18.408 Topics in Theoretical Computer Science Fall 2022
Lectures 10,11

Dor Minzer

In previous lectures we have proved a PCP theorem with poly-logarithmic number of queries; namely that gap-CSG$[1, 1/\log(n)^c]$ is NP-hard on instances with alphabet size and number of queries that are both $\mathsf{poly}(\log n)$, where n is the size of the instance. As hinted earlier, there is a way to "recurse" and by that to take the number of queries further down to be doubly logarithmic and even triply logarithmic. How does one eventually achieve a PCP with constant number of queries, though? Today, we will assume a stronger (but similar) version of the result we have already proved, and use it to construct a PCP with constantly many queries.

1 Overview

Our starting point today will be the following PCP construction.

Theorem 1.1 (PCP with poly-loglog number of queries)**.** *There are absolute constants $\varepsilon, C > 0$ such that gap-QS$[1, 1-\varepsilon]$ is NP-hard on instances with alphabet size $O(1)$ and number of queries at most $(\log\log(n))^C$.*

This theorem differs from the result we proved over the last few lectures in several aspects. First, the alphabet size here is $O(1)$ instead of poly-logarithmic; this is not a significant difference, since we can represent a single alphabet symbol by $\mathsf{poly}(\log\log n)$ many bits, thus decrease the alphabet size at the expense of increasing the number of queries by that factor. Secondly, in the last lecture we only achieved poly-logarithmic number of queries, and here we are assuming $\mathsf{poly}(\log\log n)$ number of queries; this difference is more significant, and we will discuss it at a later point. Third, the soundness of our PCP is $1-\varepsilon$ (i.e. close to 1) as opposed to close to 0. In this aspect, the result we are using here is weaker than the one we have proved. We are doing it so as to simplify the presentation, as working in regime in which the soundness bounded away from 1 is easier than working in the regime the soundness is close to 0. It is possible though to modify the ideas presented herein and establish a result with soundness close to 0.

We will use Theorem 1.1 to prove:

Theorem 1.2 (PCP with a constant number of queries)**.** *There is an absolute constant $\varepsilon > 0$ such that gap-CSG$[1, 1-\varepsilon]$ is NP-hard on instances with alphabet size $O(1)$ and number of queries $O(1)$.*

Towards this end, we will first introduce a construction that doesn't work but will be helpful for us in conveying some of the ideas that go into the proof. We will then identify an additional structural property that we could hope to get from the instances from Theorem 1.1, and then modify our construction so that it is indeed correct assuming this additional property. We will discuss this additional property and a transformation that allows us to guarantee it in subsequent lectures.

2 Hadamard and Quadratic Hadamard Codes

Suppose we have an instance (X, E) of quadratic solvability as in Theorem 1.1; thus, we have variables $x_1, \ldots, x_n$ that are to be assigned values from $\mathbb{F}_q$, and equations $e_1, \ldots, e_m$ each containing at most $s = \mathsf{poly}(\log \log n)$ many variables. We know that in the YES case, there is an assignment $A \colon X \to \mathbb{F}_q$ satisfying all equations, and in the NO case each assignment A satisfies at most $\delta = 1/\log(n)^c$ of the equations. We want to encode the assignment A in a different way, so that by making $O(1)$ queries into the encoding we can check whether the encoded A satisfies e_i, and for that we are going to use the *quadratic Hadamard code* that you have already seen in the problem set.

