



Hardness of Reconstructing Multivariate Polynomials over Finite Fields

Parikshit Gopalan*
University of Washington.
parik@cs.washington.edu

Subhash Khot
Georgia Tech.
khot@cc.gatech.edu

Rishi Saket
Georgia Tech.
saket@cc.gatech.edu

Abstract

We study the polynomial reconstruction problem for low-degree multivariate polynomials over $\mathbb{F}[2]$. In this problem, we are given a set of points $\mathbf{x} \in \{0, 1\}^n$ and target values $f(\mathbf{x}) \in \{0, 1\}$ for each of these points, with the promise that there is a polynomial over $\mathbb{F}[2]$ of degree at most d that agrees with f at $1 - \epsilon$ fraction of the points. Our goal is to find a degree d polynomial that has good agreement with f . We show that it is NP-hard to find a polynomial that agrees with f on more than $1 - 2^{-d} + \delta$ fraction of the points for any $\epsilon, \delta > 0$. This holds even with the stronger promise that the polynomial that fits the data is in fact linear, whereas the algorithm is allowed to find a polynomial of degree d . Previously the only known hardness of approximation (or even NP-completeness) was for the case when $d = 1$, which follows from a celebrated result of Håstad [Hås01].

In the setting of Computational Learning, our result shows the hardness of (non-proper)agnostic learning of parities, where the learner is allowed a low-degree polynomial over $\mathbb{F}[2]$ as a hypothesis. This is the first non-proper hardness result for this central problem in computational learning. Our results can be extended to multivariate polynomial reconstruction over any finite field.

*Supported by Venkatesan Guruswami's Sloan foundation fellowship. Work done in part while at UT Austin supported by NSF CAREER award CCF-0643829.

1 Introduction

In the generic polynomial reconstruction problem, we are given points $\{\mathbf{x}^i\}_{i=1}^m$ and target values $f(\mathbf{x}^i)$ at those points. The goal is to find a low-degree polynomial that has good agreement with f . In addition to being a very natural problem, polynomial reconstruction has found applications in several areas of theoretical computer science including computational complexity, coding theory, derandomization and computational learning. Several important advances in Probabilistically Checkable Proofs (PCPs) and List Decoding rely on highly efficient algorithms to solve the polynomial reconstruction problem. This work addresses the complexity of the polynomial reconstruction problem for low-degree multivariate polynomials over finite fields. This version of polynomial reconstruction is encountered in several of the settings mentioned above. We elaborate on some applications of polynomial reconstruction in coding theory and computational learning.

Coding Theory: An important family of error-correcting codes known as Reed-Muller codes is obtained from low-degree multivariate polynomials over finite fields. In the Reed-Muller code $\text{RM}(d, n)$, the messages correspond to all polynomials of degree at most d in n variables, and the encoding is the vector of evaluations of the polynomial at points in \mathbb{F}^n . Hadamard codes are a class of Reed-Muller codes obtained by taking the messages to be all linear functions over \mathbb{F}^n . In the Reed-Solomon code $\text{RS}(d, n)$, the messages correspond to degree d univariate polynomials (d is super-constant), and the codewords are evaluations of the polynomial at some n points in the finite field \mathbb{F} . The decoding problems for all these codes are instances of the polynomial reconstruction problem.

A series of influential results give highly efficient list-decoding algorithms for such codes, which produce a list of all codewords that have significant agreement with the received word. The first such algorithm was given for Hadamard codes by Goldreich and Levin [GL89], and subsequently by Kushilevitz and Mansour [KM91] and Goldreich, Rubinfeld and Sudan [GRS00]. List decoding algorithms for Reed-Muller codes were given by Goldreich, Rubinfeld and Sudan [GRS00], Arora and Sudan [AS03] and Sudan, Trevisan and Vadhan [STV01]. For Reed-Solomon codes, list decoding algorithms were given by Sudan [Sud97] and Guruswami and Sudan [GS99]. Recent advances on capacity-achieving list-decodable codes by Parvaresh and Vardy [PV05] and Guruswami and Rudra [GR06] use variants of Reed-Solomon codes. These results have had great impact on both coding theory and computational complexity. In particular Reed-Muller codes have numerous applications including proof-checking and the hardness of approximation [ALM⁺98, AS98, AS03], hardness amplification [STV01], cryptography [GL89] and derandomization [TSZS01]; see [Tre04, Gur04] for more applications. Various applications require various parameter settings, for instance in PCPs, on inputs of size N , one takes $|\mathbb{F}| = (\log N)^{O(1)}$ and $n = O(\frac{\log N}{\log \log N})$.

Indeed, known algorithms for some of these codes are believed to be optimal and it is an important open problem to prove matching computational and combinatorial lower-bounds. The decoding problem for a specific code is polynomial reconstruction with an important restriction: the set of points is known to the algorithm in advance, it is only the labels $f(\mathbf{x})$ that can be generated adversarially. However, several decoding algorithms do in fact solve the general reconstruction problem [Sud97, GS99].

Computational Learning: The problem of learning parity functions over $\{0, 1\}^n$ in the presence of classification noise is a central problem in computational learning; this is another instance of multivariate polynomial reconstruction. Two kinds of noise models have been studied: in the random classification noise model, the label of each example is flipped independently with probability $\eta < \frac{1}{2}$ before it is given to the learner. In the agnostic learning model which allows worst-case noise, an adversary changes the labels of some η fraction of the points in $\{0, 1\}^n$ before the points are presented to the learner. This problem is equivalent to the well-studied problem of decoding random linear codes in coding theory.

While both these problems are widely believed to be hard, there is a considerable gap in our under-

standing of their complexity. For random classification noise, the best known algorithm due to Blum, Kalai and Wasserman runs in time $2^{O(n/\log n)}$ for any distribution [BKW03]. A $2^{O(n/\log n)}$ algorithm for learning parity with adversarial noise under the uniform distribution was given recently by Feldman *et al.* [FGKP06]. Their algorithm is a proper learning algorithm which produces a parity as hypothesis. The question of whether sub-exponential agnostic learning of parity is possible under other distributions is wide open. On one hand, Håstad shows the proper learning of parities with adversarial noise is NP-complete [Hås01]. But this does not exclude the possibility of a non-proper learning algorithm which uses some other hypothesis, such as a low-degree polynomial. Indeed, proper learning of monomials with adversarial noise is known to be NP-hard [Fel06], whereas Kalai *et al.* [KKMS05] give a $2^{O(\sqrt{n})}$ non-proper learning algorithm for all distributions which produces the sign of a real polynomial as its hypothesis. Feldman *et al.* asked whether parity with adversarial noise is hard to learn even using low-degree $\mathbb{F}[2]$ polynomials as hypothesis [FGKP06].

The problem of polynomial reconstruction arises naturally in the context of learning the circuit class AC^0 , which consists of circuits of AND, OR and NOT gates having polynomial-size and constant-depth. Linial, Mansour and Nisan showed that such circuits are learnable under the uniform distribution in quasi-polynomial time [LMN93]. The question of whether these circuits are PAC-learnable is wide open. A possible approach to this problem is suggested by the results of Razborov and Smolensky, who show that such circuits with n inputs can be approximated by multivariate polynomials of degree $(\log n)^{O(1)}$ over $\mathbb{F}[p]$ under any distribution [Raz87, Smo87]. Thus even a weak learning algorithm for such polynomials over $\mathbb{F}[p]$ with adversarial noise under arbitrary distributions would have important consequences for computational learning. It would imply a PAC-learning algorithm for AC^0 , and in fact for the larger class $AC^0[p]$, where MOD- p gates are also allowed. Currently, $AC^0[p]$ circuits are not known to be learnable even under the uniform distribution. This problem of proper learning of polynomials under arbitrary distributions with adversarial noise is a generalization of the reconstruction problem which we address.

Hardness Results: In contrast with the tremendous progress on the algorithmic side, relatively few negative results are known for polynomial reconstruction. For linear polynomials in n variables, a tight hardness result follows from the celebrated work of Håstad on solving linear equations over finite fields [Hås01]. For $d = 2$ and higher, we are unaware of even an NP-completeness result for $\mathbb{F}[2]$ or even polynomial-sized fields. Goldreich *et al.* show that the polynomial reconstruction problem is NP-complete for univariate polynomials over exponentially large fields [GRS00]. The problem of Reed-Solomon decoding was recently shown to be NP-complete [GV05] by Guruswami and Vardy, again over large fields. Cheng and Wan show some connections between Reed-Solomon decoding and the discrete log problem over finite fields [CW04].

1.1 Our Results

We consider the Polynomial Reconstruction problem $\text{POLYREC}(d, n)$ for multivariate polynomials in n variables over $\mathbb{F}[2]$ of degree at most d , for d constant. The input to this problem is a set of point-value pairs $\{\mathbf{x}^i, f(\mathbf{x}^i)\}_{i=1}^m$ where $\mathbf{x}^i \in \mathbb{F}[2]^n$ and $f(\mathbf{x}^i) \in \mathbb{F}[2]$ and a degree bound d . Our goal is to find the multivariate polynomial $P(X_1, \dots, X_n)$ of degree at most d that satisfies $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for most points \mathbf{x}^i . We will allow the possibility that the same vector \mathbf{x} is repeated multiple times (but with the same label $f(\mathbf{x})$).

If there is a polynomial $P(X_1, \dots, X_n)$ such that $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for all i , it can be found using polynomial interpolation. Assume that some polynomial agrees with $f(\mathbf{x})$ on $1 - \epsilon$ fraction of the points. Can we find a polynomial with good agreement? Our main result is that this problem is hard to approximate.

Theorem 1 *For any $\epsilon, \delta > 0$, given an instance of $\text{POLYREC}(d, n)$ over $\mathbb{F}[2]$, it is NP-hard to distinguish between the following cases:*

1. YES INSTANCE: *There is a linear polynomial satisfying $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for $1 - \epsilon$ fraction of the points.*

2. NO INSTANCE: *Every polynomial of degree at most d satisfies $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for at most $1 - 2^{-d} + \delta$ fraction of the points.*

In the case $d = 1$, our result matches the tight bound of $\frac{1}{2} + \delta$ for linear equations which follows from Håstad’s work [Hås01], but via a very different proof technique. To our knowledge, for $d \geq 2$, this is the first hardness of approximation or even NP-completeness for a fixed field. Theorem 1 gives a strong guarantee in the YES case: the polynomial fitting the data is linear. This implies the NP-hardness of agnostic learning of parity even if the learning algorithm is allowed $\mathbb{F}[2]$ polynomials of degree d for any constant d , making significant progress on the problem raised in [FGKP06].

Theorem 1 suggests limits to PAC-learning AC^0 via polynomial reconstruction, by showing that there are distributions under which the reconstruction problem is NP-hard. It implies the hardness of proper learning for low-degree polynomials, even if the learning algorithm is allowed to ask queries. Recently, Gopalan *et al.* [GKRZ07] have given a list-decoding algorithm that decode $RM(d, n)$ codes polynomials up to an error-radius $1 - 2^{-d} + \delta$ over $\mathbb{F}[2]^n$. This shows that degree d polynomials can be learnt up to error-rate $1 - 2^{-d} + \delta$ under the uniform distribution with queries. Indeed, the soundness factor of $1 - 2^{-d}$ in our result comes from the minimum distance of order d Reed-Muller codes over \mathbb{F}_2 (which is 2^{-d}). Thus, unlike for univariate reconstruction, where the specific subset of points $S \subseteq F$ (or equivalently, the distribution on points) does not affect the number of errors that can be corrected by known algorithms [Sud97, GS99], the point-set $S \subseteq \mathbb{F}^n$ is crucial in the multivariate case.

