Electronic Colloquium on Computational Complexity, Report No. 92 (2019)

REVISITING ALPHABET REDUCTION IN DINUR'S PCP

VENKATESAN GURUSWAMI[†], JAKUB OPRŠAL[‡], AND SAI SANDEEP[†]

Abstract. Dinur's celebrated proof of the PCP theorem alternates two main steps in several iterations: gap amplification to increase the soundness gap by a large constant factor (at the expense of much larger alphabet size), and a composition step that brings back the alphabet size to an absolute constant (at the expense of a fixed constant factor loss in the soundness gap). We note that the gap amplification can produce a Label Cover CSP. This allows us to reduce the alphabet size via a direct long-code based reduction from Label Cover to a Boolean CSP. Our composition step thus bypasses the concept of Assignment Testers from Dinur's proof, and we believe it is more intuitive — *it is just a gadget reduction.* The analysis also uses only elementary facts (Parseval's identity) about Fourier Transforms over the hypercube.

1. INTRODUCTION

Constraint Satisfaction Problem (CSP) is a canonical NP-complete problem. Assuming $P \neq NP$, no polynomial time algorithm can find a satisfying assignment to a satisfiable CSP instance. If we are happy with the easier goal of satisfying a $1 - o(1)$ fraction of constraints, does there exist an efficient algorithm to do so? Answering this in the negative, the fundamental PCP theorem [ALM+98, AS98] implies that for some fixed integers $k, q \geq 2$ and $c < 1$, it is NP-hard to find an assignment satisfying a fraction c of constraints in a satisfiable CSP of arity k over alphabet $\{0, 1, \ldots, q-1\}$. Further this result holds for the combinations $(q, k) = (2, 3)$ and $(3, 2)$. The PCP theorem lies at the center of a rich body of work that has yielded numerous inapproximability results, including many optimal ones.

The PCP theorem was originally proved using algebraic techniques such as the low-degree test and the sum-check protocol. In a striking work, Dinur [Din07] gave an alternate combinatorial proof of the PCP theorem. Her proof works by amplifying the 'Unsat value' of a CSP instance — the fraction of constraints any assignment should violate. The goal is to show that it is NP-hard to distinguish if the Unsat value of a CSP instance is equal to 0 or at least a constant $c > 0$. Starting with a NP-hard problem such as 3-coloring with m constraints, we can already deduce that it is NP-hard to identify if Unsat value is equal to 0 or at least $1/m$. The Unsat value is increased slowly and iteratively via two steps — *gap amplification* and *alphabet reduction.* In gap amplification, we incur a constant factor blow up in the size of the instance, and get a constant factor improvement in the Unsat value. However, this step also blows up the

[†]Computer Science Department, Carnegie Mellon University, Pittsburgh, USA.
[‡]Computer Science Department, Durham University, Durham, UK.
E-mail addresses: venkatg@cs.cmu.edu, jakub.oprsal@durham.ac.uk, spallerl@andrew.cmu.edu.

The first author was supported in part by NSF grants CCF-1526092 and CCF-1814603. Most of the work has been done during a visit of the second author to Carnegie Mellon University. The second author has also received funding from the UK EPSRC grant EP/R034516/1. The third author was supported in part by NSF grant CCF-1422045.

ISSN 1433-8092

alphabet size. To alleviate this, alphabet reduction brings back the alphabet size to an absolute constant while losing a constant factor in the Unsat value (and blows up the instance size by a constant factor). Combining both the steps, we can increase the Unsat value by a constant factor (say 2) incurring a constant factor blow up in the size of the instance. Repeating this $\log m$ times proves the PCP theorem.

In this paper, we revisit the alphabet reduction step. Alphabet reduction is achieved by a method called *composition*, where we compose the 'outer' CSP over a large alphabet with an 'inner' PCP that is already proved to exist. In approximation view, this is equivalent to a gadget reduction where we replace a single constraint over the original larger alphabet by a set of several new constraints over a fixed, smaller alphabet. This is applied to every constraint in the original CSP instance. The key issue with this is the *consistency check* — we need a way to refer to the *same* assignment for original variables in different constraints. Dinur achieves this consistency by combining a good error correcting code with an inner PCP with a stronger property called Assignment Testers, introduced earlier in [DR06] and independently under the name probabilistically checkable proofs of proximity in [BSGH$^+$06].

Our main contribution here is a more direct proof of alphabet reduction. We start with an observation that the resulting CSP after gap amplification has very structured constraints. Specifically, it has the structure of *Label Cover* (not necessarily bipartite) with rectangular constraints, i.e., an arity two CSP whose relations are a disjoint union of rectangles.[1] Therefore, we do not need a general alphabet reduction procedure — instead, it suffices to focus on the case when the outer CSP has the Label Cover structure, and is over a fixed, albeit large, alphabet. We then follow the influential Label Cover and Long Code framework, originally proposed in [BGS98] and strengthened in [Hås01], to reduce the CSP obtained from Gap amplification, now viewed as Label Cover, to a CSP over a much smaller fixed alphabet (in fact a Boolean CSP).

Our main result is the following, which can be viewed as reproving a special case of alphabet reduction from [DR06, Din07].

Theorem 1.1. *There is a polynomial time reduction from Label Cover with rectangular constraints with soundness $1 - \delta$ to a fixed template CSP with soundness $1 - C\delta$ for an absolute constant $C > 0$.*

We analyse our reduction using Fourier analysis as pioneered by Håstad [Hås01]. Usually, in this framework, we reduce from low soundness Label Cover to strong (and at times optimal) soundness of CSP. But here we start with a high soundness Label Cover, and we reduce to high soundness CSP.