2.1 The Hadamard Code

Recall that the Hadamard code over $\mathbb{F}_q$ is defined as follows:

Definition 2.1. *For $v \in \mathbb{F}_q^s$, we define the Hadamard Encoding of v as the truth table of the function $h_v \colon \mathbb{F}_q^s \to \mathbb{F}_q$ defined by*

$$h_v(y) = \langle v, y \rangle .$$

In words, for each vector $v \in \mathbb{F}_q^s$ we consider the linear function $h_v(y) = \langle v, y \rangle$, and the truth table of h_v is the Hadamard encoding of the vector v. It is easy to prove that the Hadamard code is a linear error correcting code and has relative distance $1 - 1/q$. Note that if we wish to encode a vector of length s, then the length of the encoding of v is q^s, i.e. exponential in s. Thus, we can afford to use such encodings only if q^s is at most polynomially large in n, which is the reason we would like to apply the Hadamard code only to encode strings of length at most $s = O(\log n)$; in our case, the strings will be of length $s = \mathsf{poly}(\log \log n)$.

The Hadamard code is locally testable. Indeed, note that if h_v is a legitimate Hadamard codeword, then $h_v(x + y) = h_v(x) + h_v(y)$, which suggests the following randomized local test for the Hadamard. Given a table of values $f \colon \mathbb{F}_q^s \to \mathbb{F}_q$ which is supposed to be an Hadamard encoding of some vector v, perform the following test, which we call the linearity test:

1. Pick $x, y \in \mathbb{F}_q^s$ uniformly.
2. Query $f(x)$, $f(y)$ and $f(x + y)$.
3. Accept if $f(x + y) = f(x) + f(y)$.

The following lemma asserts that correctness of the test:

Lemma 2.2. *Suppose that $f \colon \mathbb{F}_q^s \to \mathbb{F}_q$ passes the linearity test with probability at least $1 - \varepsilon$ for $\varepsilon < 1/8$. Then, there exists $v \in \mathbb{F}_q^s$, such that*

$$\Pr_{x \in \mathbb{F}_q^s} [f(x) = h_v(x)] \geqslant 1 - 2\varepsilon.$$

Proof. The proof is identical to the proof shown earlier in the course for $q = 2$. One defines

$$g(x) = \mathsf{majority}_{y \in \mathbb{F}_q^s} f(x + y) - f(y)$$

and shows that $\Pr_{x \in \mathbb{F}_q^s} [f(x) = g(x)] \geqslant 1 - 2\varepsilon$, and that provided that $\varepsilon < 1/8$, g passes the linearity test with probability 1, in which case g can easily be seen to be a Hadamard codeword. □

The last important feature of the Hadamard code is that it enables us to verify linear constraints on the encoded vector v. Indeed, suppose we had some vector of coefficients $\alpha_1, \ldots, \alpha_s$ and we want to test that $\sum_{i=1}^{s} \alpha_i v_i = c$. How can we do it using the Hadamard encoding of v?

Well, if we have the legitimate Hadamard encoding of v, namely h_v, the above constraint can simply be re-written as $h_v(\vec{\alpha}) = c$ where $\vec{\alpha} = (\alpha_1, \ldots, \alpha_s)$. In our situation though, we will have oracle access to a table $f : \mathbb{F}_q^s \to \mathbb{F}_q$ that we can only guarantee to be close to a table of a legitimate Hadamard codeword h_v. Thus, it could be the case that the input $\vec{\alpha}$ just happens to be one of the inputs in which f and h_v differ. Can we verify the linear constraint on v despite of that?

Instead of directly asking for the value of f at $\vec{\alpha}$, we can use self correction! Namely, we can sample $x \in \mathbb{F}_q^s$ uniformly, and then observe that each one of the inputs x and $x + \vec{\alpha}$ is distributed uniformly on $\mathbb{F}_q^s$, hence we expect f and h_v to agree on them. Then, we can read off $f(x)$ and $f(x + \vec{\alpha})$, and check that $f(x+\vec{\alpha}) - f(x) = c$. The idea is that with high probability over x, $f(x+\vec{\alpha}) - f(x) = h_v(x+\vec{\alpha}) - h_v(x) = h_v(\vec{\alpha})$, hence this test will pass only if v satisfies the linear constraint.