Our results can be extended to any finite field $\mathbb{F}[q]$. Let $s(d, q)$ denote the maximum over all non-zero polynomials $P(X_1, \dots, X_n)$ in $\mathbb{F}[q]$ of degree d of the probability that P is 0 at a random point in $\mathbb{F}[q]^n$. By the Schwartz-Zippel lemma, if $d = a(q - 1) + b$ for $0 \leq b \leq q - 1$, then $s(d, q) = 1 - \frac{q-b}{q^{a+1}}$. Note that the minimum distance of the order d Reed-Muller code $RM_q(d, n)$ over $\mathbb{F}[q]$ is precisely $1 - s(d, q)$.

Theorem 2 *For any $\epsilon, \delta > 0$, given an instance of POLYREC(d, n) over $\mathbb{F}[q]$, it is NP-hard to distinguish between the following cases:*

1. YES INSTANCE: *There is a linear polynomial satisfying $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for $1 - \epsilon$ fraction of the points.*
2. NO INSTANCE: *Every polynomial of degree d satisfies $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for at most $s(d, q) + \delta$ fraction of the points.*

The proof of Theorem 2 is presented in Appendix A. We note that the difficulty in proving hardness depends on the relative sizes of d and q . When $d < q - 1$, one can prove a hardness of $1 - \epsilon$ versus $\frac{d+1}{q} + \delta$ using a simple reduction from linear equations over $\mathbb{F}[q]$ which we present in Appendix B.

An interesting open problem is to improve the soundness to $\frac{1}{2} + \delta$ for all degrees over $\mathbb{F}[2]$, and $\frac{1}{q}$ over $\mathbb{F}[q]$. However it seems that this will call for some substantially new techniques or a different approach to this work.

2 Overview of the Reduction

The main technical contribution of our work is to apply the machinery of PCPs to the polynomial reconstruction problem. Our result is proved by a reduction from LABELCOVER (see Section 6 for the definition of LABELCOVER). However, the fact that polynomial reconstruction for $d \geq 2$ is not a CSP in the usual sense means that there are several obstacles to overcome. To do so, we introduce some new primitives such as Dictatorship Testing for Polynomials and Consistency Testing via Folding which we believe could be useful in other contexts. For simplicity, let us consider polynomials over $\mathbb{F}[2]$.

Dictatorship Testing for low-degree Polynomials: Like most reductions from LABELCOVER, our first goal is to give a *dictatorship test* for low-degree polynomials, using constraints of the form $\langle \mathbf{x}, f(\mathbf{x}) \rangle$ for $\mathbf{x} \in \{0, 1\}^k$. Our goal is that the polynomials X_i for $i \in [k]$, which we think of as the dictatorship of i will pass this test with good probability. On the other hand, for every polynomial $P(X_1, \dots, X_k)$ of degree d which passes the test with good probability, we wish to *decode* it to a dictatorship. While this may not always be possible, we will settle for a list of indices from $[k]$ whose length is constant (independent of k).

We propose the following test: we sample a random vector $\eta \in \{0, 1\}^k$ where each η_i is 1 with probability ε , and check that $P(\eta) = 0$. In other words, polynomials passing the test must be noise-stable at $\mathbf{0}^k$. Dictatorships pass this test with probability $1 - \varepsilon$. But there are several polynomials that will do well, for instance $X_1 X_2$ will pass with probability $1 - \varepsilon^2$. While this polynomial is *close* to a dictatorship, the polynomial $X_1(X_2 + \dots + X_k)$ which depends on all k variables passes w.p. close to $1 - \frac{\varepsilon}{2}$. Indeed, any polynomial which can be written as

$$P(X_1, \dots, X_n) = X_1 P_1(X_1, \dots, X_n) + \dots + X_c P_c(X_1, \dots, X_n)$$

where the P_i s are arbitrary polynomials of degree $d - 1$ will pass the test w.p $1 - c\varepsilon$. If we view the set of monomials as a hypergraph on $[k]$, polynomials whose hypergraphs have small vertex covers will be noise stable at $\mathbf{0}^k$. We will use this as our notion of being *close* to a dictatorship. We prove an inverse theorem: if $P(X_1, \dots, X_k)$ passes our test with good probability, the corresponding hypergraph must have a small maximum matching and hence a small vertex cover. We view this as a list-decoding of $P(X_1, \dots, X_k)$.

It is unclear why this decoding should be of any use: indeed running the decoding a second time on the same hypergraph might produce a different matching. Note however that the vertex sets of any two maximal matchings must have some intersection. Indeed, the usefulness of this decoding procedure stems from the fact that given any $d + 1$ vertex covers in a d -regular hypergraph, some two will intersect.

It is interesting to contrast this dictatorship test with Fourier based dictatorship testing [Hås01, KKMO04]. In those tests, one is allowed to query the function being tested in two or more points, but in our setting we are allowed just *one* query. What makes this possible however is the promise that the function being tested is a low-degree polynomial, as opposed to an arbitrary Boolean function. In a departure from Fourier based dictatorship testing, our analysis uses only basic facts about polynomials. However, giving a test with better might require new algebraic techniques.

Consistency Testing via Folding: Our strategy for reducing from LABELCOVER is the following: to each vertex v in the LABELCOVER instance, we assign variables X_1^v, \dots, X_k^v where k is the number of labels possible. In the YES case, if the labelling of vertices is given by $l : V \rightarrow [k]$, then we want the polynomial $\sum X_{l(v)}^v$ to satisfy most of the constraints. Further, given any polynomial Q that satisfies sufficiently many constraints, we want to be able to decode it to a label for each vertex. To assign a label for vertex v , we consider the restriction of Q to the variables X_1^v, \dots, X_k^v obtained by setting the other variables to 0, which we denote by $Q(X^v)$. We then run the decoding procedure for the dictatorship test on it and pick a random label from the list. Our hope is that this will assign labels in a way that satisfies a constant fraction of the LABELCOVER constraints.

The next gadget we need is a way of testing whether two vertices have been assigned consistent labels. For this, let us consider a toy problem where there are just two vertices and we want to test if they are assigned the same label. Following the outline above, we associate them with variables X_1, \dots, X_k and Y_1, \dots, Y_k respectively. We want a test that passes the polynomials $X_i + Y_i$. Further, we want to assign labels to each vertex based on $U(X_1, \dots, X_k) = Q(X_1, \dots, X_k, \mathbf{0}^k)$ and $V(Y_1, \dots, Y_k) = Q(\mathbf{0}^k, Y_1, \dots, Y_k)$ respectively. If Q passes our test, these labels should be the same with constant probability (independent of k). We can run the dictatorship test on each U using vectors of the form $(\eta, \mathbf{0}^k)$ and similarly on V .

Assuming they pass these tests, we want to check that they are identical polynomials after setting $X_i = Y_i$. The obvious approach is to take $\mathbf{r} \leftarrow \{0, 1\}^k$ and check that $Q(\mathbf{r}, \mathbf{r}) = 0$. But in fact this will not do, since we have no control on monomials of the form $X_i Y_j$. Indeed, for any choice of restrictions U and V , one can adjust the coefficients of the $X_i Y_j$ terms so that the polynomial Q satisfies $Q(\mathbf{r}, \mathbf{r}) = 0$. This strongly suggests that a different approach is necessary to enforce consistency.

Our solution is to enforce the consistency constraints via what we call global folding. Let us write the vector $(x_1, \dots, x_k, y_1, \dots, y_k) \in \{0, 1\}^{2k}$ in a different basis as $(x_1 + y_1, \dots, x_k + y_k, y_1, \dots, y_k)$. Observe that in this basis, the polynomials $X_i + Y_i$ that pass the test only depend on the first k co-ordinates. We will enforce this condition on every polynomial. In place of the point-value pair $\langle (\mathbf{x}, \mathbf{y}), f(\mathbf{x}, \mathbf{y}) \rangle$, we add the point-value pair $\langle (x_1 + y_1, \dots, x_k + y_k), f(\mathbf{x}, \mathbf{y}) \rangle$. Clearly, this does not hurt the completeness of the test. However, one could hope for better soundness, since we have restricted the space of polynomials from all polynomials in X_i s and Y_j s to those that only depend on $X_i + Y_i$. Equivalently, we are forcing the adversary to pick a polynomial that is constant on cosets of the subspace H defined by $X_i + Y_i = 0$. To analyze the probability that some polynomial P of k variables passes this new test, we *unfold* it and write it as $Q(X_1, \dots, X_k, Y_1, \dots, Y_k) = P(X_1 + Y_1, \dots, X_k + Y_k)$. Note that this enforces the constraint that mapping X_i to Y_i sends U to V . Thus in fact, if P passes the dictatorship tests, then our decoding will assign the same labels to u and v with some probability.

Similarly, we enforce all the LABELCOVER constraints via a suitable folding. If a solution to the LABELCOVER instance exists, it will give a linear polynomial that lies in a low dimensional subspace of all linear functions on $\{0, 1\}^{nk}$. This sub-space is defined by linear equations that encode the constraints of the LABELCOVER instance. We identify this sub-space and perform the dictatorship test for every vertex after projecting points onto it. Assume that some polynomial P in this low dimensional subspace passes our tests with good probability. To decode P , we unfold it to a polynomial Q in nk dimensions. The polynomial Q has some nice symmetry properties which encode the constraints of the label-cover instance. We exploit these symmetries to show that our decoding procedure will find a good solution to the LABELCOVER instance. The novelty of our approach is that the LABELCOVER constraints are enforced via the folding and unfolding, and not through explicit consistency tests.

This is an idealized view of our reduction, which brushes over several technical issues. The constraints that we must enforce are more complicated than equality constraints (or even permutations), they are defined in terms of projection maps. For technical reasons, we use a hypergraph version of LABELCOVER, as opposed to the usual bipartite graph version. Also, we need to ensure that the polynomials passing our dictatorship tests are not 0, this is done by another kind of folding which we call local folding. Readers familiar with Håstad's PCP will note the similarity between the folding used there and local folding.

3 Dictatorship Testing for Low-Degree Polynomials

Linear polynomials are polynomials of degree 1 with no constant. By degree d multivariate polynomials, we mean all polynomials of degree at most d . In particular it includes linear polynomials. Over $\mathbb{F}[2]$ we assume that all polynomials are multilinear. Let $\mathbf{0}^k$ and $\mathbf{1}^k$ denote the all 0s and all 1s vector respectively. We use $\eta \xleftarrow{\varepsilon} \{0, 1\}^k$ to denote sampling η from the ε -biased distribution, where each $\eta_i = 1$ independently w.p. ε . We will use $\eta \leftarrow \{0, 1\}^k$ to denote sampling from the uniform distribution.

We analyze the following test on polynomials $P(X_1, \dots, X_k)$ of degree at most d :

Algorithm 1 BASIC DICTATORSHIP TEST:
Pick $\eta \xleftarrow{\varepsilon} \{0,1\}^k$ and test if $P(\eta) = 0$.

Note that the zero polynomial passes the present test with probability 1; later we will modify the test to ensure that the polynomial is non-zero. We use the following fact about low-degree polynomials:

Fact 3 Let $P(X_1, \dots, X_k)$ be a non-zero polynomial of degree d over $\mathbb{F}[2]$. Then

$$\Pr_{\eta \xleftarrow{\varepsilon} \{0,1\}^k} [P(\eta) = 0] \leq 1 - 2^{-d}.$$

Given a polynomial $P(X_1, \dots, X_k)$, we will associate it with a hypergraph $H(P)$, with vertex set is $[k]$ and edge set E . E contains the hyperedge $e \subset [k]$ if the monomial $\prod_{i \in e} X_i$ is present in $H(P)$. The degree bound of d implies that $|e| \leq d$. If we denote the constant term by $c \in \{0,1\}$, then $P(X_1, \dots, X_k) = \sum_{e \in E} \prod_{i \in e} X_i + c$. A matching in a hypergraph is a set of independent edges (with no common vertices). It is easy to see that taking all the vertices in a maximal matching gives a vertex cover for the hypergraph.

