Regev [AR05].

As for the inapproximability of CVP, it was proved by Dinur *et al.* [DKRS03] that it is NP-hard to approximate to a factor within $n^{c/\log\log n}$ for some constant $c > 0$. This improved an earlier result of [ABSS97] showing that it is quasi-NP hard to approximate to a factor within $2^{\log^{1-\varepsilon} n}$ for any constant $\varepsilon > 0$. Obtaining inapproximability results for CVPP has been a more challenging task: Feige and Micciancio [FM04] proved a $5/3 - \varepsilon$ factor NP-hardness for NCPP for any constant $\varepsilon > 0$. This was improved to $3 - \varepsilon$ by Regev [Reg04]. These authors observed that a factor C hardness for NCPP implies a factor $C^{1/p}$ hardness for CVPP under the ℓ_p norm for any $1 \leq p < \infty$. Also, hardness results in the ℓ_∞ case can be obtained by using the norm-embedding technique due to Regev and Rosen [RR06].

The inapproximability results were improved in [AKKV05] who proved a factor C NP-hardness for CVPP and NCPP for any constant C. [AKKV05] showed how their result can be extended to a hardness of $(\log n)^\delta$ for some $\delta > 0$ under the assumption that NP $\not\subseteq$ DTIME($2^{poly(\log n)}$). They also give another reduction which achieves a hardness factor of $(\log n)^{1-\varepsilon}$ for NCPP for any $\varepsilon > 0$. This latter reduction is under a certain unproved hypothesis about a pre-processing version of the PCP Theorem. [1]

1.1 Main Result and Overview

The following is the main theorem of this paper.

Theorem 1.1 (Main Theorem). *Unless NP $\subseteq$ DTIME($2^{\log^{O(1/\varepsilon)} n}$),* NCPP *and* CVPP *are hard to approximate to a factor within $2^{\log^{1-\varepsilon} n}$ for an arbitrarily small constant $\varepsilon > 0$.*

This improves on the previous hardness factor of $(\log n)^\delta$ for some $\delta > 0$ due to [AKKV05] and essentially matches the almost polynomial factor inapproximability of Dinur *et al.* [DKRS03] for CVP. We emphasize that unlike the case of CVP where the best approximation algorithm achieves a factor of $2^{n \log\log n/\log n}$, the best approximation algorithm for CVPP achieves an approximation factor of $O(\sqrt{n/\log n})$.

Overview of the proof. We will show a hardness factor of $2^{\log^{1-\varepsilon} n}$ for the MINIMUM WEIGHT SOLUTION PROBLEM WITH PRE-PROCESSING (MWSPP). The input to this problem consists of a set of *fixed* linear forms described by $B_f \in \mathbb{F}_2^{l \times N}$, a set of *variable* linear forms $B_v \in \mathbb{F}_2^{l' \times N}$ and a target vector $t \in \mathbb{F}_2^l$. The goal is to find a solution $x \in \mathbb{F}_2^N$ to the system $B_f x = t$, which minimizes the Hamming weight of the vector $B_v x$. We allow arbitrary pre-processing on all parts of the input except the vector t. It is easy to check that MWSPP is a reformulation of NCPP. Henceforth, we focus on the MWSPP problem. See Section 2.1 for preliminaries and definitions and the equivalence of MWSPP with NCPP.

Our reduction builds on the second reduction of [AKKV05] to MWSPP. The authors in [AKKV05] make a certain hypothesis about the pre-processing version of the PCP Theorem. This hypothesis

[1] The authors claim to have a proof, but do not include it in the paper.

leads to the hardness of approximation of a pre-processing version of the LABEL COVER problem. Recall that an instance of LABEL COVER is given by a bipartite graph $G = (V, W, E, [R], [S])$ and for each edge $e = (v, w) \in V \times W$, a function $\pi_e : [R] \mapsto [S]$. A labeling to the graph consists of an assignment $A : V \mapsto [R]$, $W \mapsto [S]$. An edge $e = (v, w)$ is said to be satisfied by an assignment A if $\pi_e(A(v)) = A(w)$. The value of an instance is the maximum fraction of edges that can be satisfied by any labeling.

It is a consequence of the PCP Theorem [AS98, ALM$^+$98] and Raz's Parallel Repetition Theorem [Raz98] that for every constant R, given an instance of LABEL COVER it is NP-hard to distinguish whether the value of the instance is 1 or at most $R^{-\gamma}$ for some absolute constant $\gamma > 0$. The authors in [AKKV05] show that a similar hardness holds for the LCPP problem under their hypothesis. In the LCPP problem, the label set $[R]$ for each vertex $v \in V$ comes with a partition, and an *allowable set* from the partition. The vertices in V are required to be assigned labels from their respective allowable sets. Pre-processing is allowed on all parts of the LCPP instance except for (the choice of) the allowable set for each vertex $v \in V$.

The reduction of [AKKV05] from LCPP to MWSPP uses constructions of LABEL COVER with an additional property called *smoothness*. An instance of LCPP is called δ-smooth if any two labels $i \neq i'$ of $v \in V$ map to different labels of $w \in W$ with probability at least $1 - \delta$ over the choice of neighbors w of v. The smoothness property was introduced in [Kho02] and has been used for several hardness of approximation reductions [FGRW09, GRSW10, KS11]. The hardness factor achieved by the the reduction from LCPP to MWSPP is bounded by $1/\delta$ and $1/s$ where δ is the smoothness parameter and s is the soundness of the LCPP instance. The reduction of [AKKV05] fails to give a hardness factor better than $(\log n)^{1-\varepsilon}$ for MWSPP (even assuming their hypothesis) because they use constructions of LABEL COVER which require size $n^{\Omega(1/\delta)}$ to ensure δ-smoothness. To get a better hardness factor using this reduction, we require instances of LCPP with *very good* smoothness and soundness simultaneously (relative to the size of the instance).

Our main technical contribution is to construct instances of HYPER-GRAPH LABEL COVER WITH PRE-PROCESSING (HLCPP) with *very good* soundness and smoothness. We achieve this using the low degree test [AS03, RS97], which is guaranteed to work even for very small success probability, and combine it with the sum check protocol [LFKN92], which is used to reduce the number of queries. The HLCPP problem we consider is a labeling problem similar to LCPP which differs from the latter as follows.

- The vertex set is multi-layered.
- The constraints are given by hyper-edges rather than edges. A hyper-edge contains several edges and the constraint associated with a hyper-edge is a boolean AND of constraints associated to all its edges.
- The constraints associated to edges are not *many-to-one* (projection) constraints as in LCPP but the more general *many-to-many* constraints.

We give an outline of our reduction and the analysis below.

Our reduction and analysis. We will start with an instance of $\mathbb{F}_q$-QUADRATIC CONSTRAINT SATISFACTION PROBLEM ($\mathbb{F}_q$-QCSP) for $q = 2^r$. The instance consists of k homogeneous degree 2 polynomial equations over $\mathbb{F}_q$ with n variables, where $k = \text{poly}(n)$. Each equation is of the form $p(z_1, \ldots, z_n) = v$, and further, depends on at most 3 variables. It can be shown that deciding if there is an assignment which satisfies all the equation is NP-hard (see Theorem 2.3), even when the l.h.s. of these equations (p's) are available for pre-processing. We denote the pre-processing version by $\mathbb{F}_q$-QCSPP. Our first step is to boost soundness, i.e., to reduce the fraction of satisfied equations by any assignment, while keeping the number of variables small. This is done by combining an instance of $\mathbb{F}_q$-QCSPP with an appropriate Reed-Muller code over q. We will eventually set $q = n^{\log^{O(1/\varepsilon)} n}$. This allows us to construct an $\mathbb{F}_q$-QCSPP instance where it is hard to distinguish between perfectly satisfiable instances and those where any assignment satisfies at most k/q fraction of the polynomial equations. An important feature of this reduction is that the variable set remains the same, so the number of variables is n, number of equations is q and the soundness is k/q (which is essentailly same as $1/q$). This quantitative setting of parameters is crucial for our result as the number of variables becomes negligible compared to the number of equations, and the reciprocal of the soundness. The details of this reduction appear in Section 2.2.

Each equation can now depend on almost all of the n variables and the next task is to deal with this. This is done by reducing checking an assignment for such a system of polynomial equations to the task to constructing a PCP which makes $O(\log n)$ queries and has soundness $1/q^e$ for some small constant $e > 0$. This is achieved by combining the low degree test and the sum check protocol and is the technical heart of the PCP construction.

First, the variables are identified with $\{0,1\}^{\log n}$ and embedded as a subcube of $\mathbb{F}_q^m$ where $m \stackrel{\text{def}}{=} \log n$. With this mapping, any assignment can be thought of as a function from $\{0,1\}^m$ to $\mathbb{F}_q$ and can be encoded as a polynomial over $\mathbb{F}_q^m$ of degree at most m. In this setting, if the equation was $\sum_{i,j\in[n]} c(i,j) z_i z_j = v = \sum_{\alpha,\beta\in\{0,1\}^m} c(\alpha,\beta) z(\alpha) z(\beta)$; z, c can be thought of as polynomials of degree at most m and $2m$ respectively. The Arora-Sudan points-vs-lines low degree test can be employed to ensure that z corresponds to a small list of degree m polynomials (assignments). This test is able to list-decode an assignment with success probability as low as $1/q^e$ for some small constant $e > 0$.