Summarizing, we state the following tester for the Hadamard codeword which enables us to check whether a given table f is close to a Hadamard codeword, and if the nearby codeword satisfies a given linear constraint. The input to the test is an oracle access to $f : \mathbb{F}_q^s \to \mathbb{F}_q$, as well as a vector of coefficients $\vec{\alpha} \in \mathbb{F}_q^s$ and $c \in \mathbb{F}_q$, and we want to test if f is close to some Hadamard codeword h_v that satisfies that $\langle \vec{\alpha}, v \rangle = c$.

1. Sample $x, y \in \mathbb{F}_q^s$ uniformly and check that $f(x + y) = f(x) + f(y)$.
2. Query $f(x + \vec{\alpha})$ and check that $f(x + \vec{\alpha}) - f(x) = c$.

We summarize the above discussion with the following lemma.

Lemma 2.3. *Suppose $f : \mathbb{F}_q^s \to \mathbb{F}_q$, $\vec{\alpha}$ and c are such that the above test accepts with probability at least $1 - \varepsilon$, for $\varepsilon < 1/8$. Then there exists $v \in \mathbb{F}_q^s$ such that*

1. *v satisfies the linear constraint: $\langle \vec{\alpha}, v \rangle = c$.*
2. *f is close to h_v: $\Pr_{x \in \mathbb{F}_q^s} [f(x) = h_v(x)] \geqslant 1 - 2\varepsilon$.*

Proof. Left to the reader. □

Thus, the Hadamard code only incurs an exponential blow-up in the size of the encoding, and it allows us to check linear constraints in the encoded values. This almost fits what we need in order to go from Theorem 1.1 to Theorem 1.2, except that there we need to be able to check quadratic constraints in the encoded values. In the next section we show a variant of the Hadamard code which enables us to do that.

Remark 2.4. *A few remarks are in order.*

1. *Notice that while the linearity tester required only 3 queries, in order to check a linear constraint we required an additional query. There are ways to incorporate the linear constraint check into the linearity tester so that to keep the number of queries to be 3; this is not very important for us now as this difference will be minor, but there are some applications in which such ideas are crucial.*
2. *As discussed, there are analogs the above ideas and in particular of Lemma 2.2, in the low-soundness regime. Roughly speaking, such results are concerned with the case the field size q is though of as somewhat large, and consider the case that f passes the linearity test with probability at least $1/q + \varepsilon$. In such cases, one proves a list decoding statement, saying that there is a list $v_1, \ldots, v_k$*

where $k = k(\varepsilon) \in \mathbb{N}$ that "explains" all of the success probability of the test. Namely, the probability that $f(x+y) = f(x) + f(y)$ but f disagrees with each one of h_{v_i} on at least one of $\{x, y, x+y\}$ is very small. Incorporating the linearity checks requires more effort.

2.2 The Quadratic Hadamard Code

The quadratic Hadamard code is defined as:

Definition 2.5. *For $v \in \mathbb{F}_q^s$, we define the quadratic Hadamard encoding of v as the truth table of the function $Qh_v \colon \mathbb{F}_q^{s^2} \to \mathbb{F}_q$ defined by*

$$Qh_v(y) = h_{v \otimes v}(y) = \langle v \otimes v, y \rangle ,$$

where $v \otimes v \in \mathbb{F}_q^{s^2}$ is the vector whose i, j entry is $v_i v_j$.

One way to think about the quadratic Hadamard code is that it is a subset of the Hadamard code of vectors of length s^2, corresponding only to vectors of the form $v \otimes v$. We will want to show that we can still locally test the quadratic Hadamard code, and that it enables us to verify quadratic constraints in v. We also remark that as the size of the encoding of a string of length s is q^{s^2}, we will want to use the Hadamard encoding only on strings whose length is at most $O(\sqrt{\log n})$, and for us it will be the case that $s = \mathsf{poly}(\log \log n)$.