We highlight a couple of differences from previous works that make our proof easier:

- We have the freedom to choose any CSP rather than trying to prove inapproximability of a CSP. We choose the following 4-ary predicate R in our reduction: $(u, v, w, x) \in R$ if and only if $u \neq v \vee w \neq x$. This predicate appears in [Hås01] in the context of proving optimal hardness for NAE-4SAT.

[1] We should remark here that while it is convenient that the gap amplification produces such constraints, this is not really crucial as there is a standard reduction (the constraint-variable Label-coverization) from any CSP of fixed arity to Label Cover with a constant factor loss in soundness gap, see Section 4.

- In [Hås01] and [BGS98], the objective is to prove optimal inapproximability, or at least to get good soundness. However, our present goal is to prove 'just' a nontrivial soundness. (On the other hand, we also start with high soundness Label Cover.) This allows us to use a very convenient test distribution leading to a simple analysis. We remark that a similar statement as Theorem 1.1 can be also deduced using [BGS98, Section 4.1.1]. We believe that the test presented in this paper is more direct since we benefit from ideas in [Hås01].

Composition is an essential step in both the original proof of the PCP theorem as well as Dinur's proof and deserves further attention. Our proof of composition bypasses the concept of Assignment Testers and is more intuitive in our opinion as it is *nothing but a gadget reduction.* Our proof is elementary using only Parseval's identity from Fourier Analysis over the hypercube. Dinur's analysis used the Friedgut-Naor-Kalai theorem [FKN02] about Boolean functions with most of the Fourier mass at level 1.

It is possible to also perform alphabet reduction using the Hadamard code instead of the long code as described in [GO05, RS07]; the latter [RS07] also avoids explicit use of Assignment Testers. Long code tests correspond exactly to testing whether a function is a *polymorphism* of the CSP, and as such corresponds to gadget reductions in the algebraic approach to CSP (see e.g. [BKW17]). The PCP theorem surpasses these algebraic (gadget) reductions; this is even more evident when extending the scope from CSPs to *promise constraint satisfaction problems* (PCSPs) as there are PCSPs that can be shown to be NP-hard by using PCP theorem via a natural reduction through Label Cover, but cannot be shown to be NP-hard using only algebraic reductions [AGH17, BKO19]. In this sense, the present paper shows that this strength of the PCP theorem comes from the gap-amplification step. We remark that the reduction from Label Cover to PCP that we present in Section 3 follows the one presented in [BKO19, Section 3.3] with certain alternations that are necessary for the analytic measurement of the gap in contrast to the algebraic notions of [BKO19].

Outline. We start by formally defining CSP, Label Cover, and other preliminaries in Section 2. Then, in Section 3, we prove the main reduction from Label Cover to CSP. In Section 4, we show how the reduction can be used in the composition step of Dinur's proof.

2. PRELIMINARIES

2.1. CSPs and Label Cover

Loosely speaking, a CSP over some domain Σ is a decision problem which gets as input a set of variables and a set of constraints on the values of these variables. The goal is to decide whether there is an assignment of values from Σ to the variables such that all the constraints are satisfied. Usually, the shape of the constraints is somehow restricted. We give a formal definition of CSP for templates with a single relation (i.e., the constraints are restricted to be of the form $(x_1, \dots, x_k) \in R$ for a fixed relation $R \subseteq \Sigma^k$).

Definition 2.1 (CSP). Fix a domain Σ and a relation $A \subseteq \Sigma^k$. The constraint satisfaction problem associated with A, denoted by CSP(A), takes input as a set of variables $V = \{x_1, x_2, \cdots, x_n\}$ and a set of k-tuples of variables (constraints) $\{(x_{i_1}, \dots, x_{i_k}), \dots\}$. The goal is to decide whether there exists an asignment $s\colon V \to \Sigma$ such that for each constraint $(x_{i_1}, \dots, x_{i_k})$ we have $(s(x_{i_1}), \dots, s(x_{i_k})) \in A$.

For a Boolean domain $\Sigma = \{\textsc{true}, \textsc{false}\}$, we allow constraints to involve either variables or their negations.

Binary CSPs are traditionally referred to as Label Cover.

Definition 2.2 (Label Cover)**.** In an instance of Label Cover problem, we are given a tuple $(G = (V, E), \Sigma, \Pi)$ where

(1) G is a graph on vertex set V.
(2) Each vertex in G has to be assigned a label from Σ.
(3) For each edge $e = (u, v) \in E$, there is a relation $\Pi_e \subseteq \Sigma \times \Sigma$. This corresponds to a constraint between u and v.

A labelling of graph is a function $s\colon V \to \Sigma$ that assigns a label to each vertex of G. Such labelling is said to satisfy a constraint e if and only if $(s(u), s(v)) \in \Pi_e$.

For a Label Cover instance or in general for a CSP instance I, we use size(I) to denote $m + n$, where m is the number of constraints and n is the number of variables. We remark that Label Cover usually refers to the case when G is bipartite, and the constraint relations are functions. However, in this work, we find it convenient to consider a (closely related) version which has *rectangular relations*.

Definition 2.3 (Rectangular relation)**.** A relation $R \subseteq A \times B$ is said to be rectangular if there is a set C and functions $\pi\colon A \to C$ and $\sigma\colon B \to C$ such that $(a, b) \in R$ if and only if $\pi(a) = \sigma(b)$. Equivalently, R is rectangular if for all $a, a' \in A$ and $b, b' \in B$ such that $(a, b) \in R$, $(a, b') \in R$, and $(a', b) \in R$, we have $(a', b') \in R$.

2.2. Algebraic approach

Although not needed for our proofs, we include a definition of a polymorphism and some intuition from the algebraic approach to CSP. We refer to [BKW17] for a more detailed exposition.