Once an assignment for the variables can be decoded, the task of verifying the polynomial equations $\sum_{\alpha,\beta\in\{0,1\}^m} c(\alpha,\beta) z(\alpha) z(\beta) = c$ is equivalent to performing a weighted sum check over the sub-cube $\{0,1\}^m$. We use the sum-check protocol of [LFKN92] to verify that the decoded assignment satisfies the equations. It can be shown that the soundness of the combined low degree test and the sum check protocol is at most $1/q^f$ for a small constant $f > 0$.

The result is a PCP with $2m + 2 = O(\log n)$ layers where the first $2m$ layers correspond to the sum check protocol while the last two layers correspond to the *lines* and the *points* table respectively. Only the values to be assigned to the first table by the prover will depend on the r.h.s. of the $\mathbb{F}_q$-QCSPP instance. Further, the use of low degree polynomials in encoding the assignments implicitly gives our PCP *smoothness* properties which are used in the final reduction. While the preliminaries of the low degree test and the sum check protocol appear in Sections 2.3 and 2.4 respectively, the PCP construction appears in Section 3.1.

This view of the PCP naturally leads us to constructing an HLCPP instance which is the starting point of the reduction to MWSPP and appears in Section 3.2. Finally, the reduction from HLCPP to MWSPP is similar to the reduction of [AKKV05] from LCPP to MWSPP. This appears in Section 3. For the reduction to work, we define a notion of smoothness for HLCPP which is similar to the one for LCPP and we also need that the hyper-edges of the graph satisfy a uniformity condition which is inherited from the PCP construction.

The main differences in our reduction compared to the reduction of [AKKV05] are the following:

- As mentioned earlier, the constraints in the HLCPP graph are many-to-many constraints rather than many-to-one constraints. However, the earlier reduction to MWSPP still goes through in a relatively straightforward manner.
- We manage to construct an instance of HLCPP where the smoothness and soundness are both at most $1/q^f$ for some absolute constant $f > 0$ and the size of the instance is $q^{O(m)}$. Here $m = \log n$ where n is the number of variables in the original $\mathbb{F}_q$-QCSPP instance. It is not clear that such constructions are possible if we stick to the LCPP problem. The hardness factor can be made essentially as large as $q^{1/m}$ and we set q to be very large compared to m to get a good hardness factor relative to the size of the instance. Specifically, we set $q = n^{\log^{O(1/\varepsilon)} n}$.

2 Preliminaries

In this section we state the problems we will consider and state basic results which will be useful in the construction of our PCP and the reduction.

2.1 Problem Definitions and Basic Results

We first define the quadratic CSP problem and its pre-processing version that will be a starting point of our reduction.

Definition 2.1. $\mathbb{F}_q$-QUADRATIC CSP ($\mathbb{F}_q$-QCSP): *A* $\mathbb{F}_q$-QCSP *instance* $Q \stackrel{\text{def}}{=} \left(\{p_j\}_{j=1}^m, \{c_j\}_{j=1}^m\right)$ *consists of a set of polynomial constraints over variables* $\{z_1, z_2, \ldots, z_n\}$*. Each equation is of the form*

$$p_j(z_1, z_2, \ldots, z_n) = c_j,$$

where p_j *is a homogeneous polynomial of degree* 2*, and* $c_j \in \mathbb{F}_q$*. The goal is to find an assignment to the variables* $\{z_1, z_2, \ldots, z_n\}$ *each taking a value in* $\mathbb{F}_q$ *which satisfies as many constraints as possible. Let* $OPT(Q)$ *denote the maximum, over assignments to the variables of* Q*, of the fraction of equations satisfied.*

Definition 2.2. $\mathbb{F}_q$-QUADRATIC CSP WITH PRE-PROCESSING ($\mathbb{F}_q$-QCSPP): *Given a* $\mathbb{F}_q$-QCSP *instance*

$$Q \stackrel{\text{def}}{=} \left(\{p_j\}_{j=1}^m, \{c_j\}_{j=1}^m\right)$$

over variables $\{z_1, z_2, \ldots, z_n\}$ *taking values in* $\mathbb{F}_q$*, the* $\mathbb{F}_q$-QCSPP *problem allows arbitrary pre-processing on the polynomials* $\{p_j\}_{j=1}^m$ *before the inputs* $\{c_j\}_{j=1}^m$ *are revealed.*

The following theorem can be proved in a similar manner as Theorem 4.2 in [AKKV05]. We include a proof in Section A.1.

Theorem 2.3. $\mathbb{F}_q$-QCSPP *is NP-complete for all* $q = 2^r$.

Next we define the problem that we prove is hard to approximate and show that it is equivalent to the nearest codeword problem with pre-processing.

Definition 2.4. Minimum Weight Solution Problem with Pre-processing (MWSPP): *The input to this problem consists of a set of* fixed *linear forms described by* $B_f \in \mathbb{F}_2^{l\times N}$ *, a set of* variable *linear forms* $B_v \in \mathbb{F}_2^{l'\times N}$ *and a target vector* $t \in \mathbb{F}_2^l$*. The goal is to find a solution* $x \in \mathbb{F}_2^N$ *to the system* $B_f x = t$*, which minimizes the Hamming weight of the vector* $B_v x$*. We allow arbitrary pre-processing on all parts of the input except the vector* t*.*

Definition 2.5. Nearest Codeword Problem with and without Pre-processing: *An instance of* NCP *is denoted by* (C, t) *where* $C \in \mathbb{F}_2^{n\times k}$*,* $t \in \mathbb{F}_2^n$*. The goal is to find a solution* $x \in \mathbb{F}_2^k$ *which minimizes the Hamming distance between* Cx *and* t*. In the pre-processing version,* NCPP*, we allow arbitrary pre-processing on all parts of the input except the vector* t*.*

We note that MWSP is actually same as the NCP problem in disguise, though we find it convenient to think of it as a separate problem. To see the equivalence with NCP, let x_0 be a fixed vector such that $B_f x_0 = t$, let $w = B_v x_0$ and consider the code $C \overset{\text{def}}{=} \{B_v x \mid x \text{ s.t. } B_f x = 0\}$. Then

$$\min_{x:B_f x=0} \delta(w, B_v x) = \min_{x:B_f(x+x_0)=t} \delta(B_v x_0, B_v x) = \min_{x:B_f x=t} wt(B_v x).$$

Here $\delta(\cdot,\cdot)$ measures the Hamming distance and $wt(\cdot)$ denotes the Hamming weight of a string.

Finally we note that proving the hardness for NCPP implies the hardness for CVPP.

Theorem 2.6. *[FM04] Let* $1 \le p < \infty$*. If* NCPP *(*MWSPP*) is hard to approximate to factor* f *then* CVPP*, under the* l_p *norm, is hard to approximate to factor* $f^{1/p}$*.*

2.2 Boosting Soundness through Codes

The following lemma shows how to boost soundness of the $\mathbb{F}_q$-QCSPP instance although it increases the number of variables per equation. The proof of this lemma employs Reed-Muller codes and appears in Section A.2.

Lemma 2.7. *Let* Q *be an instance of* $\mathbb{F}_q$-QCSPP *over* n *variables and* $k = poly(n)$ *equations, for any* $q = 2^r$*. There is an instance* P *of* $\mathbb{F}_q$-QCSPP *over the same set of variables and* q *equations such that:*

- *If* $OPT(Q) = 1$ *then* $OPT(P) = 1$ *and*
- *if* $OPT(Q) < 1$ *then* $OPT(P) \le k/q$*.*

In our reduction q would be $n^{\log^{O(1/\varepsilon)} n}$ and, hence, $q \gg k$.

2.3 Low Degree Test

Now we move on to developing tools necessary for keeping the number of queries in our PCP small. The first step in this is the Low Degree Test. In this section we recall the basics, the test and state the Arora-Sudan theorem which will be used.

An affine line in $\mathbb{F}_q^m$ is parametrized by $(a,b) \in (\mathbb{F}_q^m \backslash \{0\}) \times \mathbb{F}_q^m$ such that $L_{a,b} \overset{\text{def}}{=} \{ax + b : x \in \mathbb{F}_q\}$. Sometimes, we will drop the subscript if it is clear from the context. In what follows, if it helps, one can think of $m \overset{\text{def}}{=} \log n$ and $d \overset{\text{def}}{=} m$ as will be the case in our reduction. For a polynomial $g : \mathbb{F}_q^m \mapsto \mathbb{F}_q$ of degree d and a line $L \overset{\text{def}}{=} L_{a,b}$, let $g|_L$ be the restriction of g defined as $g|_L(x) \overset{\text{def}}{=} g(ax+b)$ for $x \in \mathbb{F}_q$. For two polynomials g, h we denote $g \equiv h$ if they are identical.