Suppose we have oracle access to a functions $f \colon \mathbb{F}_q^s \to \mathbb{F}_q$ and $Qf \colon \mathbb{F}_q^{s^2} \to \mathbb{F}_q$ which are supposed to be the Hadamard and the quadratic Hadamard encoding of some $v \in \mathbb{F}_q^s$. Suppose, in addition, we have a quadratic constraint $\sum_{i,j} \alpha_{i,j} v_i v_j = c$ on the v's. How can we test that f and Qf are indeed such functions, and that v satisfies the quadratic constraint?

By the previous discussion, we can already perform a test to guarantee that f and Qf are close to Hadamard codewords of some vectors. Indeed, our tester begins by:

1. Run the linearity tester on f: namely, sample $x, y \in \mathbb{F}_q^s$ check that $f(x+y) = f(x) + f(y)$, else reject.
2. Run the linearity tester on f: namely, sample $x', y' \in \mathbb{F}_q^{s^2}$ check that $Qf(x'+y') = Qf(x') + Qf(y')$, else reject.

By Lemma 2.2, if f and Qf pass this test with probability at least $1 - \varepsilon$ where $\varepsilon < 1/8$, then there are $v \in \mathbb{F}_q^s$ and $u \in \mathbb{F}_q^{s^2}$ such that f is 2ε-close to h_v and qf is 2ε-close to h_u. Next, we would like to test that $u = v \otimes v$, and for that we begin with the basic observation that

$$h_{v \otimes v}(x \otimes y) = \langle v \otimes v, x \otimes y \rangle = \langle v, x \rangle \langle v, y \rangle = h_v(x) h_v(y),$$

hence we get a potential connection that we should test. Namely, this suggests test to test that $u = v \otimes v$, we would like to check that $h_u(x \otimes y) = h_v(x) h_v(y)$, and we will indeed do that. There is one catch: the vector $x \otimes y$ is not uniformly distributed over $\mathbb{F}_q^{s^2}$, so if we attempt to access the value of h_u on it by directly querying Qf on that point, it may be the case that this is a location in which h_u and Qf differ. To overcome this, we use local correction again, and get the tensor tester:

1. Sample $x, y \in \mathbb{F}_q^s$ and $z \in \mathbb{F}_q^{s^2}$ uniformly.
2. Check that $Qf(z + x \otimes y) - Qf(z) = f(x) f(y)$, else reject.

We have the following lemma.

Lemma 2.6. *Suppose that $f\colon \mathbb{F}_q^s \to \mathbb{F}q$ and $Qf\colon \mathbb{F}_q^{s^2} \to \mathbb{F}_q$ are functions that succeed with probability at least $1-\varepsilon$ in the linearity tester and the tensor tester, and $\varepsilon \leqslant \frac{1}{100}$. Then there is $v \in \mathbb{F}_q^s$ such that f is 2ε close to h_v, and Qf is 2ε close to $h_{v\otimes v}$.*

Proof. From Lemma 2.2 there are $v \in \mathbb{F}_q^s$ and $u \in \mathbb{F}_q^{s^2}$ such that f is 2ε-close to h_v and Qf is 2ε-close to h_u. Note that with probability at least $1-8\varepsilon$ it holds that $Qf(z+x\otimes y) = h_u(z+x\otimes y)$, $Qf(z) = h_u(z)$, $f(x) = h_v(x)$, $f(y) = h_v(y)$, so

$$\Pr_{x,y\in\mathbb{F}_q^s}[h_v(x)h_v(y) = h_u(x\otimes y)] \geqslant 1-9\varepsilon.$$

However, note that if $u \neq v\otimes v$, then the functions $P(x,y) = h_v(x)h_v(y)$ and $Q(x,y) = h_u(x\otimes y)$ are distinct functions over $\mathbb{F}_q^{2s}$ of individual degree 1 and total degree 2, hence by Schwarz-Zippel they disagree on randomly chosen x, y with probability at least $(1-1/q)^2 > 9\varepsilon$. It follows that we must have that $u = v\otimes v$. □