Definition 2.4 (Polymorphism)**.** Let $R \subseteq \Sigma^k$ be a relation. We say that a function $f : \Sigma^n \to \Sigma$ is a *polymorphism* of R if for all tuples $(a_{11}, \dots, a_{k1}), \dots, (a_{1n}, \dots, a_{kn}) \in R$, we have

$$(f(a_{11}, \dots, a_{1n}), \dots, f(a_{k1}, \dots, a_{kn})) \in R.$$

We remark that the property 'f is a polymorphism of R' can be expressed as a CSP(R) with variables corresponding to all values of f (i.e., an instance with $|\Sigma|^n$ variables).

Our proof uses a Boolean CSP with the template given by the 4-ary relation

$$R = \{(x, x', z, z') \mid x \neq x' \vee z \neq z'\}.$$

A useful exercise is to show that each polymorphism of R is a *projection* (a.k.a. *dictator*), i.e., if $f : \{0, 1\}^n \to \{0, 1\}$ is a polymorphism of R then there exists $i \in [n]$ such that $f(x) = x(i)$ or $f(x) = \neg x(i)$ for all x.

Projections can be viewed as code words of the so-called *long code*. The long code is an error-correcting code that encodes a value $i \in [n]$ into a sequence of bits p_i of length 2^n. Such a sequence can be viewed as a function $p_i\colon \{0, 1\}^n \to \{0, 1\}$, and is defined by $p_i(x) = x(i)$.

We also remark that in the conjunction with the long-code, a rectangular constraint can be expressed as an identity. More precisely, given a rectangular relation $R \subseteq [n] \times [m]$, say $R = \{(i,j) : \pi(i) = \sigma(j)\}$ for some $\pi\colon [n] \longrightarrow [k]$ and $\sigma\colon [m] \longrightarrow [k]$, then the long codes p_i and p_j of values i, j satisfy

$$p_i(x_{\pi(1)}, \ldots, x_{\pi(n)}) = p_j(x_{\sigma(1)}, \ldots, x_{\sigma(m)})$$

for all $x_1, \ldots, x_k \in \{0,1\}$ if and only if $(i,j) \in R$. This is a key property of rectangular relations that is used implicitly in our proof.

2.3. **Boolean Fourier analysis**

As usual in Boolean Fourier analysis, we treat TRUE as -1 and FALSE as $+1$. In particular, in this notation, $\neg x = -x$ and $x \oplus y = xy$. Further, we define

$$x \vee y = \begin{cases} -1 & \text{if } x = -1 \text{ or } y = -1, \text{ and} \\ 1 & \text{otherwise.} \end{cases}$$

We define an inner product space on functions from $\{-1,+1\}^n \longrightarrow \mathbb{R}$ as $\langle f, g\rangle = \mathbb{E}_x[f(x)g(x)]$. For a set $S \subseteq [n]$, let

$$\chi_S(x_1, \ldots, x_n) = \prod_{i \in S} x_i.$$

The set of such functions form an orthonormal basis for all functions from $\{-1,+1\}^n$ to $\mathbb{R}$ in the above defined inner product space. Moreover, if $S \neq \emptyset$, then $\mathbb{E}_x[\chi_S(x)] = 0$.

Definition 2.5 (Fourier expansion)**.** Given a function $f\colon \{-1,+1\}^n \longrightarrow \mathbb{R}$, we can thus write it uniquely as a linear combination of this basis—

$$f = \sum_S \hat{f}(S)\chi_S$$

The real quantities $\hat{f}(S)$ are called the *Fourier coefficients* of f. We abuse the notation $\hat{f}(i)$ to denote $\hat{f}(\{i\})$.

The following simple but crucial identity follows from the definitions and is all that we will need in our analysis.

Theorem 2.6 (Parseval's Identity)**.** *For each Boolean valued function f, i.e., $f\colon \{-1,+1\}^n \longrightarrow \{-1,+1\}$,*

$$\sum_S \hat{f}(S)^2 = 1.$$

We remark that $\chi_{\{i\}}$ is the projection p_i as defined above, and that the natural distance defined by the scalar product $\langle f, g\rangle$ on Boolean functions corresponds to *relative Hamming distance.* This is thanks to the fact that if $x, y \in \{-1, 1\}$ then $x = y$ if and only if $xy = 1$, and consequently, the relative Hamming distance of f to the corresponding long code word can be expressed as $\bigl(1 - \hat{f}(i)\bigr)/2$.

3. LABEL COVER TO CSP

This section describes our gadget reduction from Label Cover to the CSP with the Boolean domain using the following 4-ary relation

$$R = \{(x, x', z, z') \mid x \neq x' \vee z \neq z'\}.$$

We note that since we employ a standard method of *folding*, the produced CSP instance allows for negation of variables in the constraints, e.g. constraints such as $(x_1, \neg x_2, x_3, \neg x_4) \in R$ and $(x_1, \neg x_2, x_3, x_4) \in R$ are allowed.

Theorem 3.1. *There exists absolute constant C such that given a Label Cover instance (not necessarily bipartite) $G = (V, E, \Pi)$ with rectangular constraints, there is a reduction from G that outputs an instance I of* $\mathrm{CSP}(R)$ *such that* $\mathrm{size}(I) = O(\mathrm{size}(G))$ *and*

- *If G is satisfiable, then I is satisfiable as well.*
- *If no labelling can satisfy $1 - \delta$ fraction of constraints of G, then no assignment can satisfy $1 - C\delta$ fraction of constraints in I for all δ.*

The reduction can be described as a probabilistic checker of a solution to G encoded using a long code, i.e., the proof contains for each $u \in V$ a word $f_u \colon \{-1, 1\}^{|\Sigma|} \to -1, 1$. The test is then as follows: Sample an edge $e = (u, v)$ from E uniformly at random, and then with equal probability do one of the following:

(1) run a long code test inside f_u;
(2) run a long code test inside f_v;
(3) run a constraint test between f_u and f_v.