Definition 2.8 (Low Degree Test). *The Low Degree Test takes as input the value table of a function $f : \mathbb{F}_q^m \mapsto \mathbb{F}_q$ and for every (affine) line L of $\mathbb{F}_q^m$, the coefficients of a degree d polynomial g_L.*

The goal is to check that f is a degree d polynomial. The intention is that g_L is the restriction of f to the line L.

The test proceeds as follows:

1. *Pick a random point $x \in \mathbb{F}_q^m$ and a random line L containing x.*
2. *Test that $g_L(x) = f(x)$.*

The following theorem can be inferred from Theorem 1 and Lemma 14 in [AS03].

Theorem 2.9 (Soundness of Low Degree Test). *There are absolute constants $0 < c_1, c_2 < 1$ such that for $\delta \overset{\text{def}}{=} 1/q^{c_1}$, $l \overset{\text{def}}{=} q^{c_2}$, if $f : \mathbb{F}_q^m \mapsto \mathbb{F}_q$ passes the Low Degree Test (Definition 2.8) with probability p, then there are l degree d polynomials $f^1, f^2, \ldots, f^l$ such that :*

$$\Pr_{L,x}\left[g_L(x) = f(x) \ \ \& \ \ \exists\, j \in \{1,2,\ldots,l\} \ : \ g_L \equiv f^j|_L\right] \geq p - \delta.$$

In words, whenever the low degree test accepts, except with probability δ, the test picks a line L such that g_L corresponds to the restriction of one of the polynomials $f^1, f^2, \ldots, f^l$ to L.

We assume here that $d \ll q$ (in our application, $d \leq O(\log q)$).

2.4 Sum Check Protocol

We will also need the sum check protocol for our PCP. We start with some definition, state the test and the main theorem establishing the soundness of it. Think of $M = 2m$ and, hence, $\mathbb{F}_q^M = \mathbb{F}_q^m \times \mathbb{F}_q^m$ in the discussion below. Also one can think of $d = 4m$. We first need a notion of partial sums of polynomials.

Definition 2.10 (Partial Sums). *Let $g : \mathbb{F}_q^M \mapsto \mathbb{F}_q$ be a degree d polynomial. For every $0 \le j \le M-1$ and every $a_1, a_2, \ldots, a_j \in \mathbb{F}_q$ we define the partial sum $g_{a_1,a_2,\ldots,a_j}$ as a polynomial from $\mathbb{F}_q \mapsto \mathbb{F}_q$ as follows:*

$$g_{a_1,a_2,\ldots,a_j}(z) \overset{\text{def}}{=} \sum_{b_{j+2},\ldots,b_M \in \{0,1\}} g(a_1, a_2, \ldots, a_j, z, b_{j+2}, \ldots, b_M).$$

When $j = 0$ we denote the polynomial as $g_{\emptyset}$. When $j = M-1$, the summation is just $g(a_1, \ldots, a_{M-1}, z)$. Note that all the polynomials so defined are of degree at most d.

Definition 2.11 (Sum Check Protocol). *The Sum Check Protocol takes as input a value table for a function $g : \mathbb{F}_q^M \mapsto \mathbb{F}_q$, a target sum $c \in \mathbb{F}_q$ and for every $0 \le j \le M-1$ and every $a_1, a_2, \ldots, a_j \in \mathbb{F}_q$, the coefficients of a degree d polynomial $p_{a_1,a_2,\ldots,a_j}$. The goal is to check whether $\sum\limits_{z \in \{0,1\}^M} g(z) = c$. The intention is that g is a degree d polynomial and $p_{a_1,a_2,\ldots,a_j}$ correspond to partial sums of g as in Definition 2.10. The test proceeds by picking $x \overset{\text{def}}{=} (a_1, a_2, \ldots, a_M) \in \mathbb{F}_q^M$ uniformly at random and accepts if and only if all of the following tests pass.*

1. $p_{\emptyset}(0) + p_{\emptyset}(1) = c$.
2. *For all* $1 \le j \le M-1$, $p_{a_1,a_2,\ldots,a_{j-1}}(a_j) = p_{a_1,a_2,\ldots,a_j}(0) + p_{a_1,a_2,\ldots,a_j}(1)$.
3. $p_{a_1,a_2,\ldots,a_{M-1}}(a_M) = g(x)$.

The following theorem will be used in our reduction. The proof appears in Section A.3.

Theorem 2.12 (Soundness of Sum Check Protocol). *[LFKN92] Let $g^1, g^2, \ldots, g^l : \mathbb{F}_q^M \mapsto \mathbb{F}_q$ be degree d polynomials and $g : \mathbb{F}_q^M \mapsto \mathbb{F}_q$ an arbitrary function. Suppose for every $1 \le j \le l$, $\sum\limits_{z \in \{0,1\}^M} g^j(z) \neq c$. For $x \in \mathbb{F}_q^M$, let $\mathcal{P}(x)$ be the event that the Sum Check Protocol (Definition 2.11) accepts on inputs g, c and $p_{a_1,a_2,\ldots,a_j}$. Here x is the choice of randomness in the Sum Check Protocol.*

Then

$$\Pr_{x \in \mathbb{F}_q^M}\left[\mathcal{P}(x) \;\&\; \exists\, j \in \{1, \ldots, l\} \;:\; g(x) = g^j(x)\right] \le Mdl/q$$

In words, the probability that the Sum Check Protocol accepts when g is consistent with one of $g^1, g^2, \ldots, g^l$ is at most Mdl/q where $g^1, g^2, \ldots, g^l$ are degree d polynomials whose sum is not the required value.

3 The Reduction

The following is the main theorem about the reduction and implies Theorem 1.1 via Theorem 2.6.

Theorem 3.1. *Unless $NP \subseteq DTIME(2^{\log^{O(1/\varepsilon)} n})$,* MWSPP *is hard to approximate to factor $2^{\log^{1-\varepsilon} n}$ for an arbitrary small constant $\varepsilon > 0$.*

Towards the proof of this theorem, we will give a reduction from $\mathbb{F}_q$-QCSPP to MWSPP. The reduction proceeds in three steps:

- Reduction from $\mathbb{F}_q$-QCSPP to a PCP with low query complexity (Section 3.1).
- Construction of an HLCPP instance from the PCP (Section 3.2).
- Reduction from HLCPP to MWSPP (Section 3.3).

Finally, we will complete the proof in Section 3.4 where the choice of parameters is made.

3.1 Smooth PCP with Low Query Complexity

Note that the $\mathbb{F}_q$-QCSPP instance given by Lemma 2.7 has almost all the variables appearing in every equation. For the reduction to MWSPP we require a PCP where every query depends on a few variables. We will also crucially need a *smoothness* property from the PCP similar to the one described for LCPP in Section 1.1. To this end, we use the Low Degree Test of [AS03] and the Sum Check Protocol of [LFKN92], similarly as in [KP06].

3.1.1 Describing the PCP

Let P be the instance of $\mathbb{F}_q$-QCSPP given by Lemma 2.7 over variables $\{z_1, \ldots, z_n\}$. Let $m \overset{\text{def}}{=} \log n$. Here we assume that n is a power of 2. We think of the variables of P as being embedded into $\{0,1\}^m$ within $\mathbb{F}_q^m$. Henceforth, we will refer to the variables by their corresponding points in $\{0,1\}^m$. Thus, an assignment $A : \{0,1\}^m \mapsto \mathbb{F}_q$ to the variables can be extended to a degree m polynomial $f : \mathbb{F}_q^m \mapsto \mathbb{F}_q$ such that f is consistent with A on $\{0,1\}^m$.

Let the equations be $E_1, \ldots, E_q$ where each equation is of the form

$$E_i \equiv P_i(z_1, \ldots, z_n) = C_i \equiv \sum_{s,t \in [n]} c_i(s,t) z_s z_t = C_i \equiv \sum_{\alpha,\beta \in \{0,1\}^m} c_i(\alpha,\beta) z_\alpha z_\beta = C_i.$$

For an assignment A to $\{z_\alpha\}_{\alpha \in \{0,1\}^m}$, let f_A denote the degree m polynomial encoding A. Now, checking whether an equation $E_i \in P$ is satisfied by A amounts to checking

$$\sum_{\alpha,\beta \in \{0,1\}^m} c_i(\alpha,\beta) f_A(\alpha) f_A(\beta) = C_i.$$

Note that $c_i(\alpha, \beta)$ can be thought of as a degree $2m$ polynomial over $\mathbb{F}_q^{2m}$ and is a part of the preprocessing.