Recall that in the last section, we saw that if we have oracle access to a table g that is close to a Hadamard function h_u, then we can check linear equations in u. In our situation, we have access to a table Qf that is close to a Hadamard function $h_{v\otimes v}$, hence we can check linear equations in $v\otimes v$, which are simply quadratic equations in v. Thus, we reach our final tester. The tester gets oracle access to tables $f\colon \mathbb{F}_q^s \to \mathbb{F}_q$ and $Qf\colon \mathbb{F}_q^{s^2}$ as well as a vector of coefficients $\vec{\alpha} \in \mathbb{F}_q^{s^2}$ and $c \in \mathbb{F}_q$; the tester needs to verify that for some $v \in \mathbb{F}_q^s$, f is close to h_v, Qf is close to $h_{v\otimes v}$ and that $\langle \vec{\alpha}, v\otimes v\rangle = c$.

1. Run the linearity tester on f and Qf:
   (a) Sample $x, y \in \mathbb{F}_q^s$ and $x', y' \in \mathbb{F}_q^{s^2}$ uniformly.
   (b) Check that $f(x+y) = f(x)+f(y)$ and $Qf(x'+y') = Qf(x')+Qf(y')$, else reject.
2. Run the tensor tester:
   (a) Sample $x, y \in \mathbb{F}_q^s$ and $z \in \mathbb{F}_q^{s^2}$ uniformly.
   (b) Check that $Qf(z+x\otimes y) - Qf(z) = f(x)f(y)$, else reject.
3. Run the self-correction constraint tester:
   (a) Sample $x \in \mathbb{F}_q^{s^2}$ uniformly.
   (b) Check that $Qf(x+\vec{\alpha}) - Qf(x) = c$, else reject.

The following lemma summarizes the properties of the above tester.

Lemma 2.7. *Suppose that $f\colon \mathbb{F}_q^s \to \mathbb{F}q$ and $Qf\colon \mathbb{F}_q^{s^2} \to \mathbb{F}_q$ are functions that succeed with probability at least $1-\varepsilon$ in the linearity + tensor + constraint tester, and $\varepsilon \leqslant \frac{1}{100}$. Then there is $v \in \mathbb{F}_q^s$ such that f is 2ε close to h_v, and Qf is 2ε close to $h_{v\otimes v}$, and $\langle \vec{\alpha}, v\otimes v\rangle = c$.*

Proof. From Lemma 2.6 it follows that f and qf are 2ε-close to h_v and $h_{v\otimes v}$ for some $v \in \mathbb{F}_q^s$, hence with probability at least $1-5\varepsilon > 0$ over x,

$$c = Qf(x+\vec{\alpha}) - Qf(x) = h_{v\otimes v}(x+\vec{\alpha}) - h_{v\otimes v}(x) = \langle \vec{\alpha}, v\otimes v\rangle,$$

so v satisfies the quadratic constraint. □

3 Going from Theorem 1.1 to Theorem 1.2 via the Quadratic Hadamard Code?

3.1 Composing with the Hadamard Code

The ideas presented in the lecture so far suggest the following way of deducing Theorem 1.2 from Theorem 1.1. Start with an instance of quadratic equations (X, E) as in Theorem 1.1, let $q = O(1)$ be the field size, and write $X = \{x_1, \ldots, x_n\}$, $E = \{e_1, \ldots, e_m\}$. For each equation e_i, consider the set of variables $X_i \subseteq X$ that appear in it, and use the Hadamard and quadratic Hadamard encodings to encode the (supposed) assignment $A \colon X \to \mathbb{F}_q$ on these variables. For convenience, instead of thinking of the supposed A as an assignment, we think of it as a vector $\vec{v} \in \mathbb{F}_q^X$.