Theorem 4 Let $P(X_1, \dots, X_k)$ be a degree d polynomial over $\mathbb{F}[2]$ that passes the Basic Dictatorship Test with probability $1 - 2^{-d} + \delta$ for some $\delta > 0$. Then the largest matching in the hypergraph $H(P)$ is of size $\frac{C}{(2\varepsilon)^d}$ where C depends only on δ . Further the constant term c in $P(X_1, \dots, X_k)$ is 0.

Proof: Rather than setting each X_i to 1 with probability ε , we will do a two-step sampling procedure, which will have the same effect:

1. Set every variable X_i to 0 independently with probability $1 - 2\varepsilon$.
2. Independently set each remaining variable to a random $\{0,1\}$ value.

It is clear that this induces the ε -biased distribution on η . Let $S \subset [k]$ be the set of indices corresponding to variables that are not set to 0 in step 1. Let X^S denote the set of these variables. The resulting polynomial $P'(X^S)$ consists of the hypergraph induced by the vertex set S . Also

$$\Pr_{\eta \xleftarrow{\varepsilon} \{0,1\}^k} [P(\eta) = 1] = \Pr_{\eta' \xleftarrow{\varepsilon} \{0,1\}^{|S|}} [P'(\eta') = 1]$$

If $P'(X^S)$ is non-zero, then since it has degree at most d , $\Pr[P'(\eta') = 1] \geq 2^{-d}$. Now if $c = 1$, then P' also has the constant term 1, hence it is a non-zero polynomial, so $\Pr[P(\eta) = 1] = \Pr[P'(\eta') = 1] \geq 2^{-d}$, which is a contradiction.

Now assume that the hypergraph $H(P)$ contains a matching M of size $|M| \geq \frac{C}{(2\varepsilon)^d}$ where the constant C will be fixed later. For each hyperedge $e \in M$, the probability that all its vertices are chosen to be in S is $(2\varepsilon)^{|e|}$. Also, since M is a matching, these events are independent for various edges. Thus the probability that none of these edges occurs in the hypergraph induced by S is bounded by

$$\prod_{e \in M} (1 - (2\varepsilon)^{|e|}) \leq (1 - (2\varepsilon)^d)^{\frac{C}{(2\varepsilon)^d}} < e^{-C}.$$

Hence, with probability $1 - e^{-C}$, the subgraph induced by S is non-empty. Conditioned on this event, $P'(X^S)$ is a non-zero polynomial of degree at most d , hence $P'(\eta') = 1$ with probability at least 2^{-d} . Thus

$$\Pr[P(\eta) = 1] \geq (1 - e^{-C}) \cdot 2^{-d}$$

For sufficiently large C , this contradicts the fact that $\Pr[P(\eta) = 1] \leq 2^{-d} - \delta$. ■

Theorem 4 suggests the following decoding procedure:

Algorithm 2 DECODING PROCEDURE FOR THE BASIC DICTATORSHIP TEST:

1. Pick a maximal matching M in $H(P)$.
2. Output a list L of all vertices in this matching.

Clearly the set L is a small vertex cover for $H(P)$. The usefulness of this decoding procedure is because of the following simple lemma.

Lemma 5 *Let $H(P)$ be a non-empty hypergraph with some edge of size d . Let L_1, \dots, L_{d+1} be $d+1$ vertex covers for $H(P)$. Then some pair L_i, L_j where $i \neq j$ has a non-empty intersection.*

If all the vertex covers are obtained by taking all the vertices of some maximal matching, then in fact any two of them have non-empty intersection. This is implied by the following Lemma:

Lemma 6 *Let $H(P)$ be a non-empty hypergraph. Let M_1 and M_2 be maximal matchings in $H(P)$. Then the vertex sets of M_1 and M_2 must intersect.*

To see why this is useful in the decoding procedure, consider the following toy problem:

Graph Decoding: Carol has a graph G on k vertices. She relabels the vertices $\sigma(1), \dots, \sigma(k)$ for some permutation $\sigma \in \mathbb{S}_k$ and gives the (relabelled) graph $\sigma(G)$ to Alice. She relabels vertices according to $\pi \in \mathbb{S}_k$ and gives $\pi(G)$ to Bob. Alice and Bob need to produce vertices i and j so that $\sigma^{-1}(i) = \pi^{-1}(j)$. They do not know σ and π , and they are not allowed to communicate.

While in general, it is hard for Alice and Bob to succeed, suppose they are promised that the maximum matching in the graph G is at most C for $C \ll k$. Then Alice and Bob can each pick a maximal matching A and B respectively in their graphs and output a random vertex from the vertex set. It is easy to see from Lemma 6 that the strategy succeeds with probability at least $\frac{1}{4C^2}$.

4 Consistency Testing via Folding

In this section, we introduce the technique of folding polynomials over affine subspaces, which we use to enforce the label-cover constraints.

Definition 1 $P(X_1, \dots, X_k)$ is 0-folded over $\mathbf{h} \in \{0, 1\}^k$ if for all $\mathbf{x} \in \{0, 1\}^n$, $P(\mathbf{x} + \mathbf{h}) = P(\mathbf{x})$.

Every polynomial is 0-folded over $\mathbf{0}$. It is clear that the set of all such vectors \mathbf{h} forms a subspace of $\{0, 1\}^k$ which we denote by H . We say that $P(X_1, \dots, X_k)$ is 0-folded over H .

Lemma 7 *Let $\dim(H) = t$. A polynomial $P(X_1, \dots, X_k)$ is 0-folded over H iff it can be written as $P(\lambda_1, \dots, \lambda_{k-t})$ where $\lambda_i = \lambda_i(X_1, \dots, X_k)$ is a linear polynomial and*

$$H = \{\mathbf{x} \in \{0, 1\}^k \mid \lambda_i(\mathbf{x}) = 0 \text{ for } 1 \leq i \leq k - t\}.$$

Proof: Firstly, consider a polynomial of the above form. Note that $\lambda_i(\mathbf{h}) = 0$, so by linearity $\lambda_i(\mathbf{x} + \mathbf{h}) = \lambda_i(\mathbf{x})$ for all $\mathbf{h} \in H$. Hence

$$P(\mathbf{x} + \mathbf{h}) = P(\lambda_1(\mathbf{x} + \mathbf{h}), \dots, \lambda_{k-t}(\mathbf{x} + \mathbf{h})) = P(\lambda_1(\mathbf{x}), \dots, \lambda_{k-t}(\mathbf{x})) = P(\mathbf{x}).$$

For the converse, assume P is 0-folded over H . Pick a basis $\mathbf{h}(1) \dots, \mathbf{h}(t)$ for H . Complete this to a basis for $\{0, 1\}^k$ by adding $k - t$ vectors $\mathbf{f}(1), \dots, \mathbf{f}(k - t)$. We can write every $\mathbf{x} \in \{0, 1\}^k$ as

$$\mathbf{x} = \sum_{i=1}^{k-t} \lambda_i \mathbf{f}(i) + \sum_{j=1}^t \mu_j \mathbf{h}(j).$$

The co-ordinates $(\lambda_1, \dots, \lambda_{k-t})$ specify the coset of H in which \mathbf{x} lies, while μ_1, \dots, μ_t specify its position inside the coset. We can rewrite P as a polynomial in these new variables. We claim that P now depends only on $P(\lambda_1, \dots, \lambda_{k-t})$. Assume for contradiction that P depends on μ_1 . Then we can find a point $\mathbf{x} = (\lambda_1, \dots, \lambda_{k-t}, \mu_1, \dots, \mu_t) \in \{0, 1\}^k$ where P is sensitive to μ_1 , meaning that

$$P(\lambda_1, \dots, \lambda_{k-t}, \mu_1, \dots, \mu_t) = 1 + P(\lambda_1, \dots, \lambda_{k-t}, 1 + \mu_1, \dots, \mu_t)$$

In the standard basis, flipping μ_1 is equivalent to adding $\mathbf{h}(1)$. Thus we have $P(\mathbf{x}) \neq P(\mathbf{x} + \mathbf{h}(1))$ which is a contradiction. \blacksquare

Definition 2 $P(X_1, \dots, X_k)$ is 1-folded over $\mathbf{g} \in \{0, 1\}^k$ if for all $\mathbf{x} \in \{0, 1\}^k$, $P(\mathbf{x} + \mathbf{g}) = 1 + P(\mathbf{x})$.

It is easy to see that the set of all such \mathbf{g} (if it is non-empty) is a coset of H . This is because if P is 1-folded over \mathbf{g} and 0-folded over H , then it is in fact 1-folded over $\mathbf{g} + H$. Conversely, if P is folded over \mathbf{g} and \mathbf{g}' , then it is 0-folded over $\mathbf{g} + \mathbf{g}'$ since $P(\mathbf{x} + \mathbf{g} + \mathbf{g}') = 1 + P(\mathbf{x} + \mathbf{g}) = P(\mathbf{x})$.

Henceforth, when we say that P is folded over $\mathbf{g} + H$, we mean that it is 0-folded over H and 1-folded over $\mathbf{g} + H$.

Lemma 8 A polynomial $P(X_1, \dots, X_k)$ is folded over $\mathbf{g} + H$ iff it can be written as $P'(\lambda_1, \dots, \lambda_{k-t-1}) + \lambda_{k-t}$ where $\lambda_i = \lambda_i(X_1, \dots, X_k)$ is a linear polynomial and

$$\mathbf{g} + H = \{\mathbf{x} \in \{0, 1\}^k \mid \lambda_i(\mathbf{x}) = 0 \text{ for } 1 \leq i \leq k - t - 1 \text{ and } \lambda_{k-t}(\mathbf{x}) = 1\} \quad (1)$$

$$H = \{\mathbf{x} \in \{0, 1\}^k \mid \lambda_i(\mathbf{x}) = 0 \text{ for } 1 \leq i \leq k - t\} \quad (2)$$

Proof: Given a polynomial of this form, it is easy to see that $P(\mathbf{x} + \mathbf{h}) = 0$ for $\mathbf{h} \in H$, whereas $P(\mathbf{x} + \mathbf{g}') = 1 + P(\mathbf{x})$ for any \mathbf{g}' in $\mathbf{g} + H$.

For the converse, assume P is folded over $\mathbf{g} + H$. Pick a basis $\mathbf{h}(1) \dots, \mathbf{h}(t)$ for H . Complete this to a basis for $\{0, 1\}^k$ by adding \mathbf{g} and $k - t - 1$ vectors $\mathbf{f}(1), \dots, \mathbf{f}(k - t - 1)$. We can write $\mathbf{x} \in \{0, 1\}^k$ as

$$\mathbf{x} = \sum_{i=1}^{k-t-1} \lambda_i \mathbf{f}(i) + \lambda_{k-t} \mathbf{g} + \sum_{j=1}^t \mu_j \mathbf{h}(j).$$

It is clear that in this basis, $\mathbf{g} + H$ and H are described by Equations 1 and 2 respectively. By Lemma 7, P can be written as $P(\lambda_1, \dots, \lambda_{k-t})$. Further, the condition $P(\mathbf{x} + \mathbf{g}) = P(\mathbf{x}) + 1$ implies that

$$P(\lambda_1, \dots, \lambda_{k-t}) = P(\lambda_1, \dots, \lambda_{k-t-1}, 0) + \lambda_{k-t}.$$

We can check this by substituting values for λ_{k-t} . Setting $P'(\lambda_1, \dots, \lambda_{k-t-1}) = P(\lambda_1, \dots, \lambda_{k-t-1}, 0)$ proves the claim. \blacksquare

4.1 Testing Equality via Folding

Our next goal is to design a test to check if two vertices have been assigned the same labels. We will do this using folding. Given vertices u and v , each with a label in $[k]$, we wish to check if they have the same label. We assign variables X_1, \dots, X_k to vertex u , Y_1, \dots, Y_k to v . If both vertices have the label i assigned to them we expect the polynomial $X_i + Y_i$; so our test should accept all such polynomials. The decoding procedure labels u by looking at the restriction of Q to X_1, \dots, X_k , and labels v by looking at the restriction to Y_1, \dots, Y_k . If the test accepts some polynomial Q with non-trivial probability, we want the same label assigned to both the vertices.