The PCP we will construct expects the following tables:

1. **Points Table:** The value of a function $f : \mathbb{F}_q^m \mapsto \mathbb{F}_q$ at every point in $\mathbb{F}_q^m$. The intention is that f is a degree m polynomial which encodes a satisfying assignment to P within $\{0,1\}^m$, i.e., for a satisfying assignment A, $f(\alpha) = f_A(\alpha)$ for all $\alpha \in \{0,1\}^m$. The size of this table is q^m.
2. **Lines Table:** The coefficients of a degree m polynomial g_L for every (affine) line L of $\mathbb{F}_q^m$. The intention is that g_L is the restriction of f on L. The size of this table is at most $q^{2m} \cdot (m+1)$.
3. **Partial Sums Table:** For every equation $E_i \in P$, every $0 \leq j \leq 2m-1$ and every $a_1, a_2, \ldots, a_j \in \mathbb{F}_q$, the coefficients of a degree $4m$ polynomial $p_{i,a_1,a_2,\ldots,a_j}$. The intention is that $p_{i,a_1,a_2,\ldots,a_j}$ correspond to partial sums of g_i (Definition 2.10) where $g_i(\alpha,\beta) \overset{\text{def}}{=} c_i(\alpha,\beta) f(\alpha) f(\beta)$ where $\alpha \overset{\text{def}}{=} (a_1, \ldots, a_m)$ and $\beta \overset{\text{def}}{=} (a_{m+1}, \ldots, a_{2m})$. Note that g_i has degree at most $4m$ and the size of the j-th partial sum table is $q \cdot q^j \cdot (4m+1)$.

PCP Test:

Pick equation $E_i \in P$ uniformly at random. Pick $\alpha \overset{\text{def}}{=} (a_1, a_2, \ldots, a_m) \in \mathbb{F}_q^m$, $\beta \overset{\text{def}}{=} (a_{m+1}, a_{m+2}, \ldots, a_{2m}) \in \mathbb{F}_q^m$ uniformly at random. Let L be the line passing through α and β. Read the following values from the corresponding tables:

- $f(\alpha), f(\beta) \in \mathbb{F}_q$ from the Points table.
- The polynomial g_L from the Lines table.
- The polynomials $p_{i,a_1,a_2,\ldots,a_j}$ from the Partial Sums table for every $0 \leq j \leq 2m-1$.

Acceptance Criteria for the Test:

Accept if and only if all of the following tests pass.

1. $g_L(\alpha) = f(\alpha)$ and $g_L(\beta) = f(\beta)$.
2. $p_{i,\emptyset}(0) + p_{i,\emptyset}(1) = C_i$.
3. For all $1 \leq j \leq 2m-1$, $p_{i,a_1,a_2,\ldots,a_{j-1}}(a_j) = p_{i,a_1,a_2,\ldots,a_j}(0) + p_{i,a_1,a_2,\ldots,a_j}(1)$.
4. $p_{i,a_1,a_2,\ldots,a_{2m-1}}(a_{2m}) = c_i(\alpha,\beta) f(\alpha) f(\beta)$.

Note that we allow arbitrary pre-processing on everything except $\{C_i\}_{i=1}^m$.

3.1.2 Completeness and Soundness of the PCP

We prove the following theorem here:

Theorem 3.2 (Low Degree and Sum Check). *Let P be a $\mathbb{F}_q$-QCSPP instance. Then*

1. *If $OPT(P) = 1$, then the PCP Test succeeds with probability* 1.
2. *If $OPT(P) \leq k/q$ and $k < q^c$ for a small enough c, then the test succeeds above with probability at most $1/q^e$ for some constant $e > 0$.*

The proof of the theorem follows from the following two lemmas.

Lemma 3.3 (Completeness). *If there exists an assignment A to $\{z_1, \dots, z_n\}$ such that $OPT(P) = 1$, i.e., $E_1, \dots, E_q$ are all satisfied, then there is an assignment to all the tables such that the test accepts with probability* 1.

Proof. We let $f \overset{\text{def}}{=} f_A$, $g_L \overset{\text{def}}{=} f_A|_L$ for all L, and for all $i \in [q]$, $0 \leq j \leq 2m-1$, and $a_1, \dots, a_j \in \mathbb{F}_q$,

$$p_{i,a_1,\dots,a_j} \overset{\text{def}}{=} \sum_{b_{j+2},\dots,b_{2m} \in \{0,1\}} h_i(a_1, \dots, a_j, z, b_{j+2}, \dots, b_{2m}),$$

where $h_i(x, y)$ is the polynomial of degree at most $4m$ representing $c_i(x, y) f_A(x) f_A(y)$. It is clear that the test succeeds with probability 1. □

Lemma 3.4 (Soundness). *There is an absolute constant $e > 0$ such that if $OPT(P) \leq k/q$ and $k < q^c$ for a small enough c, then the PCP described above has soundness at most $1/q^e$.*

Proof. We first observe that Step 1 of the protocol is equivalent to running a low degree test (Definition 2.8) on L and α with input tables g_L and f respectively. This is because the choice of β is independent of α and uniform in $\mathbb{F}_q^m$. Let $0 < c_1, c_2 < 1$ be the constants given by Theorem 2.9. Let $f^1, f^2, \dots, f^l$ be the list of $l \overset{\text{def}}{=} q^{c_2}$ polynomials promised by Theorem 2.9.

The following events can happen on a run of the PCP:

1. The low degree test between L and α fails. That is, $g_L(\alpha) \neq f(\alpha)$. In this case, the PCP does not accept.
2. $g_L(\beta) \neq f(\beta)$. In this case, the PCP does not accept.
3. The low degree test accepts ($g_L(\alpha) = f(\alpha)$) but there is no $1 \leq i \leq l$ such that $g_L \equiv f^i|_L$. By theorem 2.9, this happens with probability at most $\delta \overset{\text{def}}{=} 1/q^{c_1}$.

If none of the events listed above occur, then we have that g_L is the restriction of f_j for some $1 \leq j \leq l$. Also, since Step 1 accepts, we must have $f(\alpha) = f^j(\alpha)$ and $f(\beta) = f^j(\beta)$.

Let E_i be an equation not satisfied by any f^j for $1 \leq j \leq l$. Note that Steps 2, 3 and 4 are equivalent to running the Sum Check Protocol (Definition 2.11) on $g_i : \mathbb{F}_q^{2m} \mapsto \mathbb{F}_q$ defined as $g_i(\alpha, \beta) \overset{\text{def}}{=} c_i(\alpha, \beta) f(\alpha) f(\beta)$. g_i has degree at most $4m$. Let $g_i^j(\alpha, \beta) \overset{\text{def}}{=} c_i(\alpha, \beta) f^j(\alpha) f^j(\beta)$. Finally, for $x \in \mathbb{F}_q^{2m}$, let $\mathcal{P}_i(x)$ be the event that the Sum Check Protocol accepts.

Applying Theorem 2.12,

$$\Pr_{x \in \mathbb{F}_q^{2m}} \left[\mathcal{P}_i(x) \ \& \ \exists\, j \in \{1, \ldots, l\} \ : \ g_i(x) = g_i^j(x) \right] \leq (2m) dl / q$$

Thus, when none of the events in the list occur, the PCP accepts with probability at most $(2m) \cdot (4m) \cdot l/q$ conditioned on choosing E_i. Note that every f^j may satisfy at most k of the q equations.

Thus, the total probability that the PCP accepts is at most $\delta + (1 - lk/q) \cdot O(m^2 l/q)$ and it is easy to check that by our choice of parameters this is smaller than $1/q^e$ for some absolute constant $e > 0$. □

3.2 PCP as Hyper-graph Label Cover

It will be useful to think of the PCP as a graph labeling problem. The labeling problem we consider is similar to the well-known LABEL COVER problem except for the following differences:

- The graph is not bipartite but consists of several layers, with edges between consecutive layers. In addition, there are hyper-edges which consist of several edges. The goal is to find a labeling which satisfies the maximum fraction of hyper-edges, where the constraint corresponding to a hyper-edge is the logical AND of the constraint corresponding to each of its edges.
- The constraints corresponding to edges are not projection constraints as in the case of LABEL COVER, but the more general *many-to-many* constraints. For an edge $e = (u, v)$, a many-to-many constraint is described by an ordered partition of the label set of u and the label set of v such that the constraint is satisfied if and only if both u and v receive labels from matching partitions. Formally, let $e = (u, v)$ be an edge and $[R_u]$, $[R_v]$ be the label sets of vertices u and v. Then the many-to-many constraint is described by a pair of maps $\pi_e : [R_u] \mapsto [R_e]$, $\sigma_e : [R_v] \mapsto [R_e]$ where $[R_e]$ is a label set associated to e. A label l to u and a label l' to v is said to satisfy edge e if $\pi_e(l) = \sigma_e(l')$.

We now formally describe the HYPER-GRAPH LABEL COVER problem. While the term HYPER-GRAPH LABEL COVER can be potentially used for a more general class of problems, in this paper we restrict our attention to a very special class of graphs useful for our reduction.