Namely, our witness will be a collection of assignments $f_i \colon \mathbb{F}_q^{X_i} \to \mathbb{F}_q$, $Qf_i \colon \mathbb{F}_q^{X_i^2} \to \mathbb{F}_q$, and our intention is that f_i, Qf_i are the Hadamard and quadratic Hadamard encodings of $\vec{v}|_{X_i}$. Thus, we think of the nodes of our CSG as the locations of the tables of these functions, and we use the tester above to define a set of constraints in order to verify it:

1. Sample $i \in \{1, \ldots, m\}$.
2. Run the linearity tester on f_i, Qf_i.
3. Run the tensor tester on f_i, Qf_i
4. Run the self-correction constraint tester on f_i, Qf_i to check the equation e_i. Namely, write e_i as $\langle \alpha_i, v|_{X_i} \rangle = c_i$, and run the self-correction constraint tester on f_i, Qf_i with α_i and c_i.

We note that the run-time of the reduction is $O(m \cdot q^{\max_i |X_i|^2})$, which is polynomial in n (this is the reason that we needed to go down to polyloglog many queries). We also note that overall, each constraint only looks at $O(1)$ locations.

Using Lemma 2.7, one can analyze this PCP construction and show (roughly speaking) that for sufficiently small $\delta > 0$, if $\{f_i, Qf_i\}_{i=1,\ldots,m}$ pass this test with probability at least $1 - \delta$, then one can find a collection of vectors $v_i \in \mathbb{F}_q^{X_i}$ such that f_i, Qf_i are 2δ-close to the Hadamard and quadratic Hadamard encodings of v_i, and that v_i satisfies the equation e_i. This seems good, except that there is one issue: how do we make sure that the vectors v_i are consistent? Namely, suppose we had some variable x which is both in X_i as well as in X_j; how do we make sure that it receives the same value in both v_i and v_j?

To address that, one may try to use the Hadamard and quadratic Hadamard encodings to encode the entire vector $\vec{v}$; this works but then the runtime of the reduction is $2^{\theta(n^2)}$ (and for this, one doesn't need much of what we've done in the course thus far). To remedy the situation we need our initial quadratic solvability instance to have the *block property*.

3.2 The Block Property

We need more structure from the PCP given to us in Theorem 1.1, and the specific structure we look for is called the block property. This additional feature can be guaranteed by a technique we have yet to see called "aggregation of queries", and we defer the discussion on how to achieve it to a later point.

Roughly speaking, the block property addresses the structure of the queries the PCP makes (in this case, the structure of the variables x_i's that an equation depends on), and says that while the total number of these variables can be quite large (poly-logarithmic or doubly logarithmic), they can be read by looking into only much fewer number of "blocks" in the witness (typically constantly many).

Below, we specialize the discussion to quadratic equations, but the definition easily extends to general constraint satisfaction graph problems. We say an instance (X, E) of quadratic equations has the (k, s)-block property if the set of variables X can be partitioned into disjoint sets $X_1 \cup \ldots \cup X_{n'}$ such that $|X_i| \leqslant s$ for each i, each one of the equations e_j contains variables from most k of the blocks X_i and the variables of each monomial in e_j appear in the same block. Furthermore, this partition is given as part of the input. We remark that in this definition, k should be thought of as a constant, say 10, and s should be thought of as small but super constant, say $\mathsf{poly}(\log \log n)$.

With this definition in mind, we now state the variant of Theorem 1.1 that we will use:

Theorem 3.1 (PCP with poly-loglog number of queries)**.** *There are absolute constants $\varepsilon, C > 0$ and $k \in \mathbb{N}$ such that gap-QS$[1, 1 - \varepsilon]$ is NP-hard on instances with alphabet size $O(1)$ satisfying the (k, s)-block property for $s = \mathsf{poly}(\log \log n)$.*

We now prove Theorem 1.2 using Theorem 3.1, and the idea similar to the one from the previous section. Let (X, E) be an instance of quadratic solvability as in Theorem 3.1, and let $X_1 \cup \ldots \cup X_{n'}$ be a partition of X into blocks as in the (k, s)-block property. For each $i = 1, \ldots, n'$, we have a pair of functions $f_i \colon \mathbb{F}_q^{X_i} \to \mathbb{F}_q$ and $Qf_i \colon \mathbb{F}_q^{X_i^2} \to \mathbb{F}_q$. The intention is that if $v \in \mathbb{F}_q^X$ is a vector representing a solution to (X, E), then f_i, Qf_i will be the Hadamard and the quadratic Hadamard encodings of $v|_{X_i}$.