We describe the long code test and constraint test below. Both query the respective tables of f_u and f_v at some 4 bits that are generated by a certain randomized algorithm, and then check whether these 4 Boolean values satisfy the predicate R defined above.

This checker can be viewed as a gadget reduction in the following sense: We replace each vertex $u \in V$ with $2^{|\Sigma|}$ Boolean variables labelled by $f_u(x)$ for $x \in \{-1, 1\}^{|\Sigma|}$ (we see an assignment to such variables as a function $f_u \colon \{-1, 1\}^{|\Sigma|} \to \{-1, 1\}$), and each edge $e = (u, v)$ with a set of weighted 4-ary constraints on f_u and f_v, each involving the relation R and some 4 values of f_u and f_v (the result is therefore an instance of $\mathrm{CSP}(R)$). These constraints depend only on the relation Π_e.

To simplify some notation, we assume $\Sigma = [n]$. We also assume that the tables for f_u's are *folded* so f_u is forced to satisfy $f(-x) = -f(x)$. This is a standard technique. Such a folding is ensured by including only one variable of each pair $x, -x$, and if the test queries f_u at the bit corresponding to some x that is not included, the bit $f(-x)$ is queried instead, and the value is negated. As a consequence of this folding, we have to allow for negation of variables in $\mathrm{CSP}(R)$. An important and useful consequence of this is that all 'even' Fourier coefficients of f vanish, i.e., $\hat{f}(\beta) = 0$ for all β such that $|\beta|$ is even. We remark that folding can be avoided in the construction of the gadget. Nevertheless, it considerably simplifies the calculations below. Further, for calculations, it is useful to view R as a predicate $\rho \colon \{\pm 1\} \to \{0, 1\}$ defined as

$$\rho(x, x', z, z') = 1 - \frac{(xx' + 1)(zz' + 1)}{4}.$$

It is easy to check that $\rho(x, x', z, z') = 1$ if and only if $(x, x', z, z') \in R$.

Let us now describe the two probabilistic checkers. The long code test has on input a table of a function f ($= f_u$), and it essentially tests whether f is a polymorphism of R. A naïve way would be to sample constraints for being a polymorphism uniformly at random. Although it is not obvious from the description below, we sample tuples from R in a non-uniform way which will simplify the acceptance probability of the test.

Long code test. Given $f\colon \{-1,1\}^n \to \{-1,1\}$ to test against being a long code word. Choose $x, y, z, \mu \in \{-1,1\}^n$ uniformly at random. Test whether

$$(f(x), f(x \oplus (\mu \vee y)), f(z), f(z \oplus (\mu \vee \neg y))) \in R. \tag{1}$$

Note that for each $x, y, z, \mu \in \{-1,1\}$, $(x, x\oplus(\mu\vee y), z, z\oplus(\mu\vee\neg y)) \in R$. This implies that any dictator function passes the test with probability 1. (The same is true about polymorphisms of R.)

Lemma 3.2. *Assuming that f is folded, the probability the long code test accepts is at most*

$$1 - \frac{3}{16}\sum_{|\alpha|>1} \hat{f}(\alpha)^2.$$

Proof. Assume $f(x) = \sum_\alpha \hat{f}(\alpha)\chi_\alpha(x)$. The probability that the test accepts is

$$\begin{aligned}
&\mathbb{E}_{x,y,z,\mu}\, \rho(f(x), f(x \oplus (\mu \vee y)), f(z), f(z \oplus (\mu \vee \neg y))) = \\
&\quad \mathbb{E}_{x,y,z,\mu}\left[1 - \frac{\big(f(x)f(x\oplus(\mu\vee y)+1\big)\big(f(z)f(z\oplus(\mu\vee\neg y))+1\big)}{4}\right] = \\
&\quad \frac{3}{4} - \frac{1}{4}\,\mathbb{E}_{x,y,\mu}\, f(x)f(x\oplus(\mu\vee y)) - \frac{1}{4}\,\mathbb{E}_{y,z,\mu}\, f(z)f(z\oplus(\mu\vee\neg y)) \\
&\qquad - \frac{1}{4}\,\mathbb{E}_{x,y,z,\mu}\, f(x)f(x\oplus(\mu\vee y))f(z)f(z\oplus(\mu\vee\neg y))
\end{aligned} \tag{2}$$

Let us attack this term by term.

$$\begin{aligned}
&\mathbb{E}_{x,y,\mu}\, f(x)f(x\oplus(\mu\vee y)) = \mathbb{E}_{x,y,\mu}\sum_{\alpha,\beta}\hat{f}(\alpha)\hat{f}(\beta)\chi_\alpha(x)\chi_\beta(x\oplus(\mu\vee y)) = \\
&\quad \mathbb{E}_{x,y,\mu}\sum_\alpha \hat{f}(\alpha)^2\chi_\alpha(y\vee\mu) = \sum_\alpha \hat{f}(\alpha)^2\,\mathbb{E}_{y,\mu}\,\chi_\alpha(y\vee\mu) = \sum_\alpha \hat{f}(\alpha)^2(-1/2)^{|\alpha|}.
\end{aligned} \tag{3}$$

The last equality follows from the fact that $\mathbb{E}_{y,\mu}[y\vee\mu] = (-1)\cdot 3/4 + 1\cdot 1/4 = -1/2$ and coordinates are chosen independently. Similarly, we get that

$$\mathbb{E}_{y,z,\mu}\, f(z)f(z\oplus(\mu\vee\neg y)) = \sum_\alpha \hat{f}(\alpha)^2(-1/2)^{|\alpha|}. \tag{4}$$