Definition 3.5. HYPER-GRAPH LABEL COVER PROBLEM

An instance $G(V, E, \mathcal{E}, [R_0], [R_1], \ldots, [R_{2m+1}], \{\pi_e, \sigma_e\}_{e \in E}, \{\mathcal{S}_v, S_v\}_{v \in \mathcal{L}_0})$ *of* HYPER-GRAPH LABEL COVER *consists of:*

- *A graph* $G(V, E)$*. The vertices are partitioned into* $2m + 2$ *disjoint layers,* $V \overset{\text{def}}{=} \mathcal{L}_0 \cup \mathcal{L}_1 \cup \cdots \cup \mathcal{L}_{2m+1}$*. The edges in E are always between a vertex in* $\mathcal{L}_i$ *and a vertex in* $\mathcal{L}_{i+1}$ *for some i.*
- *Label sets* $[R_i]$ *for vertices in layer* $\mathcal{L}_i$*. Furthermore, for every vertex* $v \in \mathcal{L}_0$*, there is a partition* $\mathcal{S}_v$ *of* $[R_0]$ *and an* **allowable set** *of labels* $S_v \in \mathcal{S}_v$*.*
- *A many-to-many constraint for every edge. Let* $e = (u, v)$ *be an edge where* $u \in \mathcal{L}_i$*,* $v \in \mathcal{L}_{i+1}$*. The instance contains projections* $\pi_e : [R_i] \mapsto [R_e]$*,* $\sigma_e : [R_{i+1}] \mapsto [R_e]$*. A labeling* (l, l') *to* (u, v) *is said to satisfy e if* $\pi_e(l) = \sigma_e(l')$*.*
- *A set of hyper-edges* $\mathcal{E}$*. Every hyper-edge consists of one vertex from the first* $2m + 1$ *layers and two vertices from the last layer, such that there is an edge between any pair of vertices in adjacent layers. A labeling to the graph satisfies a hyper-edge if all the edges contained in it are satisfied.*

The goal is to find a labeling to the vertices which satisfies the maximum fraction of hyper-edges. Vertices in $\mathcal{L}_i$ *are required to receive a label from* $[R_i]$*. Furthermore, vertices in* $\mathcal{L}_0$ *are required to receive labels from their* **allowable set***.*

We also define a pre-processing version of the HYPER-GRAPH LABEL COVER PROBLEM similar to the LCPP problem of [AKKV05].

Definition 3.6. HYPER-GRAPH LABEL COVER PROBLEM WITH PRE-PROCESSING (HLCPP)

Given an instance $G(V, E, \mathcal{E}, [R_0], [R_1], \ldots, [R_{2m+1}], \{\pi_e, \sigma_e\}_{e \in E}, \{\mathcal{S}_v, S_v\}_{v \in \mathcal{L}_0})$ *of* HYPER-GRAPH LABEL COVER*, the* HLCPP *problem allows arbitrary pre-processing on all parts of the input except the allowable sets* $\{S_v\}_{v \in \mathcal{L}_0}$*.*

We will need a notion of smoothness similar to the definition of SMOOTH LABEL COVER.

Definition 3.7. *(Smoothness)*

We say that an HLCPP *instance* $G(V, E, \mathcal{E}, [R_0], [R_1], \ldots, [R_{2m+1}], \{\pi_e, \sigma_e\}_{e \in E}, \{\mathcal{S}_v, S_v\}_{v \in \mathcal{L}_0})$ *is* δ*-smooth if for every* $0 \le i \le 2m$*,* $u \in \mathcal{L}_i$*,* $l \neq l' \in [R_i]$ *we have*

$$\Pr_{e=(u,v) \in E} \left[\pi_e(l) = \pi_e(l') \right] \le \delta$$

Here $v \in \mathcal{L}_{i+1}$ *and* (π_e, σ_e) *is the many-to-many constraint associated to e.*

Lastly, we will need that the hyper-edges of the graph are regular in a certain sense.

Definition 3.8. *(Uniformity)*

Let $G(V, E, \mathcal{E}, [R_0], [R_1], \ldots, [R_{2m+1}], \{\pi_e, \sigma_e\}_{e \in E}, \{\mathcal{S}_v, S_v\}_{v \in \mathcal{L}_0})$ be an HLCPP *instance. We say that the instance is uniform if the following conditions are satisfied:*

1. *For every $0 \leq i \leq 2m+1$, every vertex in layer $\mathcal{L}_i$ has the same number of hyper-edges passing through it.*
2. *For every $0 \leq i \leq 2m$, the following two distributions are equivalent:*
    - *Select an edge between a vertex in layer $\mathcal{L}_i$ and a vertex in layer $\mathcal{L}_{i+1}$ uniformly at random.*
    - *Select a hyper-edge $\mathcal{H} \in \mathcal{E}$ uniformly at random and then select an edge from $\mathcal{H}$ between a vertex in layer $\mathcal{L}_i$ and a vertex in layer $\mathcal{L}_{i+1}$ uniformly at random. Recall that a hyper-edge contains exactly one edge between layers $\mathcal{L}_i$ and $\mathcal{L}_{i+1}$ for $0 \leq i \leq 2m-1$ and two edges between layers $\mathcal{L}_{2m}$ and $\mathcal{L}_{2m+1}$.*

We next briefly describe how the PCP described in Section 3.1 can be thought of as an HLCPP instance.

- **Layers $\mathcal{L}_{2m}$ and $\mathcal{L}_{2m+1}$:** These are the Lines table and the Points table respectively. There is a vertex L in $\mathcal{L}_{2m}$ corresponding to every line in $\mathbb{F}_q^m$. There is a label to L for every possible univariate degree m polynomial over $\mathbb{F}_q$. Hence, the number of vertices in $\mathcal{L}_{2m}$ is at most q^{2m} and the size of the label set for each vertex is $R_{2m} = q^{m+1}$. There is a vertex α in $\mathcal{L}_{2m+1}$ corresponding to every $\alpha \in \mathbb{F}_q^m$. There is a label a to α for every possible $a \in \mathbb{F}_q$. Hence, the size of the vertex set in $\mathcal{L}_{2m+1}$ is q^m and size of the label set is $R_{2m+1} = q$.

  There is an edge between L and α if the point α belongs to the line L. The constraint between the two vertices corresponds to Step 1 of the PCP.

- **Layers $\mathcal{L}_0$ through $\mathcal{L}_{2m}$:** These are the Partial Sums table and the Lines table respectively. For $1 \leq j \leq 2m-1$, there is a vertex corresponding to $(i, a_1, a_2, \ldots, a_j)$ in $\mathcal{L}_j$ for every equation $E_i \in P$ and every $a_1, a_2, \ldots, a_j \in \mathbb{F}_q$. There is a label to $(i, a_1, a_2, \ldots, a_j)$ for every possible univariate degree $4m$ polynomial over $\mathbb{F}_q$. For $j = 0$, there is a vertex $(i, \emptyset)$ corresponding to every equation $E_i \in P$. There is a label to $(i, \emptyset)$ for every univariate degree $4m$ polynomial over $\mathbb{F}_q$. Furthermore, there is a partition of the label set into q parts, indexed by $\mathbb{F}_q$ as follows:

  $$P_a \overset{\text{def}}{=} \{\text{all polynomials } p \text{ of degree at most } 4m \text{ over } \mathbb{F}_q \text{ such that } p(0) + p(1) = a\}.$$

  The **allowable set** of labels for every vertex corresponds to the part that satisfies Step 2 of the PCP. Thus, for $0 \leq j \leq 2m-1$, the size of $\mathcal{L}_j$ is $q \cdot q^j$ while the size the label set is $R_0 = R_1 = \cdots = R_{2m-1} = q^{4m+1}$.

  For $1 \leq j \leq 2m-1$, there is an edge between a vertex $(i, a_1, a_2, \ldots, a_{j-1})$ in $\mathcal{L}_{j-1}$ and a vertex $(i', a'_1, a'_2, \ldots, a'_j)$ in $\mathcal{L}_j$ if $i = i'$ and $a_k = a'_k$ for $1 \leq k \leq j-1$. The corresponding constraints are given by Step 3 of the PCP.

There is an edge between vertex $(i, a_1, a_2, \ldots, a_{2m-1})$ in $\mathcal{L}_{2m-1}$ and vertex L in $\mathcal{L}_{2m}$ if there is an $a_m \in \mathbb{F}_q$ such that for $\alpha \overset{\text{def}}{=} (a_1, \ldots, a_m)$, $\beta \overset{\text{def}}{=} (a_{m+1}, \ldots, a_{2m})$, the line L passes through α and β. The corresponding constraints are given by Step 4 of the PCP. Note that Step 4 requires the values of the function at points α and β both of which lie on line L. Thus, a label to L specifies the values of f at α and β.

It can be checked that the constraints so defined are many-to-many constraints.[2] Note that we allow pre-processing on everything except the **allowable set** of labels for vertices in layer $\mathcal{L}_0$ as required.

It can be seen that the HLCPP instance so constructed is $4m/q$-smooth, since no two distinct degree $4m$ polynomials over $\mathbb{F}_q$ can agree on more than $4m/q$ fraction of points in $\mathbb{F}_q$.

We record this identification of the PCP with an HLCPP instance as the following theorem.