The locations of the tables f_i, Qf_i together constitute the vertices of the CSG we construct Ψ, and we next describe the constraints of Ψ.

We sample an equation $j \in \{1, \ldots, m\}$ and take the blocks $i_1, \ldots, i_k$ that equation e_j depends on. We perform the linearity and tensor testers on f_i, Qf_i for each $i \in \{i_1, \ldots, i_k\}$; then, thinking of the equation e_j as $\langle \alpha_j, v \otimes v \rangle = c_j$, we set $\alpha_{j,i} = \alpha_j|_{X_i^2}$, i.e. the coefficients of α that are associated with monomials in X_i, and use local correction to "read off" $Qf_i(\alpha_{j,i})$, and check that these values add up to c_j. More precisely:

1. Sample $j \in \{1, \ldots, m\}$, and let $i_1, \ldots, i_k$ be the k-blocks the equation $e_j \in E$ depends on. Let α_j be its vector of coefficients.
2. Run the linearity tester on f_i, Qf_i for all $i \in \{i_1, \ldots, i_k\}$.
3. Run the tensor tester on f_i, Qf_i for all $i \in \{i_1, \ldots, i_k\}$.
4. Run the self-correction constraint tester on f_i, Qf_i for all $i \in \{i_1, \ldots, i_k\}$ to "read off" $Qf_i(\alpha_{j,i})$. Namely, for each $i \in \{1, \ldots, i_k\}$ let $\alpha_{j,i} = \alpha_j|_{X_i^2}$, select $x \in \mathbb{F}_q^{X_i^2}$ randomly and take $a_i = Qf_i(x + \alpha_{j,i}) - Qf_i(x)$.
5. Check that $\sum_{i \in \{i_1, \ldots, i_k\}} a_i = c_j$.

We note that the total number of queries made in the above test is $3k + 4k + 2k = O(k) = O(1)$, so we have our desired number of queries. It is also clear that the reduction is polynomial time.

The following lemma shows the correctness of the reduction, thereby finishing the proof of Theorem 1.2.

Lemma 3.2. *Let (X, E) be a quadratic solvability instance satisfying the (k, s)-block property, and consider the constraint satisfaction graph problem Ψ constructed above. Then for all $\varepsilon > 0$ there is $\delta = \delta(k, \varepsilon) > 0$ such that the following holds:*

1. *If (X, E) is fully satisfiable, then Ψ is fully satisfiable.*
2. *If (X, E) is at most $(1 - \varepsilon)$-satisfiable, then Ψ is at most $(1 - \delta)$-satisfiable.*

Proof. For the first item, if (X, E) is satisfiable, then we take a satisfying assignment $v \in \mathbb{F}_q^X$ and set f_i, Qf_i to be the Hadamard and quadratic Hadamard encoding of $v|_{X_i}$ for all i; then the above tester passes with probability 1.

For the second item, we choose $\delta = \min\left(\frac{\varepsilon^2}{100}, \frac{1}{(2k+1)^2}\right)$ and assume counter-positively that there are f_i, Qf_i that pass this test with probability at least $1-\delta$. We show how to construct an assignment to (X, E) that satisfies at least $1-\sqrt{\delta} > 1-\varepsilon$ of the equations, and contradiction.