Moving to the next term, we get

$$\begin{aligned}
&\mathbb{E}_{x,y,z,\mu}\, f(x)f(x\oplus y\vee\mu)f(z)f(z\oplus(\neg y)\vee\mu) = \\
&\quad \sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{f}(\beta)^2\,\mathbb{E}_{y,\mu}\,\chi_\alpha(y\vee\mu)\chi_\beta((\neg y)\vee\mu) = \sum_{\alpha\cap\beta=\emptyset}\hat{f}(\alpha)^2\hat{f}(\beta)^2(-1/2)^{|\alpha\cup\beta|}.
\end{aligned} \tag{5}$$

The last equality follows since $\mathbb{E}_{y,\mu}[(\mu \vee y) \oplus (\mu \vee \neg y)] = \mathbb{E}_{y,\mu}[\neg\mu] = 0$ and $\mathbb{E}_{y,\mu}[\mu \vee y] = \mathbb{E}_{y,\mu}[\mu \vee \neg y] = -1/2$. The overall acceptance probability is then

$$(6)\quad \frac{3}{4} - \frac{1}{2}\sum_{\alpha} \hat{f}(\alpha)^2(-1/2)^{|\alpha|} - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(-1/2)^{|\alpha\cup\beta|}$$
$$= 1 - \frac{1}{2}\sum_{\alpha}\hat{f}(\alpha)^2\big((-1/2)^{|\alpha|} + 1/2\big) - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset}\hat{f}(\alpha)^2\hat{f}(\beta)^2(-1/2)^{|\alpha\cup\beta|}$$

where for the last equality, we used Parseval's identity. Further, we assumed that f is folded, and therefore $\hat{f}(\alpha) = 0$ for all α such that $|\alpha|$ is even. Restricting the sums to α and β with odd cardinality, and using that for such disjoint α and β, $|\alpha \cup \beta|$ is even, the last expression of (6) can be bounded from above by

$$(7)\quad 1 - \frac{1}{2}\sum_{|\alpha|>1} \hat{f}(\alpha)^2\,(3/8) - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(1/2)^{|\alpha\cup\beta|}$$
$$\le 1 - \frac{3}{16}\sum_{|\alpha|>1}\hat{f}(\alpha)^2 - \frac{1}{16}\sum_{i\neq j}\hat{f}(i)^2\hat{f}(j)^2 \le 1 - \frac{3}{16}\sum_{|\alpha|>1}\hat{f}(\alpha)^2$$

which concludes the proof. □

We move on to the constraint test. The constraint test has on input tables for functions f and g corresponding to some u and v such that $(u, v) \in E$, and it is supposed to test (assuming f and g are correct long code words) whether the values these functions encode satisfy the constraint given by a rectangular relation Π_e. We assume that this relation is given by $\pi, \sigma\colon [n] \to [m]$ such that $(i, j) \in \Pi_e$ if and only if $\pi(i) = \sigma(j)$.

The constraints tests two given functions $f, g\colon \{-1, 1\}^n \to \{-1, 1\}$ against satisfying

$$(8)\qquad f(x_{\pi(1)}, \dots, x_{\pi(n)}) = g(x_{\sigma(1)}, \dots, x_{\sigma(n)})$$

for all $x_1, \dots, x_m \in \{-1, 1\}$. One may check that if both f and g are dictators, say $f(x) = x(i)$ and $g(x) = x(j)$, then they satisfy (8) if and only if $\pi(i) = \sigma(j)$.

We denote by y^π the vector in $\{-1, 1\}^n$ such that $y^\pi(i) = y(\pi(i))$. The identity (8) can be then expressed as $f(x^\pi) = g(y^\sigma)$ for all $x, y \in \{-1, 1\}^m$.

Constraint test. Choose $x, z \in \{-1, 1\}^n$ and $y \in \{-1, 1\}^m$ uniformly at random, and test whether

$$(9)\qquad (f(x), f(x \oplus y^\pi), g(z), g(z \oplus (\neg y)^\sigma)) \in R.$$

In the analysis below, we will use the following notation. Let $\alpha \subseteq [n]$ and $\pi\colon [n] \to [m]$, we denote by $\pi[\alpha]$ the subset of $[m]$ defined by

$$\pi[\alpha] = \{k : |\pi^{-1}(k) \cap \alpha| \text{ is odd}\}.$$

Note that, for each $x_1, \dots, x_m \in \{-1, 1\}$,

$$(10)\qquad \prod_{i\in\alpha} x_{\pi(i)} = \prod_{i\in\pi[\alpha]} x_i$$

which implies that $\chi_\alpha(x^\pi) = \chi_{\pi[\alpha]}(x)$. Consequently, we get that χ_α and χ_β satisfy (8) if and only if $\pi[\alpha] = \sigma[\beta]$.

Lemma 3.3. *Given that both f and g are folded, the probability that the consistency test accepts is at most*

$$1 - \frac{1}{4} \sum_{i,j:\pi(i)\neq\sigma(j)} \hat{f}(i)^2 \hat{g}(j)^2.$$

Proof. We can compute the acceptance probability in the same way as before, i.e., as

$$\frac{3}{4} - \frac{1}{4}\mathbb{E}_{x,y}[f(x)f(x\oplus y^{\pi})] - \frac{1}{4}\mathbb{E}_{z,y}[g(z)g(z\oplus(\neg y)^{\sigma})] - \frac{1}{4}\mathbb{E}_{x,y,z}[f(x)f(x\oplus y^{\pi})g(z)g(z\oplus(\neg y)^{\sigma})] \tag{11}$$