Define the polynomial $D_i = X_i + Y_i$ and let \mathcal{D} denote the set of all such polynomials. These polynomials are 0-folded over the subspace H of $\{0, 1\}^{2k}$ which is defined by $X_i + Y_i = 0$ for all i , which consists of the vectors (\mathbf{z}, \mathbf{z}) for $\mathbf{z} \in \{0, 1\}^k$. We want to enforce this condition on the polynomials being tested, which means they should have the form stated in Lemma 7.

This is done by a suitable projection. Pick a basis $\mathbf{h}(1), \dots, \mathbf{h}(k)$ for H and complete it to a basis F of $\{0, 1\}^{2k}$ by adding $\mathbf{f}(1), \dots, \mathbf{f}(k)$. We can write $(\mathbf{x}, \mathbf{y}) \in \{0, 1\}^{2k}$ in this basis as $(\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$. Our test will be on polynomials $P(\lambda_1, \dots, \lambda_k)$ of degree d . We will run the basic dictatorship test on each vertex. Our test proceeds by generating points in $\{0, 1\}^{2k}$, writing them in the F -basis and projecting onto $(\lambda_1, \dots, \lambda_k)$ and testing the polynomial P at these points in $\{0, 1\}^k$.

Algorithm 3 EQUALITY TEST:

1. For vertex u , pick $\eta \xleftarrow{\varepsilon} \{0, 1\}^k$.
2. Write $(\eta, \mathbf{0}^k) = (\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$ and test if $P(\lambda_1, \dots, \lambda_k) = 0$.
3. For vertex v , pick $\eta' \xleftarrow{\varepsilon} \{0, 1\}^k$.
4. Write $(\mathbf{0}^k, \eta') = (\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$ and test if $P(\lambda_1, \dots, \lambda_k) = 0$.

In order to analyze the test, we *unfold* P and rewrite it as a polynomial in $X_1, \dots, X_k, Y_1, \dots, Y_k$ by substituting for each λ_i . We observe that folding enforces the following symmetry on P :

Claim 9 *The polynomial P satisfies $P(\mathbf{x}, \mathbf{y}) = P(\mathbf{y}, \mathbf{x})$ for $\mathbf{x}, \mathbf{y} \in \{0, 1\}^k$.*

Proof: By Lemma 7, P is folded over H , and $(\mathbf{x}, \mathbf{y}) + (\mathbf{y}, \mathbf{x}) = (\mathbf{x} + \mathbf{y}, \mathbf{x} + \mathbf{y}) \in H$. Hence (\mathbf{x}, \mathbf{y}) and (\mathbf{y}, \mathbf{x}) lie in the same coset of H . ■

Algorithm 4 DECODING PROCEDURE FOR THE EQUALITY TEST:

1. Rewrite $P(\lambda_1, \dots, \lambda_k)$ as a polynomial in $X_1, \dots, X_k, Y_1, \dots, Y_k$.
2. Run Algorithm 2 on $P(X_1, \dots, X_k, \mathbf{0}^k)$ to get list $L(u)$.
3. Run Algorithm 2 on $P(\mathbf{0}^k, Y_1, \dots, Y_k)$ to get list $L(v)$.
4. Assign $l(u) \leftarrow L(u)$ and $l(v) \leftarrow L(v)$.

In order to analyze this procedure, let us define the polynomials $U(X_1, \dots, X_k) = P(X_1, \dots, X_k, \mathbf{0}^k)$, and $V(Y_1, \dots, Y_k) = P(\mathbf{0}^k, Y_1, \dots, Y_k)$. The key observation is that P being independent of H forces the polynomials U and V to be identical.

Lemma 10 *We have $U(Z_1, \dots, Z_k) = V(Z_1, \dots, Z_k)$.*

Proof: The polynomials U and V each define a functions $\{0, 1\}^k \rightarrow \{0, 1\}$ given by

$$U(\mathbf{z}) = P(\mathbf{z}, \mathbf{0}^k), \quad V(\mathbf{z}) = P(\mathbf{0}^k, \mathbf{z}).$$

By Claim 9, $P(\mathbf{z}, \mathbf{0}^k) = P(\mathbf{0}^k, \mathbf{z})$, hence $U = V$ as functions, and hence also as polynomials. \blacksquare

Theorem 11 *Let $P(\lambda_1, \dots, \lambda_k)$ be a degree d polynomial that passes the Folded Tests for both u and v with probability at least $1 - 2^{-d} + \delta$. Then $l(u) = l(v)$ with constant probability (depending on d, δ).*

Proof: Recall that for $Q(X_1, \dots, X_k)$, $H(Q)$ denotes the hypergraph on $[k]$ corresponding to the monomials in Q . By Lemma 10, $H(U) = H(V)$. Performing the basic dictatorship test on $U(X_1, \dots, X_k)$ is equivalent to testing if $P(\eta, \mathbf{0}^k) = 0$, which is the same as testing that $P(\lambda_1, \dots, \lambda_k) = 0$ for $(\eta, \mathbf{0}^k) = (\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$. Similarly, the basic dictatorship test on $V(Y_1, \dots, Y_k)$ is the same as testing whether $P(\mathbf{0}^k, \eta') = 0$. Since both these tests succeed with probability $1 - 2^{-d} + \delta$, each of $L(U)$ and $L(V)$ is a maximal matching in $H(U) = H(V)$ of constant size. Thus by Lemma 6 choosing a random label from each results in a common label with constant probability. \blacksquare

4.2 Enforcing non-Emptiness

We show how one can use folding to ensure that the polynomials that pass the dictatorship test and the equality test are non-zero.

For the dictatorship test, observe that the polynomials X_i are 1-folded over $\mathbf{g} = \mathbf{1}^k$. To enforce this condition on every polynomial, choose a basis $\{\mathbf{f}(1), \dots, \mathbf{f}(k-1), \mathbf{g}\}$ for $\{0, 1\}^k$. We write vectors in this basis as

$$\mathbf{x} = \sum_{i=1}^{k-1} \lambda_i \mathbf{f}(i) + \lambda_k \mathbf{g}.$$

Polynomials which are folded over \mathbf{g} can be written as $P(\lambda_1, \dots, \lambda_{k-1}) + \lambda_k$. This suggests the following test:

Algorithm 5 FOLDED DICTATORSHIP TEST:

1. Sample $\eta \leftarrow_{\mathcal{E}} \{0, 1\}^k$, and write it as $\eta = (\lambda_1, \dots, \lambda_k)$.
2. Test if $P(\lambda_1, \dots, \lambda_{k-1}) = \lambda_k$.

To analyze this test, we define the unfolded polynomial $Q(X_1, \dots, X_k) = P(\lambda_1, \dots, \lambda_{k-1}) + \lambda_k$.

Theorem 12 *The polynomial $Q(X_1, \dots, X_k)$ is folded. The probability that $P(\lambda_1, \dots, \lambda_{k-1})$ passes the Folded Dictatorship Test equals the probability that $Q(X_1, \dots, X_k)$ passes the Basic Dictatorship Test.*

Proof: If $\mathbf{x} = (\lambda_1, \dots, \lambda_k)$, then $\mathbf{x} + \mathbf{1}^k = (\lambda_1, \dots, \lambda_{k-1}, 1 + \lambda_k)$. Hence

$$Q(\mathbf{x} + \mathbf{1}^k) = (1 + \lambda_k) + P(\lambda_1, \dots, \lambda_{k-1}) = 1 + Q(\mathbf{x})$$

so Q is folded over $\mathbf{1}^k$. In fact by Theorem 4, Q has no constant term, so $Q(\mathbf{0}^k) = 0$, $Q(\mathbf{1}^k) = 1$.

To show that Q passes the Basic Dictatorship Test, note that

$$P(\lambda_1, \dots, \lambda_{k-1}) = \lambda_k \iff P(\lambda_1, \dots, \lambda_{k-1}) + \lambda_k = 0 \iff Q(\eta) = 0.$$

\blacksquare

In the Equality test, we want to ensure that the polynomials $U(X_1, \dots, X_k)$ and $V(Y_1, \dots, Y_k)$ are both non-zero. Define $H = (\mathbf{z}, \mathbf{z})$ as before and let $\mathbf{g} = (\mathbf{1}^k, \mathbf{0}^k)$. The polynomials $X_i + Y_i$ are folded over the coset $\mathbf{g} + H$. We wish to enforce this condition on the polynomials that are being tested, which means they should have the form stated in Lemma 8. Pick a basis $F = \{\mathbf{f}(1), \dots, \mathbf{f}(k-t-1), \mathbf{g}, \mathbf{h}(1), \dots, \mathbf{h}(t)\}$ for $\{0, 1\}^{2k}$ and let

$$(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{k-t-1} \lambda_i \mathbf{f}_i + \lambda_{k-t} \mathbf{g} + \sum_{j=1}^t \mu_j \mathbf{h}(j).$$

Given a point $(\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$ in this basis, we test if $P(\lambda_1, \dots, \lambda_{k-1}) = \lambda_k$. Thus the test is on polynomials in $k-1$ variables.

Algorithm 6 FOLDED EQUALITY TEST:

1. For vertex u , pick $\eta \xleftarrow{\varepsilon} \{0, 1\}^k$.
2. Write $(\eta, \mathbf{0}^k) = (\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$ and test if $P(\lambda_1, \dots, \lambda_{k-1}) = \lambda_k$.
3. For vertex v , pick $\eta' \xleftarrow{\varepsilon} \{0, 1\}^k$.
4. Write $(\mathbf{0}^k, \eta') = (\lambda_1, \dots, \lambda_k, \mu_1, \dots, \mu_k)$ and test if $P(\lambda_1, \dots, \lambda_k) = 0$.

Define the unfolded polynomial $Q(X_1, \dots, X_k, Y_1, \dots, Y_k) = P(\lambda_1, \dots, \lambda_{k-1}) + \lambda_k$. We denote the restriction of Q to X_1, \dots, X_k by U and Y_1, \dots, Y_k by V .

Theorem 13 *The polynomials $U(X_1, \dots, X_k)$ and $V(Y_1, \dots, Y_k)$ are both folded over $\mathbf{1}^k$. If $P(\lambda_1, \dots, \lambda_{k-1})$ passes the Locally Folded Equality Test for both u and v with probability $1 - 2^{-d} + \delta$, then both $U(X_1, \dots, X_k)$ and $V(Y_1, \dots, Y_k)$ pass the Basic Dictatorship Test with probability $1 - 2^{-d} + \delta$.*

Proof: The proof that U and V pass the Dictatorship test follows that of Theorem 12.

Observe that the polynomial Q is folded over $\mathbf{g} + H$, which contains the points $\mathbf{g} = (\mathbf{1}^k, \mathbf{0}^k)$ and $\mathbf{g}' = (\mathbf{0}^k, \mathbf{1}^k)$. Thus

$$U(\mathbf{x} + \mathbf{1}^k) = Q((\mathbf{x}, \mathbf{0}^k) + \mathbf{g}) = 1 + Q((\mathbf{x}, \mathbf{0}^k)) = 1 + U(\mathbf{x}).$$

Similarly one can use \mathbf{g}' to show that V is folded over $\mathbf{1}^k$. ■

5 Consistency Testing

We will consider the following consistency problem: there are two vertices u and v , each of them is assigned a label $l(u), l(v) \in [k]$ respectively. The vertex u is assigned a projection function $\pi : [k] \rightarrow [t]$, while the vertex v is assigned a projection function $\sigma : [k] \rightarrow [t]$. The goal is to check whether the labels $l(u)$ and $l(v)$ satisfy $\pi(l(u)) = \sigma(l(v))$. We want a test that accepts all polynomials of the form $X_i + Y_j$ where $\pi(i) = \sigma(j)$. Let us denote the set of all such polynomials by \mathcal{D} . The test will specify target values for points of the form $(\mathbf{x}, \mathbf{y}) \in \{0, 1\}^{2k}$ projected onto a certain lower dimensional subspace.