For each $i = 1, \ldots, k$, consider f_i and choose $v_i \in \mathbb{F}_q^{X_i}$ such that the function h_{v_i} is closest to f_i among all functions of the form h_v (if there are ties, break them arbitrarily). Thus, we have the vectors $v_1, \ldots, v_k$, and using them we can define an assignment $A\colon X \to \mathbb{F}_q$ where $A(x) = v_i(x)$ if $x \in X_i$; note that this is well defined since the X_i's are disjoint. In the rest of the argument, we show that the assignment A satisfies at least $1-\sqrt{\delta}$ of the equations of (X, E).

By an averaging argument, for at least $1-\sqrt{\delta}$ fraction of the j's, once we fix j the test passes with probability at least $1-\sqrt{\delta}$, and we show that A satisfies e_j for each such j. Indeed, fix such j; then we get that as the linearity tester+tensor tester pass on each one of $i = i_1, \ldots, i_k$ with probability at least $1-\sqrt{\delta}$, Lemma 2.6 implies there is u_i such that f_i is $2\sqrt{\delta}$-close to h_{u_i} and Qf_i is $2\sqrt{\delta}$-close to $h_{u_i \otimes u_i}$.

By the choice of v_i as we must have that $v_i = u_i$. Indeed, otherwise as h_{v_i} is $(1-1/q)$-far from h_{u_i}, it is at least $1 - 1/q - 2\sqrt{\delta} > 2\sqrt{\delta}$ far from f_i in contradiction to it being the closest.

Thus, with probability at least $1 - 2k\sqrt{\delta}$ we have that $Qf_i(x + \alpha_{j,i}) = h_{v_i \otimes v_i}(x + \alpha_{j,i})$ and $Qf_i(x) = h_{v_i \otimes v_i}(x)$ for all i in the last step, so with probability at least $1 - (2k+1)\sqrt{\delta}$ we get the a_i's there sum up to c_j and

$$a_i = Qf_i(x + \alpha_{j,i}) - Qf_i(x) = h_{v_i \otimes v_i}(x + \alpha_{j,i}) - h_{v_i \otimes v_i}(x) = h_{v_i \otimes v_i}(\alpha_{j,i}).$$

Summing over i gives that with probability at least $1 - (2k+1)\sqrt{\delta} > 0$ we have

$$c_j = \sum_{i \in \{i_1, \ldots, i_k\}} a_i = \sum_{i \in \{i_1, \ldots, i_k\}} a_i = h_{v_i \otimes v_i}(\alpha_{j,i}),$$

implying that A satisfies the equation e_j. □

4 The Composition Technique

The idea we presented here is an instantiation of a technique called composition, which is analogous to composition (concatenation) of error correcting codes. We started with a PCP with relatively large number of queries, which will be referred to as the "outer PCP" and composed it with a PCP with much fewer queries which will be referred to as the "inner PCP". In our case, the outer PCP was the PCP construction from Theorem 3.1, and inner PCP was the inefficient PCP that can be constructed using the Hadamard code. For our overall construction to be efficient though, we made sure to use the inner PCP only on small enough pieces, and the main gain is that the number of queries is basically inherited from the inner PCP.

At a high level, our composed PCP construction wanted to check that some constraint in the outer PCP was satisfied (a quadratic equation), and to do that we needed to "push down" this check to the language of the inner PCP (in our case, the quadratic Hadamard code enables us to check quadratic equations).

Composition of PCPs is one of the most important ideas in PCP theory; for example, going from the $\mathsf{poly}(\log n)$-query PCP theorem we have already proved to Theorem 1.1 amounts to composing PCPs too; in that case, it is a composition of the algebraic PCP (i.e. the one constructed via sum-check and low-degree testing) with itself.

As seen here, to facilitate composition one needs to have the block property, and in the next lecture we will discuss the aggregation of queries technique that achieves that.

18.408 Topics in Theoretical Computer Science: Probabilistically Checkable Proofs
Fall 2022