We have

$$\mathbb{E}_{x,y}[f(x)f(x\oplus y^{\pi})] = \sum_{\alpha}\hat{f}(\alpha)^2\,\mathbb{E}_y[\chi_\alpha(y^{\pi})] = \sum_{\alpha}\hat{f}(\alpha)^2\,\mathbb{E}_y[\chi_{\pi[\alpha]}(y)] = 0 \tag{12}$$

where the last equality holds since $|\alpha|$ is odd, and consequently $\pi[\alpha]\neq\emptyset$. Similarly, $\mathbb{E}_{x,z}[g(z)g(z\oplus(\neg y)^{\sigma})]$ vanishes. Thus the probability that the test accepts is

$$\begin{aligned}
&\frac{3}{4} - \frac{1}{4}\mathbb{E}_{x,y,z}\, f(x)f(x\oplus y^{\pi})g(z)g(z\oplus(\neg y)^{\sigma}) \\
&= \frac{3}{4} - \frac{1}{4}\sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{g}(\beta)^2\,\mathbb{E}_y[\chi_\alpha(y^{\pi})\chi_\beta(-y^{\sigma})] = \frac{3}{4} + \frac{1}{4}\sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{g}(\beta)^2\,\mathbb{E}_y[\chi_\alpha(y^{\pi})\chi_\beta(y^{\sigma})] \\
&= \frac{3}{4} + \frac{1}{4}\sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{g}(\beta)^2\,\mathbb{E}_y[\chi_{\pi[\alpha]}(y)\chi_{\sigma[\beta]}(y)] = \frac{3}{4} + \frac{1}{4}\sum_{\alpha,\beta:\pi[\alpha]=\sigma[\beta]}\hat{f}(\alpha)^2\hat{g}(\beta)^2 \\
&= 1 - \frac{1}{4}\sum_{\alpha,\beta:\pi[\alpha]\neq\sigma[\beta]}\hat{f}(\alpha)^2\hat{g}(\beta)^2
\end{aligned} \tag{13}$$

where the second equality follows from $|\beta|$ being odd, and the last equality follows from the Parseval's identity. Since $\pi(i)=\sigma(j)$ implies that $\pi[\{i\}]=\sigma[\{j\}]$, the claim follows. □

Lemma 3.4. *Given that both f and g are folded, the probability that the joint test accepts is at most*

$$1 - \frac{1}{16}\Big(\sum_{|\alpha|>1} f(\alpha)^2 + \sum_{|\beta|>1} g(\beta)^2 + \sum_{i,j:\pi(i)\neq\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2\Big)$$

Proof. Follows directly from Lemmas 3.2 and 3.3. □

Finally, we are ready to prove the main theorem of this section.

Proof of Theorem 3.1. The completeness is straightforward. We prove the soundness. Suppose that the test passes with probability $1-\delta$. We will show that this implies that there is an assignment to the Label Cover instance that satisfies $(1-16\delta)$-fraction of constraints.

Our decoding is as follows: for a node $v\in V$, decode v to $i\in\Sigma$ with probability proportional to $\hat{f}_v(i)^2$. We will show that in expectation, this decoding satisfies at least $1-16\delta$ fraction of constraints, which proves that there exists a labelling that satisfies at least $1-16\delta$ fraction of constraints.

Let $1-\delta_e$ denote the probability that the test passes when we pick edge e. As test passes with probability $1-\delta$, we know that $\mathbb{E}_e[\delta_e]=\delta$. Suppose that we pick $e=(u,v)$ with f and

g being the functions corresponding to u and v respectively. From the above lemma, we have that

$$1 - \delta_e \leq 1 - \frac{1}{16}\Big(\sum_{|\alpha|>1} \hat{f}(\alpha)^2 + \sum_{|\beta|>1} \hat{g}(\beta)^2 + \sum_{i,j:\pi(i)\neq\sigma(j)} \hat{f}(i)^2\hat{g}(j)^2\Big), \tag{14}$$

and therefore,

$$16\delta_e \geq \sum_{|\alpha|>1} \hat{f}(\alpha)^2 + \sum_{|\beta|>1} \hat{g}(\beta)^2 + \sum_{i,j:\pi(i)\neq\sigma(j)} \hat{f}(i)^2\hat{g}(j)^2. \tag{15}$$

The probability that our decoding satisfies edge e of Label Cover is at least

$$\begin{aligned} \sum_{i,j:\pi(i)=\sigma(j)} \hat{f}(i)^2\hat{g}(j)^2 &= 1 - \sum_{\substack{\alpha,\beta \\ |\alpha|>1 \text{ or } |\beta|>1 \text{ or} \\ \alpha=\{i\} \text{ and } \beta=\{j\} \text{ but } \pi(i)\neq\sigma(j)}} \hat{f}(\alpha)^2\hat{g}(\beta)^2 \\ &\geq 1 - \sum_{\alpha,\beta:|\alpha|>1} \hat{f}(\alpha)^2\hat{g}(\beta)^2 - \sum_{\alpha,\beta:|\beta|>1} \hat{f}(\alpha)^2\hat{g}(\beta)^2 - \sum_{i,j:\pi(i)\neq\sigma(j)} \hat{f}(i)^2\hat{g}(j)^2 \\ &= 1 - \sum_{|\alpha|>1} \hat{f}(\alpha)^2 - \sum_{|\alpha|>1} \hat{g}(\alpha)^2 - \sum_{i,j:\pi(i)\neq\sigma(j)} \hat{f}(i)^2\hat{g}(j)^2 \geq 1 - 16\delta_\epsilon \end{aligned} \tag{16}$$

where the first equality follows from Perseval's identity. Thus, the expected number of constraints satisfied by the labelling is at least $\mathbb{E}_e[1 - 16\delta_e] = 1 - 16\delta$ which proves the required claim with $C = 1/16$. □

3.1. Derandomization

With a little additional argument, we can derandomize the decoding used above. Instead of decoding to i with probability $\hat{f}(i)^2$, we simply decode to the i with the largest $\hat{f}(i)^2$. We set i_f to be such i. In this light, the reduction can be analysed by analysing completeness and soundness of the gadget separately without considering the rest of the instance. The following lemma then shows that the gadget has perfect completeness and soundness 99% not depending on the parameters n, m, π and σ.