We start by constructing a subspace H on which every polynomial in \mathcal{D} vanishes. Consider the subspace H defined by the equations

$$X_i + Y_j = 0, \quad \pi(i) = \sigma(j) \tag{3}$$

We would like a parametric description of this subspace, for which we need the following definition [Hås01].

Definition 3 Given a projection function $\pi : [k] \rightarrow [t]$, for $\mathbf{z} \in \{0, 1\}^t$, we define the vector $\mathbf{z} \circ \pi \in \{0, 1\}^k$ by $(\mathbf{z} \circ \pi)_i = z_{\pi(i)}$.

This gives a linear map from $\{0, 1\}^t \rightarrow \{0, 1\}^k$ since

$$(\mathbf{z}_1 + \mathbf{z}_2) \circ \pi = \mathbf{z}_1 \circ \pi + \mathbf{z}_2 \circ \pi.$$

Lemma 14 The subspace H contains the vectors $(\mathbf{z} \circ \pi, \mathbf{z} \circ \sigma)$ for $\mathbf{z} \in \{0, 1\}^t$.

Proof: We need to check that $(\mathbf{x}, \mathbf{y}) = (\mathbf{z} \circ \pi, \mathbf{z} \circ \sigma)$ satisfies $x_i + y_j = 0$ for all $\pi(i) = \sigma(j)$. But

$$x_i = (\mathbf{z} \circ \pi)_i = z_{\pi(i)}, \quad y_j = (\mathbf{z} \circ \sigma)_j = z_{\sigma(j)} \quad \text{hence } x_i = y_j.$$

In fact a simple dimension argument shows that $H = \{(\mathbf{z} \circ \pi, \mathbf{z} \circ \sigma) \mid \mathbf{z} \in \{0, 1\}^t\}$ but we will not need this fact. ■

Let $\mathbf{g} = (\mathbf{1}^k, \mathbf{0}^k)$. Every polynomial in \mathcal{D} is folded over $\mathbf{g} + H$. We pick a basis $\mathbf{h}(1), \dots, \mathbf{h}(t)$ for H and complete this to a basis F of $\{0, 1\}^{2k}$ given by $F = \{\mathbf{f}(1), \dots, \mathbf{f}(2k - t - 1), \mathbf{g}, \mathbf{h}(1), \dots, \mathbf{h}(t)\}$ for some suitable choice of $\mathbf{f}(i)$ s. Set $g = 2k - t$. Then

$$(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{g-1} \lambda_i \mathbf{f}(i) + \lambda_g \mathbf{g} + \sum_{j=1}^t \mu_j \mathbf{h}(j)$$

Algorithm 7 FOLDED CONSISTENCY TEST:

1. For vertex u , pick $\eta \xleftarrow{\varepsilon} \{0, 1\}^k$.
2. Write $(\eta, \mathbf{0}^k) = (\lambda_1, \dots, \lambda_g, \mu_1, \dots, \mu_t)$ and test if $P(\lambda_1, \dots, \lambda_{g-1}) = \lambda_g$.
3. For vertex v , pick $\eta' \xleftarrow{\varepsilon} \{0, 1\}^k$.
4. Write $(\mathbf{0}^k, \eta') = (\lambda_1, \dots, \lambda_g, \mu_1, \dots, \mu_t)$ and test if $P(\lambda_1, \dots, \lambda_{g-1}) = \lambda_g$.

Algorithm 8 DECODING PROCEDURE FOR THE FOLDED CONSISTENCY TEST:

1. Let $Q(X_1, \dots, X_k, Y_1, \dots, Y_k) = P(\lambda_1, \dots, \lambda_{g-1}) + \lambda_g$.
2. Run Algorithm 2 on $Q(X_1, \dots, X_k, \mathbf{0}^k)$ to get list $L(u)$.
3. Run Algorithm 2 on $Q(\mathbf{0}^k, Y_1, \dots, Y_k)$ to get list $L(v)$.
4. Assign $l(u) \leftarrow L(u)$ and $l(v) \leftarrow L(v)$.

As before we define the polynomials $U(X_1, \dots, X_k) = Q(X_1, \dots, X_k, \mathbf{0}^k)$ and $V(Y_1, \dots, Y_k) = Q(\mathbf{0}^k, Y_1, \dots, Y_k)$. The relation between the two polynomials enforced by folding is a bit more intricate. The key observation is that their *projections* in Z_1, \dots, Z_t obtained by replacing X_i by $Z_{\pi(i)}$ in U and Y_j by $Z_{\sigma(j)}$ in V are the same.

Lemma 15 Define the projected polynomials

$$U_\pi(Z_1, \dots, Z_t) = U(Z_{\pi(1)}, \dots, Z_{\pi(k)}), \quad V_\sigma(Z_1, \dots, Z_t) = V(Z_{\sigma(1)}, \dots, Z_{\sigma(k)}).$$

Then $U_\pi(Z_1, \dots, Z_t) = V_\sigma(Z_1, \dots, Z_t)$.

Proof: We can view U_π and V_σ as functions $\{0, 1\}^t \rightarrow \{0, 1\}$ given by

$$U_\pi(\mathbf{z}) = Q(\mathbf{z} \circ \pi, \mathbf{0}^k), \quad V_\sigma(\mathbf{z}) = Q(\mathbf{0}^k, \mathbf{z} \circ \sigma).$$

Since the polynomial Q is folded over H , it satisfies $Q(\mathbf{z} \circ \pi, \mathbf{0}^k) = Q(\mathbf{0}^k, \mathbf{z} \circ \sigma)$ since

$$(\mathbf{z} \circ \pi, \mathbf{0}^k) + (\mathbf{0}^k, \mathbf{z} \circ \sigma) = (\mathbf{z} \circ \pi, \mathbf{z} \circ \sigma) \in H.$$

Hence $U_\pi(\mathbf{z}) = V_\sigma(\mathbf{z})$ as functions, hence $U_\pi(Z_1, \dots, Z_t) = V_\sigma(Z_1, \dots, Z_t)$ as polynomials. \blacksquare

We can now analyze Algorithm 8.

Theorem 16 Define the projections of the lists $L(u)$ and $L(v)$ as $L_\pi(u) = \{\pi(i) \mid i \in L(u)\}$ and $L_\sigma(v) = \{\sigma(j) \mid j \in L(v)\}$.

1. Both $L_\pi(u)$ and $L_\sigma(v)$ are vertex covers for the hypergraph $H(U_\pi) = H(V_\sigma)$.
2. The polynomials U_π and V_σ are each folded over $\mathbf{1}^t$.
3. The probability that $P(\lambda_1, \dots, \lambda_{g-1})$ passes the folded consistency test for vertex u equals the probability that $U(X_1, \dots, X_k)$ passes the Basic Dictatorship Test.

Proof: The hypergraph $H(U_\pi)$ is obtained from $H(U)$ by identifying the vertices in $\pi^{-1}(\ell)$ for each $\ell \in [t]$. The edges in this hypergraph are those which have an odd number of pre-images in $H(U)$. Thus the projection of any vertex cover for $H(U)$ is also a vertex cover for $H(U_\pi)$. From Algorithm 2, $L(u)$ is a vertex cover for $H(U)$, so $L_\pi(u)$ is a vertex cover for $H(U_\pi)$. Similarly $L_\sigma(v)$ is a vertex cover for $H(V_\sigma)$. By Lemma 15, since $U_\pi = V_\sigma$, both polynomials define the same hypergraph.

By the same argument used for Theorem 13, we can show that U and V are folded over $\mathbf{1}^k$. But

$$U_\pi(\mathbf{z} + \mathbf{1}^t) = U((\mathbf{z} + \mathbf{1}^t) \circ \pi) = U(\mathbf{z} \circ \pi + \mathbf{1}^k) = 1 + U(\mathbf{z} \circ \pi) = 1 + U_\pi(\mathbf{z}).$$

So U_π is folded over $\mathbf{1}^t$ and similarly for V_σ . This shows that the hypergraph $H(U_\pi) = H(V_\sigma)$ is non-empty.

The proof of Part 3 follows that of Theorem 12. \blacksquare

Thus, if P passes the test then $L(u)$ and $L(v)$ are small in size, their projections are vertex-covers for the same (non-empty) hypergraph. It is natural to ask if choosing $l(u) \leftarrow L(u)$ and $l(v) \leftarrow L(v)$ gives $\pi(l(u)) = \sigma(l(v))$ with some probability. This might not be the case. The reason is that while the vertex cover $L(u)$ obtained by taking all the vertices of a maximal matching, the projection $L_\pi(u)$ need not have this structure. Thus $L_\pi(u)$ and $L_\sigma(v)$ might be disjoint vertex covers of the same hypergraph. However, the fact that they are both vertex covers together with Lemma 5 will suffice for our analysis. We note however that if $d = 1$, then the vertex covers will intersect, so the random decoding succeeds. This gives an alternative proof of Hastad's result for $d = 1$.

6 The Reduction from Label Cover

An instance of LABELCOVER($d + 1$) for $d \geq 1$ consists of a $d + 1$ -regular hypergraph (V, E) with vertex set $V = \{v_i\}_{i=1}^n$ and an edge set $E = \{e_j\}_{j=1}^m$, where $|e_j| = d + 1$. The hypergraph is connected, and any $S \subset V$ of size δn induces a constant fraction $\gamma(\delta)m$ of edges. Every vertex in V is to be assigned a label $l(v) \in [k]$. Every hyperedge $e = (v_1^e, \dots, v_{d+1}^e)$ is associated with a d -tuple of projection functions $\{\pi_i\}_{i=1}^{d+1}$

where $\pi_i : [k] \rightarrow [t]$ and $t < k$. A vertex labelling strongly satisfies edge e if $\pi_i(l(v_i^e)) = \pi_j(l(v_j^e))$ for every $v_i^e, v_j^e \in e$. A vertex labelling weakly satisfies edge e if $\pi_i(l(v_i^e)) = \pi_j(l(v_j^e))$ for some pair $v_i^e, v_j^e \in e$.

This is a slightly non-standard hypergraph version of label cover. A similar kind of acceptance predicate is used by Feige in proving the hardness of set-cover [Fei98]. The only reason we cannot use his result directly is because we need to condition that large subsets of vertices induce many edges. The following theorem is proved using a simple reduction from the standard bipartite version of LABELCOVER. We give a proof in Appendix C for completeness.

Theorem 17 *For any $\beta > 0$, given an instance of LABELCOVER($d+1$), it is NP-hard to distinguish between the following cases:*

1. YES INSTANCE : *There is some vertex labelling that strongly satisfies every edge.*
2. NO INSTANCE : *There is no vertex labelling that weakly satisfies β fraction of the edges.*

We need some notation in order to describe the reduction to POLYREC(d, n). To each vertex $v \in V$, we assign k variables X_1^v, \dots, X_k^v . Since there are a total of nk variables $X_1^{v_1}, \dots, X_k^{v_n}$, our points will be in nk dimensions, partitioned into n groups, one for each vertex, and each group having k dimensions, one for each possible vertex label. Given $\mathbf{z} \in \mathbb{Z}^{nk}$, we use \mathbf{x}^v to denote the vector in $\{0, 1\}^k$ obtained by projecting onto the co-ordinates assigned to vertex v . To a labelling l of vertices, we associate the polynomial $Q_l(X_1^{v_1}, \dots, X_k^{v_n}) = \sum_v X_{l(v)}^v$.

Our first goal is to identify a subspace H such that if l satisfies all the LABELCOVER constraints, then Q_l is 0-folded over H . Unlike for the simple tests considered so far, we do not know what the set of polynomials Q_l is, or whether it is non-empty. However, one can identify vectors that must lie in H from the constraints of the LABELCOVER instance.