Theorem 3.9. *There is a reduction from an* $\mathbb{F}_q$-QCSPP *instance P over n variables to an* HLCPP *instance* $L = G(V, E, \mathcal{E}, [R_0], [R_1], \ldots, [R_{2m+1}], \{\pi_e, \sigma_e\}_{e \in E}, \{\mathcal{S}_v, S_v\}_{v \in \mathcal{L}_0})$ *where* $m = \log n$, *such that*

1. *If* $OPT(P) = 1$, *then* $OPT(L) = 1$.
2. *If* $OPT(P) \leq k/q$ *and* $k < q^c$ *for a small enough c, then* $OPT(L) \leq 1/q^e$ *for some constant* $e > 0$.

Furthermore, the HLCPP *instance L is* $(4m/q)$*-smooth (Definition 3.7) and uniform (Definition 3.8).*

3.3 Reduction to MWSPP

The reduction from HLCPP to MWSPP is very similar to the reduction from LCPP to MWSPP described in [AKKV05].

Let $G(V, E, \mathcal{E}, [R_0], [R_1], \ldots, [R_{2m+1}], \{\pi_e, \sigma_e\}_{e \in E}, \{\mathcal{S}_v, S_v\}_{v \in \mathcal{L}_0})$ be an instance of HLCPP. For each vertex $v \in V$ and each label l to v we have a variable $w_{v,l}$. We now describe the fixed linear forms B_f of the MWSPP instance. Below, $\bigoplus$ denotes addition over $\mathbb{F}_2$.

Vertex constraints:

1. $\forall 1 \leq j \leq 2m+1, \forall v \in \mathcal{L}_j,\ \bigoplus_{l \in [R_j]} w_{v,l} = 1$.
2. $\forall v \in \mathcal{L}_0, \forall S \in \mathcal{S}_v$,

$$\bigoplus_{l \in S} w_{v,l} = 1 \text{ if } S = S_v \text{ and } 0 \text{ otherwise.} \tag{3.1}$$

Notice that only the r.h.s. depends on the input (which is S_v).

[2] Actually, the constraint between vertices in layers $\mathcal{L}_{2m-1}$ and $\mathcal{L}_{2m}$ is not many-to-many when $c_i(\alpha, \beta) = 0$ but this happens for at most $2m/q$ fraction of vertices for every equation hence we can afford to ignore these vertices and any hyper-edges containing them.

Edge constraints:

Let $e = (u, v)$ be an edge where $u \in \mathcal{L}_i$, $v \in \mathcal{L}_{i+1}$. Let $\pi_e : [R_i] \mapsto [R_e]$, $\sigma_e : [R_{i+1}] \mapsto [R_e]$ be the projections describing the many-to-many constraint associated to e. For every element $a \in [R_e]$ we add the following fixed linear form:

$$\bigoplus_{l \in [R_i] : \pi_e(l) = a} w_{u,l} = \bigoplus_{l \in [R_{i+1}] : \sigma_e(l) = a} w_{v,l}. \tag{3.2}$$

We now describe the variable forms B_v for the MWSPP instance. Let q_j be the number of vertices in layer $\mathcal{L}_j$. Let $\tilde{q} \overset{\text{def}}{=} \prod_{j=0}^{2m+1} q_j$. For every layer $\mathcal{L}_j$, $0 \le j \le 2m+1$, every vertex $v \in \mathcal{L}_j$ and every label l to v, we have the variable form $w^j_{v,l}$ repeated $\tilde{q}/q_j$ times. This completes the description of the MWSPP instance. It remains to prove the completeness and the soundness of this reduction which we do next.

3.3.1 Soundness of the MWSPP instance

Here we show that the MWSPP instance constructed has a large gap.

Theorem 3.10 (Reduction from $\mathbb{F}_q$-QCSPP to MWSPP). *Let h be such that $1/(m^3h)^{3m} \ge 1/q^e$ for large enough m and for some fixed small constant e.*

- **Completeness:** *If P is satisfiable then the* MWSPP *instance constructed in Section 3.3 has a solution of weight at most $(2m+2) \cdot \tilde{q}$.*
- **Soundness:** *If P is such that $OPT(P) \le k/q$ then the* MWSPP *instance constructed in Section 3.3 has no solution of weight less than $h \cdot (2m+2) \cdot \tilde{q}$.*

Proof. **Completeness.** If the $\mathbb{F}_q$-QCSPP instance P is satisfiable then the HLCPP instance has a labeling which satisfies all constraints (Theorem 3.9). For an MWSPP variable $w^j_{v,l}$ corresponding to vertex v and label l to v, we let $w^j_{v,l} = 1$ if v was assigned the label l and 0 otherwise. It is easy to see that this satisfies all fixed linear forms and gives a solution of weight

$$\sum_{j=1}^{2m+1} \sum_{v \in \mathcal{L}_j} 1 \cdot \tilde{q}/q_j = \sum_{j=1}^{2m+1} q_j \cdot \tilde{q}/q_j = (2m+2) \cdot \tilde{q}.$$

Soundness. In this case we are given that $OPT(P) \le k/q$ and, hence by Theorem 3.9, any labeling to the HLCPP instance satisfies at most $1/q^e$ fraction of the hyper-edges for some small constant e. The number of hyper-edges in the instance are $|[q] \times F_q^m \times \mathbb{F}_q^m| = q^{2m+1}$. Suppose there is a solution to the MWSPP instance of weight $h \cdot (2m+2) \cdot \tilde{q}$ which satisfies all fixed linear forms. We will give a (randomized) labeling to the HLCPP instance which in expectation satisfies more than $1/(m^3h)^{3m} \ge 1/q^e$ fraction of the hyper-edges, contradicting Theorem 3.9.

Let $\{w_{v,l}\}$ be a solution of weight at most $h \cdot (2m+2) \cdot \tilde{q}$. Call a label l for v *nonzero* if $w_{v,l} = 1$. (Note that our variables are allowed only 0/1 values.) We know from our assumption that

$$\sum_{j=0}^{2m+1} \sum_{v,l} w_{v,l} \cdot \tilde{q}/q_j = h \cdot (2m+2) \cdot \tilde{q}.$$

Let n_v^j denote the number of nonzero variables for the vertex v in the j-th layer. Then the above can be written as

$$\sum_{j=0}^{2m+1} \sum_{v} n_v^j / q_j = h \cdot (2m+2).$$

Hence, for all j, $\sum_v n_v^j / q_j \leq h \cdot (2m+2)$. Hence by Markov's Inequality, for every j, the fraction of v for which $n_v^j \geq m^3 h$ is atmost $h \cdot (2m+2)/(m^3 \cdot h) \leq 3/m^2$ for large enough m. We call a label i for a vertex v *non-zero* if $w_{v,i} = 1$. We remove all vertices from the graph which have more than $r \overset{\text{def}}{=} h \cdot m^3$ non-zero labels. This removes at most $3/m^2$ fraction of vertices from each layer. Next, we remove all hyper-edges containing any vertex removed in this step. To bound this number notice that our graph has this property that number of hyper-edges per vertex of layer j is at most q^{2m+1}/q_j (by Item 1 of the uniformity property: Definition 3.8). Since number of vertices removed per layer is at most $3q_j/m^2$, the number of hyper-edges removed in layer j is at most $3q^{2m+1}/q_j$. Hence, the number of hyper-edges removed overall is at most $3 \cdot (2m+2) q^{2m+1}/m^2 \leq 9/m \cdot q^{2m+1}$ for large enough m. Thus, the total fraction of hyper-edges removed is at most $9/m$ which is negligible. Thus, we have an HLCPP instance where every vertex has at most r non-zero labels and we wish to satisfy more than $1/q^e$ fraction of the queries.

Labeling. We define a randomized labeling for the HLCPP instance: randomly assign a nonzero label independently for each vertex. This is possible as the sum (over $\mathbb{F}_2$) of the variables corresponding to each v is 1 and hence not all variables for a vertex can be 0.

The next claim shows that the expected fraction of hyper-edges satisfied is at least $r^{-3m} = (h \cdot m^3)^{-3m}$ which is larger than $1/q^e$ by our assumption.

Claim 3.11. *Conditioned on the hyper-edge not being removed, the expected fraction of hyper-edges satisfied by the randomized labeling defined above is at least r^{-3m} where $r = hm^3$.*

Proof of Claim. We first remove all edges e in the graph for which some pair of non-zero labels map to the same label via the constraint associated to e. Formally, let $e = (u, v)$ be an edge, $l \neq l'$ be two non-zero labels for u and (π_e, σ_e) be the maps describing the many-to-many constraint associated to e. We remove the edge e if $\pi_e(l) = \pi_e(l')$. Since the instance is $4m/q$-smooth (Definition 3.7), taking a union bound over all pairs of non-zero labels implies that the fraction of edges removed in the graph is at most $4mr^2/q$.