Lemma 3.5. *There is a gadget with inputs $n, m, k, \pi : [n] \to [k]$, and $\sigma : [m] \to [k]$ that produces an instance of* CSP(R) *with variables $f(a_1, \ldots, a_n)$ and $g(a_1, \ldots, a_m)$ such that*

1. *if $\pi(i) = \sigma(j)$ then p_i and p_j satisfy all the constraints, and*
2. *if at least 99% of the constraints are satisfied, then $\pi(i_f) = \sigma(i_g)$.*

Proof. Using (7) and Lemma 3.3, we know that the probability that the test accepts is at most

$$1-\frac{1}{16}\sum_{|\alpha|>1} \hat{f}(\alpha)^2-\frac{1}{48}\sum_{i\neq j}\hat{f}(i)^2\hat{f}(j)^2-\frac{1}{16}\sum_{|\alpha|>1}\hat{g}(\alpha)^2-\frac{1}{48}\sum_{i\neq j}\hat{g}(i)^2\hat{g}(j)^2-\frac{1}{12}\sum_{i,j:\pi(i)\neq\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2.$$

Given that the acceptance probability is at least 99% $> 1 - 1/96$, we get that

$$\sum_{|\alpha|>1} \hat{f}(\alpha)^2 \leq 1/6 \tag{17}$$

$$\sum_{i\neq j} \hat{f}(i)^2 \hat{f}(j)^2 \leq 1/2 \tag{18}$$

$$\sum_{|\alpha|>1} \hat{g}(\alpha)^2 \leq 1/6 \tag{19}$$

$$\sum_{i\neq j} \hat{g}(i)^2 \hat{g}(j)^2 \leq 1/2 \tag{20}$$

$$\sum_{i,j:\pi(i)\neq\sigma(j)} \hat{f}(i)^2 \hat{g}(j)^2 \leq 1/8 \tag{21}$$

From Parseval's identity and (17), we get that $1 \geq \sum_i \hat{f}(i)^2 \geq 5/6$. Recall that i_f is such i that $\hat{f}(i)^2$ is maximal. Then using the above and (18), we obtain that

$$\begin{aligned} \hat{f}(i_f)^2 \geq \hat{f}(i_f)^2 \sum_i \hat{f}(i)^2 \geq \sum_i \hat{f}(i)^4 = \sum_{i,j} \hat{f}(i)^2\hat{f}(j)^2 - \sum_{i\neq j} \hat{f}(i)^2 \hat{f}(j)^2 \\ \geq (5/6)^2 - 1/2 = 4/9. \end{aligned} \tag{22}$$

Similarly, from (19) and (20), we get $\hat{g}(i_g)^2 \geq 4/9$. Finally, since $\hat{f}(i_f)^2\hat{g}(i_g)^2 \geq (4/9)^2 > 1/8$, we have that $\pi(i_f) = \sigma(i_g)$ otherwise (21) cannot be true. □

Theorem 3.1 can be also directly obtained from this lemma albeit with a worse constant than in the above proof: Let $C = 1\%$ and assume that $\delta < 1$. Given that the resulting CSP instance has an assignment fails no more than $C\delta$-fraction of the constraints, we derive that in at least $(1 - \delta)$-fraction of the gadgets, no more than C-fraction of constraints are unsatisfied. Lemma 3.5 then shows that the assignment $s\colon u \mapsto i_{f_u}$ is an assignment of the Label Cover instance that satisfies all the constraints corresponding to these gadgets. This completes the proof.

4. CSP TO LABEL COVER

In this section, our goal is to show how the previous reduction can be used in the alphabet reduction step of Dinur's proof of PCP Theorem.

We first prove that the previous reduction can be combined with standard reductions to get back Label Cover from the CSP.

Theorem 4.1 (Alphabet reduction). *Given a Label Cover instance $G = (V, E, \Pi, \Sigma_0)$ with rectangular constraints, there is a polynomial time reduction that outputs another Label Cover instance with rectangular constraints $G' = (V', E', \Pi', \Sigma)$ with alphabet size Σ such that $|\Sigma|$ is an absolute constant, size$(G') = O($size$(G))$ and*

- *If G is satisfiable, then G' is satisfiable as well.*
- *If every labelling violates δ fraction of constraints of G, then every labelling violates $C\delta$ fraction of constraints in G' for an absolute constant C.*

Proof. We first convert the Label Cover instance G to a CSP instance I as in Theorem 3.1. The CSP instance can be converted to bipartite Label Cover using standard clause-variable Label-coverization technique. We include the proof here for the sake of completeness. We have n vertices $x_1, x_2, \dots, x_n$ corresponding to the variables of I on the left L, and m vertices corresponding to constraints $C_1, C_2, \dots, C_m$ of I on the right R. The label set is binary on the left, and satisfying assignments (at most 16) on the right corresponding to the possible assignments to four variables in the constraint. We add an edge between $u \in L$ and $v \in R$ if $x_u \in C_v$. The constraint on this edge enforces that the assignment to x_u is consistent with the assignment C_v assigns to x_u.