Lemma 18 *Consider a pair of vertices u, w that lie in $e \in E$. Suppose the projections associated with them by e are π and σ respectively. Given $\mathbf{z} \in \{0, 1\}^t$, define the vector $\mathbf{h} = \mathbf{h}(\mathbf{z}, e, u, w) \in \{0, 1\}^{nk}$ where*

$$\mathbf{h}^v = \begin{cases} \mathbf{z} \circ \pi & \text{if } v = u \\ \mathbf{z} \circ \sigma & \text{if } v = w \\ \mathbf{0}^k & \text{otherwise.} \end{cases} \quad (4)$$

If l satisfies $\pi(l(u)) = \sigma(l(w))$ then $Q_l(\mathbf{h}) = 0$.

Proof: Note that

$$Q_l(\mathbf{h}) = \sum_{v \in V} h_{l(v)}^v = h_{l(u)}^u + h_{l(w)}^w.$$

Also

$$h_{l(u)}^u = (\mathbf{z} \circ \pi)_{l(u)} = z_{\pi(l(u))}, \quad h_{l(w)}^w = (\mathbf{z} \circ \sigma)_{l(w)} = z_{\sigma(l(w))}.$$

But $\pi(l(u)) = \sigma(l(w))$, hence $h_{l(u)}^u + h_{l(w)}^w = 0$. ■

We take H to be the span of all the vectors \mathbf{h} above, over all choices of $e \in E$, $u, w \in e$ and $\mathbf{z} \in \{0, 1\}^t$.

Let $\mathbf{g}(v) = \sum_{i=1}^k \mathbf{e}_i^v$ be the indicator for the co-ordinates of the vertex v . Let $\mathbf{g} = \mathbf{g}(v_1)$. Observe that every polynomial associated to a labelling satisfies $Q_l(\mathbf{g}(v)) = 1$.

Lemma 19 *The affine subspace $\mathbf{g} + H$ contains the vectors $\mathbf{g}(v)$ for all $v \in V$.*

Proof: Assume that $u, w \in e$ for some $e \in E$. Let π and σ denote the associated projections. Then $\mathbf{g}(u) + \mathbf{g}(w) \in H$, since this vector is obtained by setting $\mathbf{z} = \mathbf{1}^t$ in Equation 4. Since the hypergraph is connected, it follows that all the vectors $\mathbf{g}(v)$ lie in the same coset of H . ■

We will ensure that the polynomials we test are folded over $\mathbf{g} + H$. Let the dimension of the space H be h , and select a basis $\{\mathbf{h}(j)\}_{j=1}^h$ for it. Complete this to a basis F of $\{0, 1\}^{nk}$ by adding \mathbf{g} and some other vectors $\mathbf{f}(1), \dots, \mathbf{f}(nk - h - 1)$. Let $g = nk - h$. One can write any $\mathbf{x} \in \{0, 1\}^{nk}$ as

$$\mathbf{x} = \sum_{i=1}^{g-1} \lambda_i \mathbf{f}(i) + \lambda_g \mathbf{g} + \sum_{j=1}^h \mu_j \mathbf{h}(j)$$

Let $\eta(v)$ denote the random variable where each co-ordinate corresponding to vertex v is sampled from the ε -biased distribution and all other co-ordinates as 0. We now state the reduction.

Algorithm 9 LABELCOVER TEST

1. Compute the basis F described above.
2. Pick a vertex $v \leftarrow V$ and sample the vector $\eta(v)$.
3. Write $\eta(v) = (\lambda_1, \dots, \lambda_g, \mu_1, \dots, \mu_h)$.
4. Test if $P(\lambda_1, \dots, \lambda_{g-1}) = \lambda_g$.

We give a decoding procedure that uses the polynomial P to assign labels to every vertex. If P passes the test with good probability, then the resulting labelling is guaranteed to weakly satisfy a good fraction of constraints. This implies that if we reduce from a NO instance of LABELCOVER then no polynomial passes the test. Given a polynomial $Q(X_1^{v_1}, \dots, X_k^{v_n})$, for each vertex $v \in V$, we use $Q(X^v)$ to denote the restriction of Q to the variables $\{X_i^v\}_{i=1}^k$, obtained by setting all other variables to 0.

Algorithm 10 DECODING PROCEDURE FOR THE LABELCOVER TEST

1. Set $Q(X_1^{v_1}, \dots, X_k^{v_n}) = P(\lambda_1, \dots, \lambda_{g-1}) + \lambda_g$.
2. For every vertex $v \in V$,
 - 2a. Run Algorithm 2 on $Q(X^v)$ to get list $L(v)$.
 - 2b. Set $l(v) \leftarrow L(v)$.

Theorem 20 Assume that $P(\lambda_1, \dots, \lambda_{g-1})$ passes the LABELCOVER test with probability $1 - 2^{-d} + 2\delta$ for $\delta > 0$. Then the labelling $l(v)$ weakly satisfies γ' fraction of the constraints in expectation for some $\gamma'(\varepsilon, \delta, d)$.

Proof: By an averaging argument, for a δ fraction of vertices in V , the probability of passing the LABELCOVER test is at least $1 - 2^{-d} + \delta$; denote this set by S and call such vertices *good*. The *good* set of edges $E(S)$ induced by S is at least a γ fraction of all edges for some constant $\gamma(\delta)$.

Pick an edge $e \in E(S)$, and pick any two vertices $u, w \in e$. Both these will be good vertices. Let $Q(X^u, X^w)$ denote the restriction of the polynomial $Q(X_1^{v_1}, \dots, X_k^{v_n})$ to the variables $\{X_i^u, X_j^w\}_{i,j=1}^k$ obtained by setting the other variables to 0. This polynomial is 0-folded over the set of vectors $H' = (\mathbf{z} \circ \pi, \mathbf{z} \circ \sigma)$. It is folded over $\mathbf{g} + H'$ where $\mathbf{g} = (\mathbf{1}^k, \mathbf{0}^k)$. So we can apply Theorem 16 to conclude that the projections of the polynomials $Q(X^u)$ and $Q(X^w)$ under π and σ respectively are identical, and $\pi(L(u))$ and $\sigma(L(w))$ each give a vertex cover for the (non-empty) hypergraph H of this projected polynomial. Further, since u and w are good vertices, by Theorem 4 both $L(u)$ and $L(w)$ are small. Hence their projections $L_\pi(u)$ and $L_\sigma(w)$ are also small.

Since this is true for any pair of vertices in e , we have $d + 1$ vertex covers of H . But each edge of H has size at most d , so by Lemma 5 some two of them intersect, assume that these are $\pi(L(u))$ and $\sigma(L(v))$. In other words, there are labels $\ell_1 \in L(u)$ and $\ell_2 \in L(v)$ so that $\pi(\ell_1) = \sigma(\ell_2)$. Since each of these lists is of constant size (depending only on ϵ, δ, d), there is a constant probability $p = p(\epsilon, \delta, d)$ that these are the labels chosen for u and v respectively by the random decoding in Step 2b. In this case, the constraint is weakly satisfied. Thus the expected number of satisfied constraints is $\gamma'(\epsilon, \delta, d) = p \cdot \gamma$. ■

By taking the soundness β of the label cover instance to be a sufficiently small constant, we conclude that in the NO case, there is no polynomial that satisfies the LABELCOVER test with probability $1 - 2^{-d} + \delta$, else we would reach a contradiction. Finally, we need to massage the LABELCOVER test to produce an instance of POLYREC(d, n). The LABELCOVER test produces a distribution \mathcal{D} on polynomially many constraints of the form $\langle \mathbf{x}, f(\mathbf{x}) \rangle$. If for some \mathbf{x} , $\langle \mathbf{x}, 0 \rangle$ and $\langle \mathbf{x}, 1 \rangle$ each occur with non-zero probability, then transfer all the probability mass for \mathbf{x} to the label 0/1 that has more weight under \mathcal{D} . It is easy to see that the completeness and soundness can only change by ϵ . We now repeat each \mathbf{x} sufficiently many times to simulate the distribution, to get an instance of POLYREC(d, n). This proves Theorem 1.

7 Towards Optimal Inapproximability

It would be very interesting to try and improve the soundness guarantee in Theorem 1 to $\frac{1}{2} + \delta$. This problem looks to be fairly hard.

The first step is finding a better dictatorship test/analysis for low-degree polynomials, which is interesting in its own right and might require some new techniques. Consider the case $d = 2$, where the soundness of our test is $\frac{3}{4}$. Our analysis is tight: the quadratic from $Q_1 = (X_1 + \dots + X_{k/2})(X_{k/2+1} + \dots + X_k)$ passes the test w.p. $\frac{3}{4}$ and does not have a small vertex cover. However, this polynomial is not folded over 1^k , and we only need our test to work for folded polynomials. There are folded quadratic forms with no small vertex cover that pass the test with good probability: assume $k/2$ is even and set $Q_2 = (X_1 + \dots + X_{k/2})(X_{k/2+1} + \dots + X_k) + X_{k+1}$. One can show that Q_2 is folded, it passes our test with probability close to $\frac{3}{4}$, and it does not have a small vertex cover. However, it seems natural that the decoding of Q_2 ought to be the label $k + 1$. This suggests the need for a new decoding algorithm.

The next step would be to design a consistency test that solves the Graph Decoding problem from Section 3, and can also handle more sophisticated constraints involving projections (see Section 5). While the UNIQUE GAMES CONJECTURE usually gives a way to translate dictatorship tests to a hardness result, it does not help in our setting since the approach of folding the consistency constraints crucially needs perfect completeness. Indeed, translating the dictatorship tests to a hardness result might require LABELCOVER instances where the projections have special properties, which seem out of the reach of current techniques.

It would be interesting to see whether the ideas in this paper can be applied to the problem of univariate polynomial reconstruction and Reed-Solomon decoding. A first step might be to consider the setting where the field size is larger but the number of variables is logarithmic in the input size n .

Acknowledgments

We would like to thank Venkatesan Guruswami for comments that helped simplify the presentation.

References

- [ALM⁺98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.
- [AS98] S. Arora and S. Safra. Probabilistic checking of proofs : A new characterization of NP. *J. ACM*, 45(1):70–122, 1998.
- [AS03] S. Arora and M. Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003.
- [BKW03] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003.
- [CW04] Q. Cheng and D. Wan. On the list and bounded distance decodibility of Reed-Solomon codes. In *Proc. 45th IEEE Symp. on Foundations of Computer Science (FOCS'04)*, pages 335–341, 2004.
- [Fei98] U. Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, 1998.
- [Fel06] V. Feldman. Optimal hardness results for maximizing agreements with monomials. In *Proc. IEEE Conference on Computational Complexity*, pages 226–236, 2006.
- [FGKP06] V. Feldman, P. Gopalan, S. Khot, and A. K. Ponnuswami. New results for learning noisy parities and halfspaces. In *Proc. 47th IEEE Symp. on Foundations of Computer Science*, 2006.
- [GKRZ07] P. Gopalan, A. Klivans, A. Rao, and D. Zuckerman. List-Decoding Reed-Muller codes over $\text{GF}[2]$. In *Manuscript*, 2007.
- [GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proc. 21st ACM Symposium on the Theory of Computing*, pages 25–32, 1989.
- [GR06] V. Guruswami and A. Rudra. Explicit capacity-achieving list decodable codes. In *Proc. 38th ACM Symposium on the Theory of Computing*, pages 1–10, 2006.
- [GRS00] O. Goldreich, R. Rubinfeld, and M. Sudan. Learning polynomials with queries: The highly noisy case. *SIAM J. Discrete Math.*, 13(4):535–570, 2000.
- [GS99] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and Algebraic-Geometric codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.
- [Gur04] V. Guruswami. *List Decoding of Error-Correcting Codes*, volume 3282 of *Lecture Notes in Computer Science*. Springer, 2004.
- [GV05] V. Guruswami and A. Vardy. Maximum-likelihood decoding of Reed-Solomon codes is NP-hard. In *Proc. 16th ACM-SIAM Symp. on Discrete algorithms*, 2005.
- [Hås01] J. Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
- [KKMO04] S. Khot, G. Kindler, E. Mossel, and R. O’Donnell. Optimal inapproximability results for Max-Cut and other 2-variable CSPs? In *Proc. 45th IEEE Symp. on Foundations of Computer Science (FOCS'04)*, pages 146–154, 2004.