Next, we remove all hyper-edges containing any edge removed in the previous step. Using Item 2 of the uniformity property (Definition 3.8) and a union bound, it can be seen that the total fraction of hyper-edges removed is at most $3m \cdot 4mr^2/q \leq 12m^2r^2/q$, which is negligible by our choice of parameters. Thus, we have an HLCPP instance where every vertex has at most r non-zero labels and the many-to-many constraint maps all non-zero labels to distinct labels.

For a hyper-edge to be satisfied, its vertex in $\mathcal{L}_0$ should receive a label from allowable set. By Equation (3.1), there is at least one non-zero label from this set. Thus, with probability at least $1/r$, we pick an allowed label for a hyper-edge.

For $0 \leq j \leq 2m$, we will show that if we have assigned label l to vertex $u \in \mathcal{L}_j$, then the probability of assigning a consistent label to any of its neighbors in $\mathcal{L}_{j+1}$ is at least $1/r$. By a consistent label we mean one which satisfies the constraint on the edge.

Suppose we have picked a label l for a vertex $u \in \mathcal{L}_j$. We claim that the left side of Equation 3.2 is 1, since there is no non-zero label l' for u such that $\pi_e(l) = \pi_e(l')$. This means that the r.h.s. is also 1 (since the fixed linear forms are satisfied). Hence there must be a non-zero label for v which satisfies the constraint associated with the edge $e = (u, v)$, and this label is assigned to v with probability at least $1/r$ (over the random choice of a labeling). Hence, the constraint between u and v is satisfied with probability at least $1/r$.

This shows that for a fixed hyper-edge, the probability (over the randomized labeling) it is satisfied is at least $r^{-(2m+3)}$ which is the number of vertices in the hyper-edge. Thus, the expected fraction of hyper-edges satisfied is at least $r^{-(2m+3)} \geq r^{-3m}$. This completes the proof of the claim. $\square$

Noticing that by our choice of parameters $1/q^e < r^{-3m}$, we obtain a contradiction. Hence, this completes the soundness proof and, hence, the theorem. $\square$

3.4 Choice of Parameters and the Proof of Main Theorem

Proof of Theorem 3.1. Let Q be the $\mathbb{F}_q$-QCSPP instance given by Theorem 2.3 over n variables and $k = poly(n)$ equations. We apply Lemma 2.7 to get an $\mathbb{F}_q$-QCSPP instance over n variables and q equations where $q \stackrel{\text{def}}{=} 2^{(\log n)^{(4/\varepsilon)}}$. We then apply the series of reductions described in Sections 3.1, 3.2 and 3.3.

Let N be the size of the MWSPP instance constructed in Section 3.3. It can be checked that $N \leq q^{100m}$, where $m \stackrel{\text{def}}{=} \log n$ for large enough m. Hence, $N \leq q^{\log^2 n}$ for large enough n. We need m and h to satisfy

${}^{1}/{}_{r^{3m}} = {}^{1}/{}_{r^{3\log n}} = {}^{1}/{}_{(m^3h)^{3m}} \geq {}^{1}/{}_{q^e}$. This is true if $\log h \leq {}^{\log q}/{}_{\log^2 n}$ and $\log q \gg \log n \log\log n$ and n is let to be large enough.

We set $h \stackrel{\text{def}}{=} q^{\log^{-2} n}$. For a large enough positive integer $D = {}^{4}/{}_{\varepsilon}$, let q be such that $\log q \stackrel{\text{def}}{=} \log^D n$. Hence, $\log q \gg \log n \log\log n$. Moreover $\log N \leq \log^{D+2} n$ and $\log h = \log^{D-2} n$. This implies that

$$\log^{1-\varepsilon} N = \log^{(D+2)(1-\varepsilon)} n \leq \log^{D-2} n = \log h$$

Finally, $N \leq q^{\log^2 n} = 2^{\log^{O(1/\varepsilon)} n}$. Summarizing, our reduction is deterministic, the hardness factor is $2^{\log^{1-\varepsilon} N}$ and takes time $2^{\log^{O(1/\varepsilon)} n}$ and, hence, holds under the hypothesis $NP \not\subseteq DTIME(n^{\log^{O(1/\varepsilon)} n})$. $\square$

References

[ABSS97] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. Hardness of approximate optima in lattices, co des, and linear systems. *Journal of Computer and System Sciences*, 54(2):317–331, 1997.

[AKKV05] Mikhail Alekhnovich, Subhash Khot, Guy Kindler, and Nisheeth K. Vishnoi. Hardness of approximating the closest vector problem with pre-processing. In *FOCS*, pages 216–225. IEEE Computer Society, 2005.

[ALM$^+$98] 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.

[AR05] Dorit Aharonov and Oded Regev. Lattice problems in NP $\cap$ co-NP. *J. ACM*, 52:749–765, September 2005.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of np. *J. ACM*, 45(1):70–122, 1998.

[AS03] Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003.

[DKRS03] Irit Dinur, Guy Kindler, Ran Raz, and Shmuel Safra. Approximating CVP to within almost-polynomial factors is NP-hard. *Combinatorica*, 23(2):205–243, 2003.

[FGRW09] V. Feldman, V. Guruswami, P. Raghavendra, and Y. Wu. Agnostic learning of monomials by halfspaces is hard. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science*, October 2009.

[FM04] U. Feige and D. Micciancio. The inapproximability of lattice and coding problems with preprocessing. *Journal of Computer and System Sciences*, 69(1):45–67, 2004.

[GRSW10] Venkatesan Guruswami, Prasad Raghavendra, Rishi Saket, and Yi Wu. Bypassing UGC from some optimal geometric inapproximability results. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:177, 2010.

[Kho02] S. Khot. Hardness results for coloring 3-colorable 3-uniform hypergraphs. In *43rd IEEE FOCS*, pages 23–32, 2002.

[KP06] Subhash Khot and Ashok Kumar Ponnuswami. Better inapproximability results for maxclique, chromatic number and min-3lin-deletion. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (1)*, volume 4051 of *Lecture Notes in Computer Science*, pages 226–237. Springer, 2006.

[KS11] Subhash Khot and Rishi Saket. On the hardness of learning intersections of two halfspaces. *J. Comput. Syst. Sci.*, 77(1):129–141, 2011.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[LLS90] J.C. Lagarias, H.W. Lenstra, and C.P. Schnorr. Korkine-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10:333–348, 1990.

[MV10] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In *STOC*, pages 351–358, 2010.

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998.

[Reg04] Oded Regev. Improved inapproximability of lattice and coding problems with preprocessing. *IEEE Transactions on Information Theory*, 50(9):2031–2037, 2004.

[RR06] Oded Regev and Ricky Rosen. Lattice problems and norm embeddings. In *STOC*, pages 447–456, 2006.

[RS97] Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability pcp characterization of np. In *STOC*, pages 475–484, 1997.

[Sch87] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987.

A Omitted Proofs

A.1 $\mathbb{F}_q$-QCSPP is NP-complete

Theorem A.1. $\mathbb{F}_q$-QCSPP *is NP-complete for all* $q = 2^r$.

Proof. We reduce 3SAT to $\mathbb{F}_q$-QCSPP . For this proof, it is convenient to view the input for 3SAT in the following form: the input is (V, E), where $V, E \in \{0,1\}^{m \times n}$ and corresponds to a 3SAT formula $\phi = C_1 \wedge \cdots \wedge C_m$ with variables $\{x_1, \ldots, x_n\}$. Each row of V corresponds to a clause C_i and V_{ij} is 1 if and only if x_j appears in C_i. Thus, each row of V has exactly three 1's. The entry E_{ij} is 1 if and only if the variable x_j appears as a negated literal in C_i.

Since 3SAT is in NP, for every n, there is a circuit $\mathcal{C}_n$ which takes as input (V, E) and an assignment $a \in \{0,1\}^n$, such that, $\mathcal{C}_n(a, V, E) = 1$ if a is a satisfying assignment for ϕ, and 0 otherwise.

Now we present the reduction, which is exactly the same as in Theorem 4.2 of [AKKV05], except that we work over $\mathbb{F}_q$ rather than $\mathbb{F}_2$. Let (V, E) be the input corresponding to a 3SAT instance ϕ. We may assume that every gate in $\mathcal{C}_n$ has fan-in 2 and fan-out 1. For every bit in the input (a, V, E) to $\mathcal{C}_n$, there is a variable in $\mathbb{F}_q$: x_i is supposed to be assigned the i-th bit of a, x_{ij} is supposed to be assigned V_{ij}, while x'_{ij} is supposed to be assigned E_{ij}.

Associated to the output of the i-th internal gate[3] in $\mathcal{C}_n$ is a variable z_i. Further, let y_0 be the variable corresponding to the output gate which outputs whether an assignment a satisfies ϕ or not.

The computation of any gate can be written as a quadratic polynomial (over $\mathbb{F}_2$) in its inputs (call these z, z') and output (call it z''): $z'' = zz'$ for an AND gate, $z'' = 1 + (1+z)(1+z')$ for an OR gate, and $z'' = 1 + z$ for a NOT gate.