If there is a satisfying labelling to G, there is a satisfying assignment to I. Using this, we can assign the variables on the left the satisfying assignment, and the corresponding assignment to tuples for the vertices of constraints on the right, and thus get a satisfying assignment to G'. Suppose that every labelling violates at least δ fraction of constraints of G. From Theorem 3.1, every assignment violates at least $C\delta$ fraction of constraints in I. Suppose there is a labelling to G' that satisfies δ' fraction of constraints. Consider the assignment obtained by this labelling on the left. This assignment violates at least $C\delta m$ number of constraints in I. Note that this should violate at least $C\delta m$ constraints in G' and thus $\delta' \geq C'\delta$ for an absolute constant C'. The constraints are in fact projections, and thus are rectangular too. □

In order for us to use this as Composition step in the PCP of Dinur, we need the final observation that the output of Gap Amplification applied to a CSP with rectangular constraints results in a Label Cover with rectangular constraints. Dinur achieves gap amplification by 'graph powering' which is described more formally below.

Definition 4.2 (Constraint Graph Powering)**.** Given a d-regular Label Cover (a.k.a. Constraint graph) $G = (V, E, \Sigma, \Pi)$, we obtain tth power of it $G^t = (V, E', \Sigma', \Pi')$ as follows:

- Vertices. The vertices in G^t are the same as vertices in G.
- Edges. u and v are connected by k parallel edges in E' if there are exactly k paths between u and v in G.
- Alphabet. The alphabet of G^t is $\Sigma^{d^{\lceil t/2 \rceil}}$. A value $a \in \Sigma^{d^{\lceil t/2 \rceil}}$ is interpreted as assigning values $a \colon \Gamma(u) \to \Sigma$ to $d^{\lceil t/2 \rceil}$ elements from Σ. This value is treated as u's opinion on $\Gamma(u)$, the set of all vertices within $\lceil t/2 \rceil$ distance from u.
- Constraints. An edge $(u, v) \in E'$ is satisfied by $a, b \in \Sigma^{d^{\lceil t/2 \rceil}}$ if and only if the following holds: there is an assignment $\sigma \colon \Gamma(u) \cup \Gamma(v) \to \Sigma$ that satisfies every constraint $c(e)$ where $e \in E \cap (\Gamma(u) \times \Gamma(v))$, and such that

$$\forall u' \in \Gamma(u), \sigma(u) = a_{u'}; \forall v' \in \Gamma(v), \sigma(v) = b_{v'}$$

  where $a_{u'}$ (and respectively $b_{v'}$) is the value a (and resp. b) assigned to u' (and resp. v').

The output G^t is also a binary CSP, and hence can be viewed as a Label Cover. We claim that if every constraint of G is rectangular, then every constraint of G^t is rectangular as well. Let $e = (u, v)$ be an edge in E' with constraint relation as R_e. Suppose $(a, b), (a', b), (a, b') \in R_e$. This implies that for all $(u', v') \in E \cap (\Gamma(u) \times \Gamma(v))$ with constraint relation c_e,

$$(a_{u'}, b_{v'}), (a'_{u'}, b_{v'}), (a_{u'}, b'_{v'}) \in R_{c_e}.$$

Since R_{c_e} is rectangular, $(a'_{u'}, b'_{v'}) \in R_{c_e}$ as well. As this holds for all such u' and v', $(a', b') \in R_e$, thus proving that R_e is a rectangular relation.

Combined with the preprocessing step, the gap amplification theorem of Dinur can be rewritten as follows.

Theorem 4.3 (Gap amplification). *Fix a parameter t. Given a Label Cover $G = (V, E, \Pi, \Sigma)$ where Σ is an absolute constant, there is a polynomial time reduction to output a rectangular Label Cover instance $G' = (V', E', \Pi', \Sigma')$ with the alphabet size $|\Sigma'| = c(|\Sigma|, t)$ such that*

- *If G is satisfiable, G' is satisfiable as well.*
- *If every labelling violates at least δ fraction of the constraints of G, then every labelling violates at least $\Omega(\delta\sqrt{t})$ fraction of the constraints of G'.*

Choosing t large enough constant and iterating Theorem 4.1 and Theorem 4.3 $\log(m)$ times proves the PCP theorem.

Acknowledgments

We would like to thank Irit Dinur, Oded Goldreich, Prahladh Harsha, Johan Håstad, Jaikumar Radhakrishnan, and Madhu Sudan for useful comments and feedback.

References

[AGH17] Per Austrin, Venkatesan Guruswami, and Johan Håstad. $(2 + \epsilon)$-Sat is NP-hard. *SIAM J. Comput.*, 46(5):1554–1573, 2017.

[ALM$^+$98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.

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

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability-towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998.

[BKO19] Jakub Bulín, Andrei Krokhin, and Jakub Opršal. Algebraic approach to promise constraint satisfaction. In *Proceedings of the 51st Annual ACM SIGACT Symposium on the Theory of Computing (STOC '19)*, New York, NY, USA, 2019. ACM.

[BKW17] Libor Barto, Andrei Krokhin, and Ross Willard. Polymorphisms, and how to use them. In Andrei Krokhin and Stanislav Živný, editors, *The Constraint Satisfaction Problem: Complexity and Approximability*, volume 7 of *Dagstuhl Follow-Ups*, pages 1–44. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2017.

[BSGH$^+$06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM J. Comput.*, 36(4):889–974, 2006.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3):12, 2007.

[DR06] Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. *SIAM J. Comput.*, 36(4):975–1024, 2006.

[FKN02] Ehud Friedgut, Gil Kalai, and Assaf Naor. Boolean functions whose Fourier transform is concentrated on the first two levels. *Adv. in Applied Math.*, 29:427–437, 2002.

[GO05] Venkatesan Guruswami and Ryan O'Donnell. The PCP theorem and hardness of approximation: Notes on lectures 7–9. https://courses.cs.washington.edu/courses/cse533/05au/, 2005.

[Hås01] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

[RS07] Jaikumar Radhakrishnan and Madhu Sudan. On Dinur's proof of the PCP theorem. *Bull. Amer. Math. Soc.*, 44:19–61, 2007.