- [KKMS05] A. Kalai, A. Klivans, Y. Mansour, and R. Servedio. Agnostically learning halfspaces. In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, pages 11–20, 2005.
- [KM91] E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. In *Proc. 23rd ACM Symposium on the Theory of Computing*, pages 455–464, 1991.
- [LMN93] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *J. ACM*, 40(3):607–620, 1993.
- [PV05] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, pages 285–294, 2005.
- [Raz87] A. Razborov. Lower bounds for the size of circuits of bounded depth with basis $\{\wedge, \oplus\}$. *Mathematical Notes of the Academy of Science of the USSR*, 41:333–338, 1987.
- [Raz98] R. Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998.
- [Smo87] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proc. 19th ACM Symposium on Theory of Computing*, pages 77–82, 1987.
- [STV01] M. Sudan, L. Trevisan, and S. P. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.
- [Sud97] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.
- [Tre04] L. Trevisan. Some applications of coding theory in computational complexity. *Quaderni di Matematica*, 13:347–424, 2004.
- [TSZS01] A. Ta-Shma, D. Zuckerman, and S. Safra. Extractors from Reed-Muller codes. In *Proc. 42nd IEEE Symp. on Foundations of Computer Science*, pages 638–647, 2001.

A Extension to Arbitrary Finite Fields

We now sketch the proof of Theorem 2. The reduction follows the same scheme as in the case of polynomials over $\mathbb{F}[2]$. Here we describe the Dictatorship Test and the Consistency Test for polynomials over $\mathbb{F}[q]$.

Dictatorship Testing

Analogous to the $\mathbb{F}[2]$ case, we use $\eta \stackrel{\varepsilon}{\leftarrow} \mathbb{F}[q]^k$ to denote sampling η from the distribution, where each η_i is independently set to 0 with probability $1 - \varepsilon$ and with probability ε is selected uniformly from $\mathbb{F}[q] - \{0\}$. The Basic Dictatorship Test in this case is as follows,

Algorithm 11 BASIC DICTATORSHIP TEST:
Pick $\eta \stackrel{\varepsilon}{\leftarrow} \mathbb{F}[q]^k$ and test if $P(\eta) = 0$.

This test is analyzed using the following form of the Schwartz-Zippel lemma.

Fact 21 Let $P(X_1, \dots, X_k)$ be a non-zero polynomial of degree d over $\mathbb{F}[q]$, such that $d = a(q - 1) + b$, for $0 \leq b \leq q - 1$. Then

$$\Pr_{\eta \leftarrow \mathbb{F}[q]^k} [P(\eta) = 0] \leq s(d, q)$$

where $s(d, q) = 1 - \frac{q-b}{q^{a+1}}$.

For a polynomial P over $\mathbb{F}[q]$, we have a corresponding hypergraph $H(P)$, similar to the $\mathbb{F}[2]$ case. For every monomial

$$m = c \cdot \prod_{i \in T \subseteq [k]} x_i^{e_i} \quad (c \neq 0)$$

of P , $H(P)$ contains a hyperedge consisting of the vertices x_i with multiplicity e_i for $i \in T$. Analogous to the $\mathbb{F}[2]$ case, we obtain the following theorem:

Theorem 22 Let $P(X_1, \dots, X_k)$ be a degree d polynomial over $\mathbb{F}[q]$ that passes the Basic Dictatorship Test with probability $s(d, q) + \delta$ for some $\delta > 0$. Then the largest matching in the hypergraph $H(P)$ is of size $C'(q, \varepsilon, \delta, d)$. Further the constant term c in $P(X_1, \dots, X_k)$ is 0.

Proof: Our proof proceeds in a similar manner to the $\mathbb{F}[2]$ case. For convenience, we define $\varepsilon' = \left(\frac{q-1}{q}\varepsilon\right)$.

We do a two step sampling procedure for η :

1. Set every variable η_i to 0 independently with probability $1 - \varepsilon'$.
2. Independently set each of the remaining variables to a value from $\mathbb{F}[q]$.

Clearly this induces an ε -biased distribution on η . Let X^S denote the set of these variables. The resulting polynomial $P'(X^S)$ consists of the hypergraph induced by the vertex set S . Also

$$\Pr_{\eta \stackrel{\varepsilon}{\leftarrow} \mathbb{F}[q]^k} [P(\eta) \neq 0] = \Pr_{\eta' \leftarrow \mathbb{F}[q]^{|S|}} [P'(\eta') \neq 0]$$

If $P'(X^S)$ is non-zero, then since it has degree at most d , $\Pr[P'(\eta') \neq 0] \geq 1 - s(d, q)$. Now if $c \neq 0$, then P' also has the non zero constant term, hence it is a non-zero polynomial, so $\Pr[P(\eta) \neq 0] = \Pr[P'(\eta') \neq 0] \geq 1 - s(d, q)$, which is a contradiction.

Now assume that the hypergraph $H(P)$ contains a matching M of size $|M| \geq \frac{C}{(\varepsilon')^d}$ where the constant C will be fixed later. For each hyperedge $e \in M$, the probability that all its vertices are chosen to be in S is at least $(\varepsilon')^{|e|}$. Also, since M is a matching, these events are independent for various edges. Thus the probability that none of these edges occurs in the hypergraph induced by S is bounded by

$$\prod_{e \in M} (1 - (\varepsilon')^{|e|}) \leq (1 - (\varepsilon')^d)^{\frac{C}{(\varepsilon')^d}} < e^{-C}.$$

Hence, with probability $1 - e^{-C}$, the subgraph induced by S is non-empty. Conditioned on this event, $P'(X^S)$ is a non-zero polynomial of degree at most d , hence $P'(\eta') \neq 0$ with probability at least q^{-d} . Thus

$$\Pr[P(\eta) \neq 0] \geq (1 - e^{-C}) \cdot (1 - s(d, q)).$$

For sufficiently large C , this contradicts the fact that $\Pr[P(\eta) = 0] \geq s(d, q) + \delta$. ■

Consistency Testing

We consider the following consistency problem as described in Section 5. There are two vertices u and v , each of them is assigned a label $l(u), l(v) \in [k]$ respectively. The vertex u is assigned a projection function $\pi : [k] \rightarrow [t]$, while the vertex v is assigned a projection function $\sigma : [k] \rightarrow [t]$. The goal is to check whether the labels $l(u)$ and $l(v)$ satisfy $\pi(l(u)) = \sigma(l(v))$. We want a test that accepts all polynomials of the form $X_i + Y_j$ where $\pi(i) = \sigma(j)$. Let us denote the set of all such polynomials by \mathcal{D} . The test will specify target values for points of the form $(\mathbf{x}, \mathbf{y}) \in \mathbb{F}[q]^{2k}$ projected onto a certain lower dimensional subspace. First we need the following definition,

Definition 4 Given a projection function $\pi : [k] \rightarrow [t]$, for $\mathbf{z} \in \{0, 1\}^t$, we define the vector $\mathbf{z} \circ \pi \in \mathbb{F}[q]^k$ by $(\mathbf{z} \circ \pi)_i = z_{\pi(i)}$.

We construct a subspace H on which every polynomial in \mathcal{D} vanishes. Consider the subspace H defined by the equations

$$X_i + Y_j = 0, \quad \pi(i) = \sigma(j) \tag{5}$$

As before, we have the following analogous lemma.

Lemma 23 The subspace H contains the vectors $(\mathbf{z} \circ \pi, -\mathbf{z} \circ \sigma)$ for $\mathbf{z} \in \{0, 1\}^t$.

Proof: We need to check that $(\mathbf{x}, \mathbf{y}) = (\mathbf{z} \circ \pi, -\mathbf{z} \circ \sigma)$ satisfies $x_i + y_j = 0$ for all $\pi(i) = \sigma(j)$. But

$$x_i = (\mathbf{z} \circ \pi)_i = z_{\pi(i)}, \quad y_j = -(\mathbf{z} \circ \sigma)_j = -z_{\sigma(j)} \quad \text{hence } x_i = -y_j.$$

Let $\mathbf{g} = (\mathbf{1}^k, \mathbf{0}^k)$. Every polynomial in \mathcal{D} is folded over $\mathbf{g} + H$. We pick a basis $\mathbf{h}(1), \dots, \mathbf{h}(t)$ for H and complete this to a basis F of $\mathbb{F}[q]^{2k}$ given by $F = \{\mathbf{f}(1), \dots, \mathbf{f}(2k - t - 1), \mathbf{g}, \mathbf{h}(1), \dots, \mathbf{h}(t)\}$ for some suitable choice of $\mathbf{f}(i)$ s. Set $g = 2k - t$. Then

$$(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{g-1} \lambda_i \mathbf{f}(i) + \lambda_g \mathbf{g} + \sum_{j=1}^t \mu_j \mathbf{h}(j)$$

Algorithm 12 FOLDED CONSISTENCY TEST:

1. For vertex u , pick $\eta \xleftarrow{\varepsilon} \mathbb{F}[q]^k$.
2. Write $(\eta, \mathbf{0}^k) = (\lambda_1, \dots, \lambda_g, \mu_1, \dots, \mu_t)$ and test if $P(\lambda_1, \dots, \lambda_{g-1}) = \lambda_g$.
3. For vertex v , pick $\eta' \xleftarrow{\varepsilon} \mathbb{F}[q]^k$.
4. Write $(\mathbf{0}^k, \eta') = (\lambda_1, \dots, \lambda_g, \mu_1, \dots, \mu_t)$ and test if $P(\lambda_1, \dots, \lambda_{g-1}) = \lambda_g$.

Algorithm 13 DECODING PROCEDURE FOR THE FOLDED CONSISTENCY TEST:

1. Let $Q(X_1, \dots, X_k, Y_1, \dots, Y_t) = P(\lambda_1, \dots, \lambda_{g-1}) + \lambda_g$.
2. Run Algorithm 2 on $Q(X_1, \dots, X_k, \mathbf{0}^k)$ to get list $L(u)$.
3. Run Algorithm 2 on $Q(\mathbf{0}^k, Y_1, \dots, Y_t)$ to get list $L(v)$.
4. Assign $l(u) \leftarrow L(u)$ and $l(v) \leftarrow L(v)$.

We define the polynomials $U(X_1, \dots, X_k) = Q(X_1, \dots, X_k, \mathbf{0}^k)$ and $V(Y_1, \dots, Y_t) = Q(\mathbf{0}^k, Y_1, \dots, Y_t)$. As before, we have the following analogous lemma which we state here without proof.

Lemma 24 Define the projected polynomials

$$U_\pi(Z_1, \dots, Z_t) = U(Z_{\pi(1)}, \dots, Z_{\pi(k)}), \quad V_\sigma(Z_1, \dots, Z_t) = V(Z_{\sigma(1)}, \dots, Z_{\sigma(t)}).$$

Then $U_\pi(Z_1, \dots, Z_t) = V_\sigma(Z_1, \dots, Z_t)$.

The following analysis of the Folded Consistency Test proceeds in a similar manner as before.

Theorem 25 Define the projections of the lists $L(u)$ and $L(v)$ as $L_\pi(u) = \{\pi(i) \mid i \in L(u)\}$ and $L_\sigma(v) = \{\sigma(j) \mid j \in L(v)\}$.

1. Both $L_\pi(u)$ and $L_\sigma(v)$ are vertex covers for the hypergraph $H(U_\pi) = H(V_\sigma)$.
2. The polynomials U_π and V_σ are each folded over $\mathbf{1}^t$.
3. The probability that $P(\lambda_1, \dots, \lambda_{g-1})$ passes the folded consistency test for vertex u equals the probability that $U(X_1, \dots, X_k)$ passes the Basic Dictatorship Test.