Note that $\mathbb{F}_2$ is a sub-field of $\mathbb{F}_q$ since $q = 2^r$ is a power of 2. Thus, each element of $\mathbb{F}_q$ can be naturally identified with a vector in $\mathbb{F}_2^r$ such that addition in $\mathbb{F}_q$ corresponds to vector addition and multiplication in $\mathbb{F}_q$ corresponds to taking dot products. In any such representation, the element $0 \in \mathbb{F}_q$ corresponds to the all 0's vector while the element $1 \in \mathbb{F}_q$ corresponds to the all 1's vector. The crucial observation is that the polynomials for the AND, OR and NOT gates described above act as co-ordinate wise AND, OR and NOT gates when the inputs and the output are elements of $\mathbb{F}_2^r$ ($\mathbb{F}_q$), when all computation is done over $\mathbb{F}_q$.

We write such an equation for every gate in $\mathcal{C}_n$. Each equation is of degree at-most 2 and has at-most 3 variables. Note that every such equation depends only on the description of $\mathcal{C}_n$. Finally, we add the additional set of equations $y_0 = 1$, $x_{ij} = V_{ij}$ and $x'_{ij} = E_{ij}$. Hence, we get a $\mathbb{F}_q$-QCSPP instance over the set of variables

$$\{x_i : i \in [n]\} \cup \{x_{ij} : i \in [m], j \in [n]\} \cup \{x'_{ij} : i \in [m], j \in [n]\} \cup \{z_i : 1 \le i \le \text{size}(\mathcal{C}_n)\} \cup \{y_0\}.$$

[3] A gate is said to be internal if its output is not an output of the circuit.

Notice that $\mathcal{C}_n$ can be generated by a polynomial time algorithm which is given as input 1^n. Hence, this reduction is a polynomial time reduction.

We claim that this quadratic system has a solution (over $\mathbb{F}_q$) if and only if ϕ has a satisfying solution. The corresponding claim when all variables take values in $\mathbb{F}_2$ follows by construction. Now note that if there is a solution over $\mathbb{F}_q$ then taking the last co-ordinate of the variables (when viewed as vectors over $\mathbb{F}_2^r$) is a valid solution over $\mathbb{F}_2$, since all gates act co-ordinate wise.

The reduction described above gives constraints which are of degree at most 2, but not homogeneous. This is easy to fix by introducing an auxiliary variable z_0 and adding the constraint $z_0 z_0 = 1$. We then multiply all terms of degree less than 2 by z_0.

This completes the proof of the lemma.

□

A.2 Boosting Soundness through Codes

We first need some basic definitions.

Definition A.2. Codes: *A matrix $C \in \mathbb{F}_q^{m \times k}$ is said to be a generator of the linear code $\{Cx : x \in \mathbb{F}_q^k\}$ with distance $1 - \delta$ if for any $x \neq y \in \mathbb{F}_q^k$, $C(x)$ and $C(y)$ agree on at most δm co-ordinates.*

Fact A.3 (Reed-Muller Codes). *For any q, let $\mathbb{F}_q$ be the field over q elements. There is a family of linear codes with generator matrix $C_k \in \mathbb{F}_q^{q \times k}$ with distance $1 - k/q$. These are the so called Reed Muller codes over $\mathbb{F}_q$, where the message is thought of as the coefficients of a degree k polynomial and the codeword the evaluation of this polynomial on all the points in $\mathbb{F}_q$.*

Lemma A.4. *Let Q be an instance of $\mathbb{F}_q$-QCSPP over n variables and $k = poly(n)$ equations, for any $q = 2^r$. There is an instance P of $\mathbb{F}_q$-QCSPP over the same set of variables and q equations such that:*

- *If $OPT(Q) = 1$ then $OPT(P) = 1$ and*
- *if $OPT(Q) < 1$ then $OPT(P) \leq k/q$.*

Proof. Let $R \in \mathbb{F}_q^{q \times k}$ be the Reed-Muller code matrix as in Fact A.3. Let $p_1, \ldots, p_k$ be the equations of Q and let $r \in \mathbb{F}_q^k$ be a row of R. We add the constraint $\sum_{i=1}^k r_i p_i$ to P which is a $\mathbb{F}_q$-linear combination of the equations in Q. Thus, P has q equations.

It is clear that any satisfying assignment to all equations of Q is also a satisfying assignment for all equations of P. This shows that if $OPT(Q) = 1$ then $OPT(P) = 1$.

On the other hand, suppose $OPT(Q) < 1$ and fix any assignment A to the variables of Q. An equation $e_i \in Q$ is of the form $p_i(z_1, \ldots, z_n) = c_i$. Let $v^A \in \mathbb{F}_q^k$ be defined as $v_i^A \overset{\text{def}}{=} p_i(A(z_1), \ldots, A(z_n)) -$

c_i. Since $OPT(Q) < 1$, $v^A \neq 0^k$. Thus, by the code property we have that $C_k \cdot v^A$ is zero in at most k co-ordinates. Notice that $C_k \cdot v^A$ has a 0 in a co-ordinate if and only if the corresponding equation in P is satisfied by A. Since this holds for every assignment A, the theorem follows.

□

A.3 Sum Check Protocol

Theorem A.5 (Soundness of Sum Check Protocol). *[LFKN92] Let $g^1, g^2, \ldots, g^l : \mathbb{F}_q^M \mapsto \mathbb{F}_q$ be degree d polynomials and $g : \mathbb{F}_q^M \mapsto \mathbb{F}_q$ an arbitrary function. Suppose for every $1 \leq j \leq l$, $\sum_{z \in \{0,1\}^M} g^j(z) \neq c$. For $x \in \mathbb{F}_q^M$, let $\mathcal{P}(x)$ be the event that the Sum Check Protocol (Definition 2.11) accepts on inputs g, c and $p_{a_1,a_2,\ldots,a_j}$. Here x is the choice of randomness in the Sum Check Protocol.*

Then

$$\Pr_{x \in \mathbb{F}_q^M}\left[\mathcal{P}(x) \;\&\; \exists\, j \in \{1,\ldots,l\} \;:\; g(x) = g^j(x)\right] \leq Mdl/q$$

In words, the probability that the Sum Check Protocol accepts when g is consistent with one of $g^1, g^2, \ldots, g^l$ is at most Mdl/q where $g^1, g^2, \ldots, g^l$ are degree d polynomials whose sum is not the required value.

Proof. We will prove the theorem by induction on M.

Base Case: M=1 We consider two cases:

1. $p_\emptyset = g^j_\emptyset$ for some $1 \leq j \leq l$. In this case Step 1 fails by our assumption on g^j.

2. $p_\emptyset \neq g^j_\emptyset$ for all $1 \leq j \leq l$. In this case,

$$\begin{array}{rl} & \Pr_{x \in \mathbb{F}_q}\left[\mathcal{P}(x) \;\&\; \exists\, j \in \{1,\ldots,l\} \;:\; g(x) = g^j(x)\right] \\ \leq & \Pr_{x \in \mathbb{F}_q}\left[g(x) = p_\emptyset(x) \;\&\; \exists\, j \in \{1,\ldots,l\} \;:\; g(x) = g^j(x)\right] \quad \text{(Since Step 3 accepts)} \\ = & \Pr_{x \in \mathbb{F}_q}\left[\exists\, j \in \{1,\ldots,l\} \;:\; p_\emptyset(x) = g^j(x)\right] \\ \leq & ld/q \end{array}$$

The last inequality uses the fact that any two distinct degree d polynomials can agree on at most d/q fraction of the points followed by a union bound.

Inductive Case: M = N We again consider two cases as before:

1. $p_\emptyset = g^j_\emptyset$ for some $1 \leq j \leq l$. In this case Step 1 fails by our assumption on g^j.

2. $p_\emptyset \neq g^j_\emptyset$ for all $1 \leq j \leq l$. In this case, the fraction of points $a \in \mathbb{F}_q$ such that

$$\sum_{b_2,\dots b_N \in \{0,1\}} g^j(a, b_2, \dots, b_N) = p_\emptyset(a) \tag{A.1}$$

for some $1 \leq j \leq l$ is at most ld/q.

Note that for a fixed $a \in \mathbb{F}_q$, Steps 2 and 3 are equivalent to running the Sum Check Protocol for checking

$$\sum_{b_2,\dots b_N \in \{0,1\}} g(a, b_2, \dots, b_N) = c'$$

where $c' \overset{\text{def}}{=} p_\emptyset(a)$. For $x \in \mathbb{F}_q^{N-1}$, let $\mathcal{P}_a(x)$ be the event that this protocol accepts.

If Equation A.1 does not hold for any $1 \leq j \leq l$ then we can use the inductive assumption to get

$$\Pr_{x \in \mathbb{F}_q^{N-1}} \left[\mathcal{P}_a(x) \;\; \& \;\; \exists\, j \in \{1, \dots, l\} \; : \; g(a, x) = g^j(a, x) \right] \leq (N-1)dl/q$$

Thus, the total probability of acceptance is at most $ld/q + (N-1)dl/q \leq Ndl/q$.

□