Proof: Consider an assignment \mathbf{z} to the variables Z_1, \dots, Z_t such that $\mathbf{z}_i = 0$ for all $i \in L_\pi(u)$. Now, $U_\pi(\mathbf{z}) = U(\mathbf{z} \circ \pi)$. We have $(\mathbf{z} \circ \pi)_j = \mathbf{z}_{\pi(j)} = 0$ for all $j \in L(u)$ since $\mathbf{z}_i = 0$ for all $i \in L_\pi(u)$, and $\pi(j) \in L_\pi(u)$ for all $j \in L(u)$. Since $L(u)$ is a vertex cover for $H(U)$, therefore, by setting all the variables in $L(u)$ to zero, we obtain $U(\mathbf{z} \circ \pi) = 0$. Hence, $U_\pi(\mathbf{z}) = 0$. Since \mathbf{z} was an arbitrary assignment which set variables in $L_\pi(u)$ to zero, $U_\pi(Z_1, \dots, Z_t)$ vanishes over all such assignments. Therefore, a variable of $L_\pi(u)$ is present in every monomial of $U_\pi(Z_1, \dots, Z_t)$, and so $L_\pi(u)$ is a vertex cover for $H(U_\pi)$. Similarly, $L(V_\sigma)$ is a vertex cover for $H(V_\sigma)$. By Lemma 24, since $U_\pi = V_\sigma$, $H(U_\pi) = H(V_\sigma)$.

Using arguments similar to Theorem 13, we can show that U and V are folded over $\mathbf{1}^k$, and since

$$U_\pi(\mathbf{z} + \lambda \mathbf{1}^t) = U((\mathbf{z} + \lambda \mathbf{1}^t) \circ \pi) = U(\mathbf{z} \circ \pi + \lambda \mathbf{1}^k) = \lambda + U(\mathbf{z} \circ \pi) = \lambda + U_\pi(\mathbf{z}).$$

So U_π is folded over $\mathbf{1}^t$ and similarly for V_σ . This shows that the hypergraph $H(U_\pi) = H(V_\sigma)$ is non-empty. The proof of Part 3 follows that of Theorem 12. We omit the details here. ■

The reduction from LABELCOVER($d + 1$) proceeds along similar lines as Theorem 20. We omit the details.

B Reduction from MAX-LIN

In this section, we consider the polynomial reconstruction problem $\text{POLYREC}(d, n)$ over any finite field $\mathbb{F}[q]$, where $d < q$. We prove a hardness result for the reconstruction problem via a simple reduction from Håstad's result for linear equations.

The $\text{MAX-LIN}(n, q)$ problem consists of point-value pairs $\{\mathbf{x}^i, f(\mathbf{x}^i)\}_{i=1}^m$ with $\mathbf{x}^i \in \mathbb{F}[q]^n$ and $f(\mathbf{x}^i) \in \mathbb{F}[q]$. Our goal is to find a linear polynomial, which satisfies the maximum number of points. The following theorem is due to Håstad [Hås01],

Theorem 26 *For any $\varepsilon, \delta > 0$, given an instance of $\text{MAX-LIN}(n, q)$, it is NP-hard to distinguish between the following cases:*

1. YES INSTANCE: *There is a linear polynomial that satisfies $1 - \varepsilon$ fraction of the points.*
2. NO INSTANCE: *Every linear polynomial satisfies at most $\frac{1}{q} + \delta$ fraction of the points.*

We use this to prove a hardness of $1 - \varepsilon$ versus $\frac{qd+q-d}{q^2} + \delta$ for $\text{POLYREC}(d, n)$. Note that since $d < q$, the soundness lies between d/q and $(d+1)/q$.

Theorem 27 *For any $\varepsilon, \delta > 0$, given an instance of $\text{POLYREC}(d, n)$ over $\mathbb{F}[q]$ with $d < q$, it is NP-hard to distinguish between the following cases:*

1. YES INSTANCE: *There is a linear polynomial satisfying $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for $1 - \varepsilon$ fraction of the points.*
2. NO INSTANCE: *Every polynomial of degree $d < q$ satisfies $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for at most $\frac{qd+q-d}{q^2} + \delta$ fraction of the points.*

Proof: Let the instance of $\text{MAX-LIN}(n, q)$ be given by $\langle \mathbf{x}^i, f(\mathbf{x}^i) \rangle$ for $i = 1, \dots, m$. Our instance of $\text{POLYREC}(d, n)$ is given by the point-value pairs $\langle \lambda \mathbf{x}^i, \lambda f(\mathbf{x}^i) \rangle$ for every $\lambda \in \mathbb{F}[q]$ and every $i \in [m]$.

If the MAX-LIN instance is a YES instance, then there is a linear polynomial $P(X_1, \dots, X_n)$ that satisfies $P(\mathbf{x}^i) = f(\mathbf{x}^i)$ for $1 - \varepsilon$ fraction. It is easy to see that the same polynomial satisfies at least $1 - \varepsilon$ fraction of the constraints for the $\text{POLYREC}(d, n)$ problem.

Suppose that MAX-LIN instance is a NO instance. Assume that there is a degree d polynomial $P(X_1, \dots, X_n)$ that satisfies $\frac{qd+q-d}{q^2} + \delta$ fraction of the points. We call a point \mathbf{x} *good* if P satisfies at least $d+1$ of the pairs $\langle \lambda \mathbf{x}, \lambda f(\mathbf{x}) \rangle$ over all values of $\lambda \in \mathbb{F}[q]$. By an averaging argument, at least $\frac{1}{q} + \gamma$ fraction of the points are good, for some constant γ . Since P satisfies at least $d+1$ of the pairs $\langle \lambda \mathbf{x}, \lambda f(\mathbf{x}) \rangle$, the univariate polynomial in λ given by $P(\lambda \mathbf{x}) - \lambda f(\mathbf{x})$ has $d+1$ roots in $\mathbb{F}[q]$. However, it has degree at most d , so it must be the zero polynomial. Equating the coefficients of λ on both sides, we get $P'(\mathbf{x}) = f(\mathbf{x})$ for every good point, where P' is the linear part of P . Thus P' is a linear polynomial satisfying $\frac{1}{q} + \gamma$ fraction of the constraints of the original MAX-LIN instance. Taking the soundness of the MAX-LIN instance sufficiently small, we get a contradiction. \blacksquare

C Hardness for LABELCOVER($d+1$)

In this section we give a reduction from the standard LABELCOVER instance to the LABELCOVER($d+1$) instance as defined in section 6, where d is a fixed constant.

Definition 5 *A LABELCOVER instance $\mathcal{L}(G(U, V, E), [t], [k], \{\pi^{v,u}\}_{\{u,v\} \in E})$ is a bipartite graph G with bipartition U, V and edge set E with a projection $\pi^{v,u} : [k] \mapsto [t]$, for every $\{u, v\} \in E$, with $u \in U$ and $v \in V$. Moreover, every vertex in U has the same degree and every vertex in V has the same degree.*

A vertex labelling $l(w)$ for all $w \in U \cup V$, satisfies an edge $\{u, v\} \in E$, (where $u \in U$ and $v \in V$) iff $\pi^{v,u}(l(v)) = l(u)$.

The following theorem is a consequence of the PCP Theorem [ALM⁺98, AS98] and Raz's Parallel Repetition Theorem [Raz98].

Theorem 28 *For every constant $\beta > 0$, given an instance \mathcal{L} of LABELCOVER, it is NP-hard to distinguish between the following cases:*

1. YES INSTANCE : *There is some vertex labelling that satisfies all the edges of \mathcal{L} .*
2. NO INSTANCE : *There is no vertex labelling that satisfies β fraction of edges of \mathcal{L} .*

We now give a reduction from LABELCOVER to the LABELCOVER($d + 1$) problem which will prove Theorem 17.

Proof: Given an instance of LABELCOVER, $\mathcal{L}(G(U, V, E), [t], [k], \{\pi^{v,u}\}_{\{u,v\} \in E})$, we construct an instance \mathcal{L}' of LABELCOVER($d + 1$) in the following manner:

1. The vertex set of \mathcal{L}' is $V' = V$.
2. A hyperedge e' is added in the following manner. Pick a random $u \in U$ and pick vertices v_1, v_2, \dots, v_{d+1} , uniformly at random from the neighbors of u in G . Set $e' = \{v_i\}_{i=1}^{d+1}$, and the associated $d + 1$ -tuple of projections to be $\{\pi_i\}_{i=1}^{d+1}$, where $\pi_i = \pi^{v_i, u}$ for all $1 \leq i \leq d + 1$.
3. Add all such hyperedges possible to the edge set E' .

Consider a subset $S \subseteq V' = V$ of size $\delta|V'|$. Let u be any vertex in U of the instance \mathcal{L} . Let p_u be the fraction of neighbors of u in S . Since, every vertex of U has the same degree and every vertex of V has the same degree, $\mathbb{E}_{u \in_R U}[p_u] = \delta$. The way edge set E' of \mathcal{L}' is constructed implies that the fraction of hyperedges in E' induced by S is the probability that all $d + 1$ vertices uniformly chosen at random from neighbors of a vertex u (which is chosen uniformly at random from U), lie in S . For a given $u \in U$, the probability that $d + 1$ vertices chosen uniformly at random from its neighbors lie in S is p_u^{d+1} . Therefore the fraction of edges of E' induced by S is $\mathbb{E}_{u \in_R U}[p_u^{d+1}] \geq (\mathbb{E}_{u \in_R U}[p_u])^{d+1} = \delta^{d+1}$. Hence, a constant fraction of hyperedges in E' are induced by a subset S of constant fraction of vertices in V' .

Note that by applying Parallel Repetition on LABELCOVER we can increase the degrees of vertices in U arbitrarily while reducing the soundness. Since $d + 1$ is a fixed constant, we can arbitrarily reduce the fraction of hyperedges of LABELCOVER($d + 1$) which have repeated vertices and hence remove these hyperedges from the instance.

If \mathcal{L} is a YES instance, then there is a labelling l that satisfies all the edges of \mathcal{L} . Clearly, the labelling l restricted to V will strongly satisfy all the hyperedges of \mathcal{L}' .

If \mathcal{L} is a NO instance, then there is no labelling that satisfies β fraction of the edges of \mathcal{L} . Now, suppose that there is a labelling l that weakly satisfies α fraction of hyperedges of \mathcal{L}' . For every vertex $u \in U$, define q_u to be the probability that two (distinct) random neighbors of u are labelled consistently by l . Since every vertex in U has equal degree and every vertex of V has equal degree, and by union bound, we obtain, $\mathbb{E}_u[q_u] \geq \alpha / \binom{d+1}{2}$. Let $2\alpha' = \alpha / \binom{d+1}{2}$. Call a vertex $u \in U$ 'good' if $q_u \geq \alpha'$. By averaging, at least α' fraction of vertices U are good. Let $u \in U$ be a 'good' vertex, i.e. l labels at least α' fraction of pairs $\{v_i, v_j\}$ consistently where v_i and v_j are neighbors of u . Again, by averaging, there must be a neighbor v' of u which is consistently labelled with at least $\alpha'/2$ fraction of neighbors of u . Now, extending the labelling

l to u , by setting $l(u) = \pi^{v',u}(l(v'))$ will satisfy at least $\alpha'/2$ fraction of edges incident on u in \mathcal{L} . By labelling every 'good' vertex in a similar manner, we obtain a labelling l that satisfies at least $\alpha'^2/2$ fraction of edges of \mathcal{L} . Since $d + 1$ is a fixed constant, for any $\alpha > 0$, choosing β to be small enough, we get a contradiction. So, there is no labelling of \mathcal{L}' that weakly satisfies α fraction of the hyperedges. ■